European Commission | Horizon 2020 European Union funding for Research & Innovation

Big Data technologies and extreme-scale analytics

MARVEL

Multimodal Extreme Scale Data Analytics for Smart Cities Environments

D3.4: MARVEL's federated learning realization [†]

Abstract: This document reports on MARVEL's federated learning realisation consisting of the component FedL and several novel federated learning methodologies that have been developed within the course of the project. First, FedL is described in general terms, and subsequently its MARVEL R1 realisation for visual crowd counting tasks, on the three MARVEL pilots: 1) GRN in Malta, for the use case GRN4 Junction traffic trajectory collection; 2) MT in Trento, for the use case MT1 Monitoring of crowded areas; and 3) the experimental pilot UNS in Novi Sad, for the use case UNS1 Drone experiment, for monitoring large space public events. Further, novel federated methodologies for model personalisation and clustering, decentralised and unsupervised anomaly detection, and, distributed inference and social learning are described, with particular implications and relevance for MARVEL. Finally, a novel strategy for the design of federated learning protocols based on the metric of large deviations is presented.

Contractual Date of Delivery	*31/12/2022*
Actual Date of Delivery	*15/01/2023*
Deliverable Security Class	*Public*
Editor	*Dragana Bajovic (UNS)*
Contributors	*UNS, CNR, AU, TAU, FBK, ITML, ATOS, GRN, MT*
Quality Assurance	*Alessio Brutti (FBK)* *Lukas Esterle (AU)*

[†] *The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957337.*

The *MARVEL* Consortium

Part. No.	Participant organisation name	Participant Short Name	Role	Country
1	FOUNDATION FOR RESEARCH AND TECHNOLOGY HELLAS	FORTH	Coordinator	EL
2	INFINEON TECHNOLOGIES AG	IFAG	Principal Contractor	DE
3	AARHUS UNIVERSITET	AU	Principal Contractor	DK
4	ATOS SPAIN SA	ATOS	Principal Contractor	ES
5	CONSIGLIO NAZIONALE DELLE RICERCHE	CNR	Principal Contractor	IT
6	INTRASOFT INTERNATIONAL S.A.	INTRA	Principal Contractor	LU
7	FONDAZIONE BRUNO KESSLER	FBK	Principal Contractor	IT
8	AUDEERING GMBH	AUD	Principal Contractor	DE
9	TAMPERE UNIVERSITY	TAU	Principal Contractor	FI
10	PRIVANOVA SAS	PN	Principal Contractor	FR
11	SPHYNX TECHNOLOGY SOLUTIONS AG	ST	Principal Contractor	CH
12	COMUNE DI TRENTO	MT	Principal Contractor	IT
13	UNIVERZITET U NOVOM SADU FAKULTET TEHNICKIH NAUKA	UNS	Principal Contractor	RS
14	INFORMATION TECHNOLOGY FOR MARKET LEADERSHIP	ITML	Principal Contractor	EL
15	GREENROADS LIMITED	GRN	Principal Contractor	MT
16	ZELUS IKE	ZELUS	Principal Contractor	EL
17	INSTYTUT CHEMII BIOORGANICZNEJ POLSKIEJ AKADEMII NAUK	PSNC	Principal Contractor	PL

Document Revisions & Quality Assurance

Internal Reviewers

1. Alessio Brutti, FBK
2. Lukas Esterle, AU

Revisions

Version	Date	By	Overview
0.5	15/01/2023	*Dragana Bajovic*	Addressed comments from the PC
0.4	14/01/2023	*Despina Kopanaki*	Comments from the PC
0.4	11/01/2023	*Dragana Bajovic*	Addressed IR comments and submitted to PC
0.3	10/01/2023	*Alessio Brutti, Lukas Esterle*	IR comments, Second Round
0.3	05/01/2023	*Dragana Bajovic*	Addressed IR comments
0.2	28/12/2022	*Alessio Brutti, Lukas Esterle*	IR Comments, First Round
0.2	22/12/2022	*Dragana Bajovic*	First Consolidated Version
0.11	11/11/2022	*Dragana Bajovic*	Final ToC
0.1	07/11/2022	*Dragana Bajovic*	Release of ToC to partners

Contents

List of Tables

List of Figures

List of Abbreviations

AAC	Automated Audio Captioning
AAVC	Automated Audio-Visual Captioning
AE	Autoencoder
AI	Artificial Intelligence
AIMR	AI Model Repository
AT	Audio Tagging
AUC	Area Under Curve
AVAD	Audio-Visual Anomaly Detection
AVCC	Audio-Visual Crowd Counting
CNN	Convolutional Neural Network
CRNN	Convolutional Recurrent Neural Network
DFB	Data Fusion Bus
DL	Deep Learning
DNN	Deep Neural Network
DP	Differential Privacy
DPO	Data Protection Officer
DX.X	Deliverable X.X
E2F2C	Edge-to-Fog-to-Cloud
EC	European Commission
ES	Elastic Search
ER	Error Rate
ERM	Empirical Risk Minimiser
EU	European Union
FFS	Feature Feature Selection
FFT	Fast Fourier transform
FL	Federated Learning
FLOPS	Floating Point Operations per Second
FN	False Negative
FP	False Positive
FS	Feature Selection
GA	Grant Agreement
GPU	Graphics Processing Unit
gRPC	Google Remote Procedure Calls
HDD	Hierarchical Data Distribution
HPC	High-Performance Computing
IID	Independent and Identically Distributed
KPIs	Key Performance Indicators
ML	Machine Learning
MSE	Mean Square Error
MQTT	MQ Telemetry Transport
MVP	Minimum Viable Product
NUS	Non-uniform Sampling
PANN	Pretrained Audio Neural Networks
R1	First release of MARVEL framework
R2	Second release of MARVEL framework
RNN	Recurrent Neural Network
SA	Self-Attention

SGD	Stochastic Gradient Descent
TAD	Text Anomaly Detection
TEE	Trusted Execution Environment
TP	True Positive
TN	True Negative
TX.X	Task X.X
VAD	Voice Activity Detection
VCC	Visual Crowd Counting
ViAD	Visual Anomaly Detection
VPN	Virtual Private Network
WP	Work Package

Executive summary

This document is the final report for the work carried out in the context of Task 3.2 of the MARVEL project: "MARVEL's personalized federated learning realization for extreme-scale analytics," describing the research, innovation, and development contributions carried out in the context of the task. Specifically, the deliverable describes the task's contributions along the following five axes.

First, the task has led to the development of a novel open-source software component for personalised federated learning dubbed FedL. The deliverable describes the methodological approach behind the developed component, including efficient sampling of heterogeneous devices. It also provides implementation and deployment details of the component, how the component is integrated into the MARVEL framework, as well as extensive performance evaluation results.

Second, the deliverable describes how the developed FedL component has been applied to three MARVEL's use cases: 1) GRN4 Traffic junction trajectory collection, in the Malta pilot; 2) MT1 Monitoring of crowded areas, in the Trento pilot; and 3) UNS1 Drone experiment, for monitoring large space public events in the experimental pilot in Novi Sad. It also explains how the FedL component can be applied to two highly relevant deep learning tasks, also utilised in MARVEL use cases: visual crowd counting (VCC), and audio-visual crowd counting (AVCC).

Third, the deliverable describes the privacy-preserving features of the employed federated learning methods, as well as additional privacy-preserving mechanisms, such as differential privacy, that have been employed in the design of FedL and more broadly for the work carried out in the task.

Fourth, the deliverable describes the Edge-to-Fog-to-Cloud (E2F2C) infrastructural patterns and systems that have been employed in the context of the development of FedL and more broadly, in the context of work on this task.

Fifth, the deliverable describes the task results in a broader context of personalised federated learning for extreme-scale analytics. Namely, the task produced several novel methodologies and results for: 1) personalised federated learning and clustering; 2) decentralised and unsupervised anomaly detection; 3) federated feature selection for efficient data compression and communication; 4) nonlinear mappings design for more robust federated learning; and 5) development of large deviations (rare events) metrics for performance evaluation and design of federated and distributed learning methods. These results have led to several publications in the respective field. In this context, the objective of this deliverable is to provide brief summaries of the results of these papers, the progress beyond state-of-the-art that they achieve, and their relevance to the MARVEL project.

1 Introduction

1.1 Purpose and scope of the document

The main purpose of this document is to report on research, innovation, and development results achieved in the context of Task 3.2 of the MARVEL project, explain the task's contributions achieved towards WP3 and overall project's objectives, and link the results achieved with the related MARVEL's key performance indicators (KPIs).

In more detail, the deliverable describes the following contributions achieved. First, the task has led to the development of a novel open-source software component for personalised federated learning dubbed FedL. This document provides details on the scientific methodology behind FedL, its implementation, deployment and MARVEL R1 integration details, and extensive performance evaluation. It also describes how FedL has been applied to specific deep learning models for VCC and AAVC, and the three MARVEL's use cases: 1) GRN in Malta, for traffic analysis and anomaly detection; 2) MT in Trento, for city monitoring and situational awareness; and 3) experimental pilot UNS in Novi Sad, for monitoring large space public events. Second, the deliverable describes the privacy enhancement methodologies and tools utilised in the context of FedL and more broadly the work carried out in the task. Third, it outlines the edge-fog-cloud architectural and deployment patterns utilised in the task. Next, the deliverable describes research results obtained in the context of the task that have contributed to the state-of-the-art in personalised federated learning. These research efforts have resulted in a number of scientific publications, whose results are briefly outlined in the deliverable, focusing on their contributions beyond state-of-the-art, as well as their relevance to MARVEL. These research results include 1) novel methodologies for personalised federated learning and model clustering; 2) decentralised and unsupervised anomaly detection; 3) federated feature selection for efficient data compression and communication; 4) nonlinear mappings design for more robust federated learning; and 5) rare event (large deviations) metrics for the analysis and design of federated and distributed learning methods. We refer the reader to the accompanying papers for full details on the research results achieved, also provided in the Appendix of this document. Finally, the deliverable describes how the FedL component will be utilised and exploited in the Release 2 (R2) period of the MARVEL project, e.g., in the context of the second integrated solution and further MARVEL use cases.

1.2 Intended readership

The deliverable is primarily intended for researchers, developers and practitioners in the domain of federated learning and related domains. Secondly, the deliverable is of interest to professionals with a non-technical background that work in the domain of smart cities and related domains, as the deliverable describes some examples on useful applications of deep learning in the domain. Thirdly, the deliverable may be of interest to a more broad audience generally interested in the subject. While the text necessarily includes some technical content, an intention has been made to reduce the degree of highly technical details, such as extensive mathematical definitions and proofs. An interested reader is referred to the accompanying scientific papers.

1.3 Contribution to WP3 and project objectives

The deliverable contributes to the following WP3 objectives:

- (i) *Define and execute measures ensuring privacy preservation during data processing*: The deliverable describes how differential privacy has been incorporated in the FedL component for improving privacy preservation;
- (ii) *Follow a personalised federated learning approach to train and execute ML models at the Edge, Fog and Cloud*: The task has developed a novel component for personalised federated learning dubbed FedL; the report also describes how the models have been trained via FedL and executed over various Edge-to-Fog-to-Cloud architectural and deployment patterns;
- (iii) *Train new or updated ML algorithms for audio-visual classification and analytics*: The deliverable describes how new models for several audio-visual tasks have been trained, including VCC and AVCC tasks.
- (iv) *Optimise ML deployment in the E2F2C infrastructure in a continuous manner*: Even though this WP3 objective is not central to the current task, the deployment of models for each use case has been optimised with respect to the computational and storage capabilities of edge, fog and cloud elements for each specific use case.
- (v) *Deploy ML algorithms at all layers of E2F2C*: The deliverable describes how FedL has been deployed at edge, fog, and cloud layers for each of the three use cases considered in the deliverable.
- (vi) *Implement light-weight ML models for deployment at the edge*: The task has developed federated unsupervised anomaly detection methods wherein lightweight models such as lightweight autoencoders have been deployed at the edge.

Regarding contributions to the overall MARVEL's objectives, we refer the reader to the analysis below of the specific MARVEL project's KPIs that are relevant to the current task.

1.4 Positioning of FedL in MARVEL's architecture

Figure 1 shows the MARVEL's conceptual architecture, grouping the components in seven subsystems, and described in detail in D1.3[1], with subsequent revisions reported in D6.1[2]. MARVEL's federated learning component FedL is part of the Optimised E2F2C Processing and Deployment Subsystem, shown in violet in the figure (other subsystems are shaded in the figure), whose functional role is to enable optimised realisation and deployment of various MARVEL services and functionalities. This subsystem consists of two types of components: 1) components that aim at optimising the platform's operation, in terms of model accuracy through federated training – FedL, model size through model compression – DynHP, and GPU-based pattern matching acceleration – GPURegex; and 2) Kubernetes-based deployment through MARVdash platform,

[1]MARVEL D1.3: Architecture definition for MARVEL framework, 2021. https://doi.org/10.5281/zenodo.5463897

[2]MARVEL D6.1: Demonstrators execution – initial version, 2022. https://doi.org/10.5281/zenodo.6862995

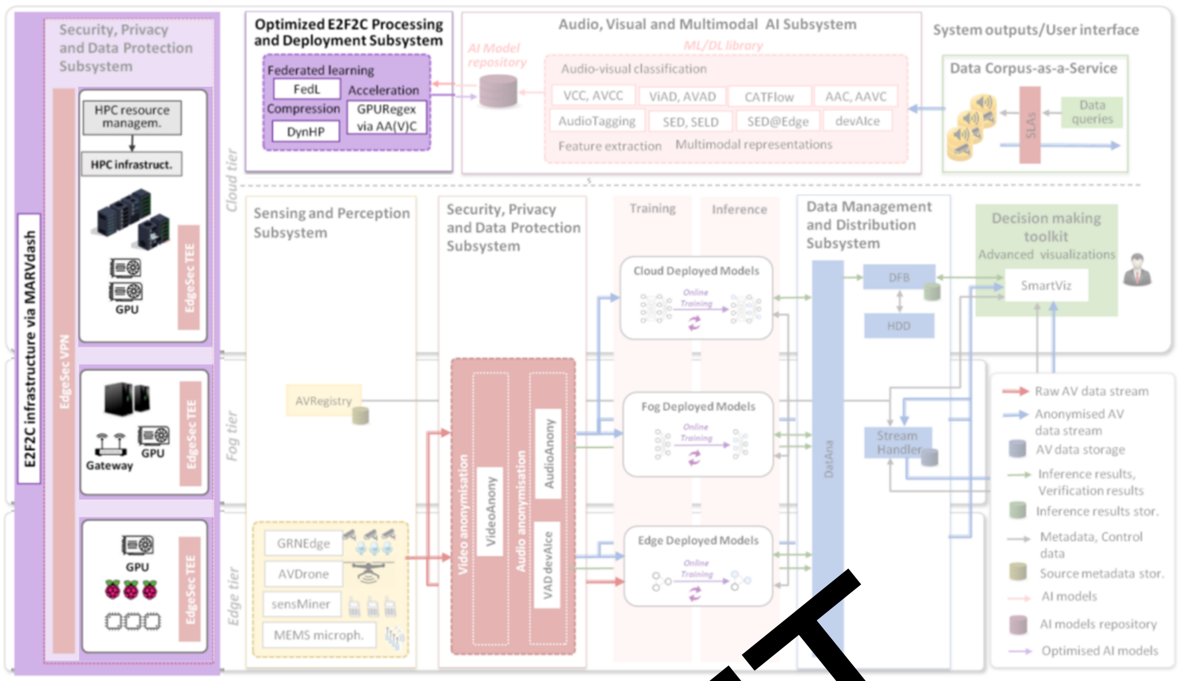


Figure 1: MARVEL's conceptual architecture

which facilitates deployment of MARVEL services. The main interface of FedL within MARVEL's architecture is with the AI model repository (AIMR). This is a database of machine learning and deep learning models produced or used within MARVEL, and is part of the Audio, Visual and Multimodal AI subsystem; examples of such models include visual and audio-visual crowd counting (VCC, AVCC), visual and audio-visual anomaly detection (ViAD, AVAD), and other. The interface of FedL with AIMR is described in detail in Section 2.3.1, while Section 2.3.2 describes the data handling pipeline of the models trained by FedL in their inference phase.

1.5 Relation to other work packages

Task 3.2 is part of WP3 which is responsible for the development of MARVEL's solution for AI-based distributed algorithms for multimodal perception and situational awareness. As such, it carries out algorithm design based on the requirements and state-of-the-art analysis set out in WP1. It performs model training based on the data that has been collected and prepared in WP2. It also makes use of the E2F2C framework patterns and data anonymisation techniques developed in WP3 (anonymisation) and WP4 (voice activity detection). The FedL component that Task 3.2 develops is integrated in the MARVEL framework in the context of WP5. FedL and other Task 3.2 contributions are then applied in real-world experiments within WP6. Exploitation of the task's results is being carried out in the context of WP7.

1.6 Connection to Key Performance Indicators

1.6.1 Project-related KPIs

KPI-O1-E3-1: *Number of incorporated safety mechanisms (e.g. for privacy, voice anonymisation)* ≥ 3. Status: **Achieved.**

This KPI is also addressed under T3.1, T4.2 and T4.3, as detailed in D3.3 and D4.4, where components for video anonymisation – VideoAnony, audio anonymisation – AudioAnony, and Voice Activity Detection – VAD, are described together with a number of safety mechanisms, including also end-to-end security and data and code integrity verification components – EdgeSec VPN and EdgeSec TEE.

To provide an additional privacy preservation mechanism, specifically for the federated training process, the MARVEL's federated learning component FedL incorporates a differential privacy mechanism within each FedL client's training. This is achieved by adding a controllable amount of noise to the client's local model updates, thus masking the original data and reducing the likelihood of gradient inversion. Details of the differential privacy implementation within FedL are provided in Section 2.2.2.

KPI-O2-E1-1: *Standard non-personalised federated learning improvement (performance and speed) at least 10%* Status: **Achieved**.

Regarding training speed, an adaptive FL protocol called FedL NUS, described in Section 2, was proposed and developed that accounts for communication and node availability statistics thus effectively increasing training speed compared to a standard protocol. For clients that exhibit heterogeneities, the protocol achieves significant improvements both on standard FL benchmarks as well on MARVEL data in terms of the training speed over the baseline FedAvg [1]. On the LEAF benchmark (widely adopted and most commonly used benchmark for FL) [2], specifically the MNIST dataset [3], FedL with NUS strategy reaches an accuracy of 88%, which is only $2.2\%(=88/90)$ lower relative to the accuracy of 90% of the baseline algorithm FedAvg, while at the same time achieving significant savings in the number of communicated data. Specifically, in the same experiments, to reach the indicated accuracy, FedAvg requires the transmission of 120 megabytes (MB) in total, while FedL NUS requires in total less than 80 MB of transmitted model data, savings of more than 33%. See Section 3.3 for details on performance results and experiments carried out.

Regarding performance improvement, a novel methodology for client model personalisation based on convex clustering was developed, which accounts for personalised data mixtures, tuning the models to the local data distribution to achieve improved model accuracy.

iKPI-1.1: *At least three (3) tools for complex/federated/distributed systems handling extremely large volumes and streams of data* Status: **Achieved**.

As a result of the work carried out in T3.2, UNS has delivered FedL component implementing the non-uniform client sampling (NUS) strategy that is particularly well-suited for federated training with a large number of participating clients with arbitrary data modalities, including multimodal data streams; see Section 2 and 3 for detailed descriptions of FedL. Second, within T3.2 a novel methodology for client clustering within FL with a matching algorithm is developed that finds hidden cluster structures in an arbitrarily large pool of clients. The relevance of this methodology is to facilitate model search for a newly arriving client by checking only a small number of model candidates, which are provided by the tool, and hence significantly reducing the complexity of the overall process when a very large number of clients/datasets are participating in the training; see Sections 4 and 5 for details. Third, a tool for Dynamic split computing for distributing DL architectures to two consecutive tiers of the E2F2C architecture is developed within T3.3 (AU, UNS). The tool can operate on DL models of large sizes (large number of layers) and is suitable for data of large volumes (such as video); this work is reported in D3.1[3]. Further, within T3.2 a tool is proposed for optimising data exchange protocols in fully distributed systems with complexity exhibited both in topology (e.g., generic topology) and inherent system randomness (link failures, node failures). For such systems, we develop a tool for adaptively optimising communication frequencies of nodes' interactions/data transmission and observe that important communication savings can be achieved with respect to a time-constant strategy, with negligible performance deterioration; the details can be found in Section 10. Finally, a contribution to this KPI is the novel analytical framework based on the theory of large deviations for stochastic gradient descent–based learning algorithms, including federated learning, and the accompanying design metrics that show advantages over the standard mean-square error metrics, being able to analytically capture the geometrical interplay between the noise distribution and loss functions, higher-order noise moments, noise skew and other parameters that impact learning performance; details are provided in Section 9. Additional contributing tools/methods to this KPI include: the tool for unsupervised anomaly detection tasks over distributed and federated edge devices, Section 6, the federated feature selection tool, Section 7, and the method for handling heavy-tailed gradient noise often exhibited with DL models training, applicable to centralised or federated settings, Section 8.

iKPI-3.1: *Accuracy ratio ($>$ 95%) of training of deep learning models with respect to full dataset training* status: **Achieved**.

On the LEAF benchmark (widely adopted and most commonly used benchmark for FL), specifically MNIST dataset, FedL with NUS strategy reaches an accuracy of 88%, which is only $2.3\% (= 88/90)$ lower relative to the accuracy of 90% of the baseline algorithm FedAvg, while at the same time achieving significant savings in the number of communicated data. Specifically, in the same experiments, to reach the indicated accuracy, FedAvg requires transmission of 120 MB in total, while FedL NUS requires in total less than 80 MB of transmitted model data, achieving savings of more than 33%. We consider here the full dataset training to be the method that adopts a bucket sampling across the users' data, that is, we consider stochastic gradient descent (SGD) where a mini-batch is formed by sampling data from all users. Compared with this benchmark, we approach its accuracy to within 95%, 90% versus 88% achieved by NUS. Similar improvements are reported for experiments over MARVEL data, as detailed in Section 3. We evaluate the performance on MARVEL data for visual crowd counting (VCC) training task using the SASNet VCC model. The obtained savings due to the employed protocol are significant: for 100 training epochs, FedAvg requires transferring $> 50\%$ more (model) data – 46 gigabytes (GB) vs 28 GB. The FedL-NUS training process results in higher accuracy for 2 clients that exhibit similarity in their data (MT and UNS), showing clear benefits of federated training (see Table 4 for details).

[3] MARVEL D3.1: Multimodal and privacy-aware audio-visual intelligence – initial version, 2021. https://doi.org/10.5281/zenodo.6821318

iKPI-12.1: *Reduced training time by at least 10% compared to standard approach.* Status: **Achieved**.

An adaptive FL protocol was proposed and developed within T3.2 that accounts for communication and node availability statistics thus effectively reducing training time compared to a standard protocol. For clients that exhibit heterogeneities, the protocol achieves an improvement of $\approx 30\%$ on both the standard benchmark dataset (LEAF) and of $\approx 50\%$ on MARVEL data in terms of the training speed over the baseline FedAvg, with higher accuracy for the majority of clients. The details are provided in Section 3.3. Additional contribution to this KPI is the analysis and unification of various nonlinear gradient mappings designed for handling heavy-tailed gradient noise and thus with potential to increase the speed and communication efficiency of ML/DL training, as elaborated in Section 8.

1.7 Structure of the report

The remainder of the deliverable is structured as follows. Overall, the report consists of two main parts. The first part contains Sections 2 and 3 and is devoted to the FedL component. Section 2 describes the component, while Section 3 explains how it is applied to the targeted MARVEL's use cases, within MARVEL R1. The second part of the deliverable (Sections 4–10) describes the research results carried out in the context of the task that led to four scientific publications and three preprints (undergoing peer review). This part of the deliverable describes these results on a section-per-paper basis. For each section (and an associated paper), main approach, progress beyond state-of-the-art, evaluation results, and relevance to MARVEL, have been described. Specifically, Sections 4 and 5 deal with clustered personalised federated learning, summarising the work reported in:

- A. Armacki, D. Bajovic, D. Jakovetic and S. Kar, “Personalized Federated Learning via Convex Clustering,” 2022 IEEE International Smart Cities Conference (ISC2), 2022, pp. 1-7, DOI: 10.1109/ISC255366.2022.9921863, Appendix A[4] and
- A. Armacki, D. Bajovic, D. Jakovetic and S. Kar, “One-Shot Federated Learning for Model Clustering and Learning in Heterogeneous Environments,” 2022, Zenodo, `https://doi.org/10.5281/zenodo.7537614`, Appendix B

Section 6 is on decentralised and unsupervised anomaly detection, reported in:

- M. Nardi, L. Valerio, and A. Passarella, “Anomaly Detection through Unsupervised Federated Learning,” Proc. of the 18th IEEE Int. Conference on Mobility, Sensing and Networking (MSN), 2022, Appendix C

Section 7 considers federated feature selection, reported in:

- P. Cassará, A. Gotta and L. Valerio, “Federated Feature Selection for Cyber-Physical Systems of Systems,” in IEEE Transactions on Vehicular Technology, vol. 71, no. 9, pp. 9937-9950, Sept. 2022, DOI: 10.1109/TVT.2022.3178612., Appendix D[5]

Section 8 considers robust learning by employing a generic nonlinearity in the learning process, reported in:

[4] `https://doi.org/10.5281/zenodo.7007248`
[5] `https://doi.org/10.5281/zenodo.6901227`

- D. Jakovetic, D. Bajovic, A. Sahu, S. Kar, N. Milosevic, D. Stamenkovic, “Nonlinear Gradient Mappings And Stochastic Optimization: A General Framework With Applications To Heavy-Tail Noise,” accepted for publication in SIAM Journal on Optimization, 2022, Zenodo, https://doi.org/10.5281/zenodo.7538446 Appendix E

Sections 9 and 10 deal with the large deviations analysis and design of distributed inference and learning methods, reported in:

- D. Bajovic, D. Jakovetic, and S. Kar, “Large deviations rates for stochastic gradient descent with strongly convex functions,” 2022, Zenodo, https://doi.org/10.5281/zenodo.7537625, Appendix F
- D. Bajovic, “Inaccuracy rates for distributed inference over random networks with applications to social learning,” 2022, Zenodo, https://doi.org/10.5281/zenodo.7537631, Appendix G

Finally, some conclusions and future work directions are outlined in Section 11.

PART I: MARVEL's federated learning realisation – component FedL

2 MARVEL's federated learning realisation: FedL

2.1 Motivation

We first discuss the motivation for privacy preservation in each of the MARVEL pilots, GRN, MT and UNS, and how federated learning can help in that aspect.

GRN pilot. GRN plans to provide stakeholders operating in the transport arena, such as transport authorities, transport consultants, and researchers in transport with new services that would assist the end-users in short-term and long-term studies and decision-making. For this purpose, GRN is collecting road traffic data from various visual and audio devices installed around the island of Malta. Inevitably the sensors may collect personal data (PD) related to the general public that needs to be removed or transformed to comply with the GDPR legislation. GRN has identified three classes of PD that could be present in video and audio data; (a) human faces and vehicle number plates are features that would enable individuals to be identified in video data, (b) human beings can be identified from the recording of their voice data, and (c) PD may be present in the content of the speech signals. The removal or transformation of these types of data would be in line with the DPO's requests to preserve the privacy of any citizen that is recorded by the MARVEL framework.

MT pilot. The final goal of the Municipality of Trento is to effectively use the vast amount of multimodal data daily provided by the audio-visual sensors distributed in the city for improving the quality of life of citizens, the quality of the city services and for supporting the administrator better understand and develop the city. Unfortunately, this scenario raises severe privacy issues as unaware citizens are recorded by the audio-visual sensors, which, in turn, poses limitations on what can be done with the recorded data. In particular, the MT Data Protection Officer (DPO) allowed only anonymised data to be shared across the consortium and introduced restrictions on the access by FBK, as the technical supporting partner of MT in MARVEL, to the raw data. Besides removing citizens' identities from the audio-visual stream, an additional option is to move the algorithms on processing nodes close to the data sources (i.e., edge) or within the authority network (i.e., fog).

UNS pilot. In general, given the nature of the UNS use cases which are all based on staged recordings, where participants gave their consent to audio-visual recordings via cameras and microphones, privacy regulations do not directly apply, and no specific issues are present. However, the use case implementations account for restrictions that would apply in any future real-life deployment and, therefore, audio and video anonymisation is applied on edge devices. With regards to model training, there is also a clear need to preserve data privacy and keep the data local (i.e., within the end user's premises) during the training process.

Federated Learning is one of the most popular techniques for privacy preservation in Machine Learning (ML) and Deep Learning (DL) solutions. ML and especially DL models are very data dependent. In both supervised and unsupervised learning scenarios high quality, data is key to having a performant model. The amount of data is also extremely important and it is always useful for the model performance to use all the available data at a given time. This characteristic of ML and DL models also comes with the question of privacy. In a traditional learning process, the data needs to be given to the algorithm which performs the model training. As the data needs to be shared, it can be very difficult to guarantee privacy. In our concrete case, we have several data collection points divided into three positional groups: MT, GRN, and UNS. To have high-performance models, ideally, we would want to use all the data available from these collection points. But the data cannot be shared due to privacy concerns, regulations and so on. While the MARVEL project implements techniques such as anonymisation, it is still paramount to have other solutions for the preservation of privacy in data, such as Federated Learning (FL). In an FL scenario, we develop a distributed system where data never leaves its source, i.e., allowed data processing entity, but rather we use the data for learning where it is collected. We are still able to share what has been learned by local FL clients by sharing the learned model parameters. In the end, we obtain a global model which is very close in performance to the theoretical non-existent model trained on all the data. The key difference being that the data never left its source, and therefore privacy risks are minimal.

2.2 FedL Component Description

In this section we will describe the MARVEL federated learning component FedL, focusing on its conceptual and implementation details. Deployment and integration in MARVEL R1 use cases is described in Section 3. We will also discuss our newly developed strategy: non-uniform sampling (NUS) strategy as an alternative to the widely used FedAvg strategy [1] and how privacy preservation can be further reinforced by the usage of techniques such as differential privacy.

FedL is a pure software component developed in Python programming language. At its base, it adopts commonly used Python numerical libraries such as NumPy [4] and pandas [5], in addition to various DL frameworks such as Pytorch [6] and Tensorflow [7].

FedL implements Federated Learning through usage of the Flower (flwr) library [8], which is a highly adaptable and extensible FL library for Python. Flower was mainly chosen as the basis for FedL implementation because it allows for custom FL strategies, which is a very important feature for our future research. Flower also supports extensions to the FL process through various interfaces, another important feature to us. Finally, it is framework agnostic which means that any DL and ML framework can be used with it as long as model exporting and importing of parameters is supported. So far we have used FedL with both TensorFlow models (custom AVCC model, AU) and PyTorch models (VCC model SASNet, open-source, adapted by AU). Our contributions and additions to the open-source library such as custom strategy and client code will be explained in the following sections.

In an FL scenario, we differentiate two types of software agents: the FL clients and the FL server. The relation is usually one-to-many, meaning that for many clients we have one server. The clients' role is to use local data to train a specific ML model and share its knowledge (through parameter sharing with the server) with other clients. The server's role is to collect the parameters sent by clients and perform some "averaging" operation in order to create a global model which is then shared with all the clients to continue local training. The advantage of such a system is that FL clients still benefit from the data of other clients without explicitly requiring access to it.

```
class AVCCClient(fl.client.NumPyClient):
  def __init__(self, *args, **kwargs):
    super().__init__(*args, **kwargs)
    self.model, self.train_sequence, self.test_sequence,
            self.val_sequence = train_backbone(0)

  def get_parameters(self):
    return self.model.get_weights()

  def fit(self, parameters, config):
    self.model.set_weights(parameters)
    self.model = train_backbone(1)
    return self.model.get_weights(), len(self.train_sequence), {}

  def evaluate(self, parameters, config):
    self.model.set_weights(parameters)
    loss, accuracy = self.model.evaluate(self.test_sequence)
    return loss, len(self.test_sequence), {"accuracy": accuracy}
```

Figure 2: Minimal code example of a FedL TensorFlow (v2) client. Only three methods need to be implemented: for obtaining the parameters, for training and for evaluation. Training parameters are shared as standard NumPy arrays.

For communication between clients and the server, the gRPC protocol[6] is used which means that stable networking has to be set up between clients and the server. In the case of the MARVEL project this was done through software-defined networking (SDN) within the main Kubernetes cluster which was set up for the project.

In real-world systems, various clients can become unavailable due to networking, or other issues. Our research and our newly developed strategy, NUS, are developed in order to address these often encountered communication constraints, by developing a client selection process where not all clients need to be available for training and only clients with globally beneficial model changes are chosen to perform the learning. This not only allows for some clients to be offline during learning but also saves significant amounts of bandwidth in model parameter transfers. Our strategy will be explained in greater detail in the following sections.

For third-party users adopting our component, the machine learning training code has to be modified with FL in mind. This means a custom client Python class has to be written which defines how model parameters are shared and how the received global model parameters are to be used for local clients.

On the server side, a configuration must be provided with hyperparameters such as number of rounds to perform, the fraction of clients used for training and evaluation, and so on. Another important setting for the server is strategy implementation. We use the custom, NUS strategy, while FedAvg [1] is generally used as a default strategy. FedAvg strategy performs a simple parameter averaging process by polling all FL clients and averaging over all the received client updates, providing an average global model. In our tests, this strategy performed well, despite its simplicity but for flaky communication and to save data transfers we encourage users to use our NUS strategy. We

[6]Google Remote Procedure Calls https://github.com/grpc/grpc/releases

will present full results and implementation details in the Results section 3.3 of this document.

2.2.1 Non-uniform sampling (NUS) strategy: theory and implementation

Communication issues are ubiquitous in our everyday life. When designing a system that has a high impact on human quality of life, these issues have to be accounted for. Flaky communication can be very difficult to solve in FL scenarios. When updates are not sent from the client to the server usually that client becomes out of sync with the global model and its learning process stagnates (client starvation). Another side effect is that other clients do not benefit from the unresponsive clients' data/updates. Finally, if a high fraction of clients must be present for the learning process, it can become completely stagnant as responsive clients wait for unresponsive clients.

In a complex system with many components which communicate via the same means (e.g., same networking interfaces), it is also paramount to minimise communication without suffering a performance loss. This is especially important for systems that span multiple layers of the architecture such as our own MARVEL system which operates on the Cloud, Fog, and Edge layers. The edge layers are very interesting and important in any FL setting as the edge devices usually collect the data and sometimes are powerful enough to immediately perform training with the collected data. As networking in any large system (ours included) can be very heterogeneous, edge devices may not have a stable communication channel with other components, and minimising data transfers would be very helpful, especially taking into account that edge devices are often wirelessly connected and hence energy savings by less frequent transfers are of high relevance.

For all these reasons, we implemented the NUS federated learning strategy to be tested and used in the FedL component of the MARVEL architecture.

We will now describe our strategy at a high level of abstraction. NUS operates not only on client parameters but also on client meta-parameters which are used to judge clients and estimate whether their updates are significant and important for the model. These meta-parameters describe what happened in one FL step (round) on the client itself. Meta-parameters include the number of processed data points, stability of the network connection to the server (with polling and response times), gradient variance, and average parameter difference to the global model. Our strategy requires clients to send this information (or a subset of) to the server upon which the server runs a client selection process to choose the subset of clients that will be polled to perform an update. Only the clients with important and significant updates to the global model are selected to share their parameters, while others are skipped. This saves the amount of data being transferred without affecting the performance of the global model.

From the implementation standpoint, NUS requires both the client and the server code to be modified.

On the client side, the modifications are minor. Every round of training a client receives specific fitting instructions. These instructions specify whether the client needs to send an update in the next iteration of the learning process. The server controls which clients are required for the next step. The client, depending on these fitting instructions either sends the model parameters or censors their message. Clients also respond to the server polling requests to signal their availability. In every training step, the clients also need to respond with meta-data about their current training stage. We report the following parameters:

1. *batch_size* – current batch size used by the client
2. *num_examples* – number of samples processed in a specific training step
3. *fit_duration* – amount of time taken for this current step
4. *gradient_norm* – current model gradient norm
5. *model_diff* – current model difference to the global model (measured with a specific metric, e.g., mean squared error).

Based on these parameters the server decides whether an update needs to be retrieved from a specific client. In more detail, for each client, the server solves an optimisation problem that optimises the client selection probabilities for the next round for all the clients. The optimisation problem takes into account each client's responsiveness, local data variance, and the norm of the distance between the client's current gradient and the global gradient estimate from the previous round. It is important to specify also that validation is being performed at every round of the FL training process. Global models are used and validation is done on a per-client basis. In this way we can monitor how the global model is performing with specific client's data.

On the server side, NUS is implemented as a custom strategy for the *Flower* framework[7]. All the clients receive the same initial model, instantiated by the server, to avoid issues with random weight initialisation in case of training a blank model. If a pre-trained model is used (e.g., from the MARVEL AI model repository, detailed in Section 2.3.1 further ahead) it can also be shared with the clients. In every training round after receiving information (as described above) from the clients, the server decides the configuration to be used for the continuation of the training process. When client model parameters are gathered, the NUS strategy performs parameter averaging in the same way as the FedAvg strategy, while the main difference is the client selection process, as described above.

2.2.2 Differential privacy implementation

By using FL in our system, we already significantly increased the privacy and safety of the employed data. The data being collected is never shared with other clients, or in other words, never has to leave its origin. Another issue to be discussed is about what exactly is being shared between clients. As we mentioned already, the FL process shares only model parameters between clients and the server. The server then performs an averaging (or similar) operation on the client updates and creates a global model. The issue with this approach in some systems is that the model parameters may contain useful (or private) information collected from the data. In the case of MARVEL, the data is encrypted before the sharing process, but the privacy aspects can be improved even further with the usage of differential privacy.

Differential Privacy (DP) is a method introduced in 2006 in [9] to enable publicly sharing information about a given dataset while disabling (or minimising) the chance

[7] https://flower.dev/

```
model = SASNet()
optimizer = torch.optim.SGD(model.parameters(), lr=0.05)

privacy_engine = PrivacyEngine(
    model,
    batch_size=32,
    sample_size=len(train_loader.dataset),
    alphas=range(2,32),
    noise_multiplier=1.3,
    max_grad_norm=1.0,
)

privacy_engine.attach(optimizer
```

Figure 3: Code snippet of DP-SGD "wrapper" around PyTorch's SGD, configuration that enables differential privacy in FedL.

to discover information about any individual datum from the dataset. A perfect differentially private method would always provide the same results no matter which sample is excluded, but this is very difficult to achieve in practice. As a realistic alternative, a privacy budget (ϵ) is introduced to achieve the desired privacy level, but at the cost of incurring certain privacy which depends on the way the privacy budget is used. The most common method to use DP is to add independent Gaussian noise to the raw input data. Another option (that we will demonstrate) is to use DP-aware optimisation algorithms such as DP-SGD [10]. This approach is more suitable for FL scenarios as it does not apply noise to the input data but rather to the gradients in every step in the gradient descent.

To use differential privacy methods in FedL, we use the state-of-the-art open-source Opacus [11] library and its DP-SGD implementation.

2.3 FedL interfaces and the inference pipeline

This section explains the interfaces of FedL within the MARVEL architecture, as well as the management of results of the FedL models in their inference phase.

2.3.1 FedL and MARVEL AI repository

The AI Model Repository (AIMR) is an internal component of the MARVEL architecture devoted to the storage of AI models designed and trained during the project. In principle, any AI component will access this repository to upload/retrieve an AI model. As an example, each instance of a FedL client can access a specific model (e.g., VCC) and train it. The new update model is then uploaded to the AIMR to be shared with the other components. Another example regards the compression pipeline, involving DynHP. In this case, DynHP accesses the AIMR to (i) retrieve the original definition of a model, (ii) find a suitable compression, and (iii) upload the compressed model to the AIMR.

The AIMR service is offered and managed by MinIO[8]. The files are organised hierarchically and per pilot. Specifically, it offers dedicated folders, one for each Use Case covered by MARVEL. Each of these folders contains all the versions of the corresponding AI Models. A human-readable JSON description (called Model Descriptor) accompanies all the models stored on the AIMR. It is based on the FIWARE standard. The description contains all the necessary information describing the specific AI Model and the pointers to the file containing the Models' parameters and/or binary file, according to how these models are implemented and deployed. This metadata could be of use for humans to have an idea of which model to select for building a docker image for a specific purpose. Both AI models and Model Descriptors follow a specific naming convention. The naming convention is hierarchical, and it includes the following fields:

- `<component_name>`
- `<output_format>`
- `<partner>`
- `<type>`
- `<version_number>`

Example: `avcc-headcount-au-full-v0.1`.

Further and complementary details on the AI model repository can be found in D5.4[9]. We discuss below the current collection of AI models offered by the AI model repository.

TAU provided four models for the AI model repository that were specifically trained to be used in several R1 use cases. Models included in the repository are provided in serialised and optimised PyTorch JIT format [10] to allow models to be transferable across various settings and platforms. These models are standalone, and they contain all processing blocks needed for inference: audio feature extraction (FFT layers) and the actual AI model. Input to these models are single-channel audio segments of appropriate length, and the output is a vector containing class-wise output values. For GRN3, the Audio Tagging (AT) model is capable of labeling given 5-second audio segments into predefined labels describing the traffic speed and amount. The model architecture is based on the structure used for PANNs audio embeddings [12]: a 10-layer CNN with 4 convolutional layers based on VGG-like CNNs [13]. For MT3, a similar model architecture is utilised to label 1-second audio segments into predefined labels describing the overall scene (e.g., fighting in the parking lot or stealing a car). For GRN4, the sound event detection (SED) model is capable of detecting the activity of four vehicle types (e.g., car and bus) in given 1-second audio segments. The model architecture for GRN4 is based on a 6-layer CNN with four convolutional layers based on AlexNet [14] using also in for pre-trained audio neural networks (PANN) audio embeddings. For MT3, the SED model is capable of detecting the activity of ten sound event classes related to human actions in parking places in given 1-second audio segments, and the model architecture is similar to the one used for GRN3 and MT3 for AT.

[8] https://min.io/

[9] MARVEL D5.4: MARVEL Integrated framework – initial version, 2022. Confidential.

[10] https://pytorch.org/docs/stable/jit.html

AU provides models for several different visual or audio-visual tasks, such as crowd counting and anomaly detection. For visual crowd counting (VCC), a SASNet model [15] is provided, which takes a high-resolution image (preferably 1024×768) as input and outputs a density map detailing the density of the crowd at each location of the input image. The values of this density map can be summed in order to obtain the total number of people present in the scene. SASNet uses the first 10 layers of VGG-16 [13], pre-trained on ImageNet, to extract features, and then adds several convolution layers at different scales in order to obtain density and confidence maps at each scale, which are then merged to obtain a single combined density map. For audio-visual crowd counting (AVCC), an AudioCSRNet model [16] is provided, which takes as input a high-resolution image (preferably 1024×576) as well as a 1-second audio waveform which corresponds to the ambient audio of the scene. The waveform should start 0.5 seconds before the image was taken and end 0.5 seconds afterwards. Similar to VCC, the output of AVCC is a density map. AudioCSRNet uses the first 10 layers of VGG-16 [13], pre-trained on ImageNet, to extract visual features, and uses the first 6 layers of VGGish [17], pre-trained on AudioSet, to extract audio features. These visual and audio features are then fused and processed together using fusion blocks which contain dilated convolutions.

The FedL component interfaces with the AIMR in both obtaining and saving models to it. At the start of an FL process, a model can be obtained from the AI model repository to be used as a starting point for further training. This can lead to very significant improvements in convergence times. FedL also uses the AIMR to store trained models after the Federated Learning process is complete. FedL provided models that are significantly different from other models in the AIMR as they represent global models, trained on all available data for a specific use case type. Other models are trained for specific use cases. Global models can be further improved with model compression and other techniques supported by the MARVEL components, such as DynHP.

For communication, FedL uses directly the MinIO interface of the AIMR for storage and retrieval of model parameters and metadata; further details of AIMR operation and interfaces with FedL as well as with other MARVEL components are reported in D5.4. The model format is dependent on the use case, but generally compatible with other components of the MARVEL framework. For this purpose, we use the general MLModel schema[11] from the Smart Data Models initiative[12].

2.3.2 Inference results management

We describe how the management of data produced by FedL models in their inference phase is achieved in MARVEL. For this functionality, data management platforms DatAna and DFB are used.

DatAna and E2F2C management of inference results. DatAna is one of the MARVEL Data Management Platforms. DatAna's main functionality is providing the means to get the results from all inference models to the cloud and to the Data Fusion Bus (DFB) for further aggregation and storage. This entails the connection with the different inference models using a message broker (based on MQTT[13]) and the provision of dedicated

[11] https://github.com/smart-data-models/dataModel.MachineLearning/blob/master/MLModel/README.md
[12] https://smartdatamodels.org/
[13] https://mqtt.org/

data pipelines for each of the inference components. These data pipelines are able to transform the inference results into specific MediaEvent, Alert or Anomaly data models, based on and extended from their Smart Data Models initiative counterparts adopted in MARVEL. These data pipelines reside either at the edge, fog or cloud layers. DatAna allows then the transfer of the transformed results from the specific layer where the computation takes place to the cloud, and eventually to the DFB via dedicated Kafka topics.

For the R1, DatAna has implemented data pipelines using Apache NiFi (deployed in the MARVEL framework via MARVdash) data flows for several use cases and inference components. In particular, DatAna implements so far NiFi data flows for CATFlow (vehicles and pedestrians), TAD, ViAD, AVAD, VCC, AVCC, AT and VAD inference models, shown in Figure 1, but it is extensible to develop new pipelines for other models in any of the existing layers and/or use cases.

During the execution of the models, the data managed within DatAna can be monitored using the NiFi graphical user interface, giving the administrator the possibility of even start and stop processors and to debug the process and visualise data in each step of the pipeline. More details on DatAna can be found in D2.2[14].

Data Fusion Bus and cloud management and storage of inference results. The Data Fusion Bus (DFB) is responsible for aggregating AI inference results received from DatAna through a Kafka messaging system, fusing and persistently storing them in an Elastic Search (ES) repository. It also re-distributes these results to SmartViz and Data Corpus both in real time through Kafka and by providing access to the historical data stored in the ES repository.

Due to the nature and central role of the DFB in the operation of the MARVEL AI Inference pipeline, it is positioned at the cloud layer. The internal operation of the DFB can be described as follows:

- The DFB receives AI inference results in real time from the DatAna component instance that is deployed at the cloud using the DFB Kafka messaging system. Distinct Kafka topics are configured for publishing the results of each AI component that has produced them. DatAna publishes AI inference results to these topics after transforming them according to the MARVEL data models that comply with the specification of Smart Data Models standard.

- The DFB relays the incoming AI inference result data streams in real time to SmartViz and Data Corpus by allowing them to subscribe to the associated Kafka topics where they are published.

- The DFB also includes an ES connector module that transfers all incoming AI inference results to the DFB Persistent Storage Repository, which is based on Elastic Search technology. The module also performs data fusion operations on AI inference results originating from selected AI components.

- The DFB exposes a REST API to SmartViz to allow it to perform data queries and access all archived inference results in its ES repository.

[14]MARVEL D2.2: Management and distribution Toolkit – initial version, 2022. https://doi.org/10.5281/zenodo.6821195

- The DFB can receive user-generated verifications of AI inference results from SmartViz when they are published as messages to a DFB Kafka topic that is reserved for this purpose. The DFB ES connector module then uses these verifications to update the respective archived AI inference result entries in the Persistent Storage repository. The verifications are also relayed to the Data Corpus, which is subscribed to the same Kafka topic where the verifications are published.

- Finally, the DFB also communicates with the MARVEL HDD component by accessing a REST API exposed by the HDD. The DFB uses this REST API to dispatch the currently applied Kafka topic partitioning information and to receive recommendations from HDD for updating the Kafka topic partitions to optimise their performance.

During the operation of the DFB, the administrator can monitor the status of the DFB Kafka messaging system through a dedicated tool, built with Grafana. The inbound and outbound traffic in each Kafka topic is visualised in real time as a graph. Other key statistics can also be visualised at the same time on the monitoring dashboard.

The DFB administrator can also access the archived data that are persistently stored in the ES repository using the associated Kibana tool. AI inference results from specific AI components can be isolated based on the implemented ES indices. The user can also filter the data using time ranges and perform composite queries based on the available fields of the stored AI inference results. In addition, Kibana offers various options for visualising data statistics. More details on DFB can be found in D2.2.

3 FedL Implementation in R1

Complementary to the description provided in Section 2, we now focus on the implementation details. This section describes FedL realisation and integration in MARVEL R1, related to the actual use cases where FedL has been deployed, including details of the infrastructure, models, data, and finally performance results. We first describe in detail the two models, VCC and AVCC, that have been used within FedL in R1.

3.1 AVCC and VCC models

The visual crowd counting (VCC) model is responsible for counting the total number of people that are present in a given scene. The input to this model is a high-resolution RGB colour image of a scene containing people, and the output that this model provides is a density map, which is a single-channel image that specifies the density of the crowd at each pixel of the input image. The values in this density map can be summed to get the total count in the form of a single number. The annotations used for training this model are in the form of head annotations, where the location of the center of each person's head is specified. This model utilises the SASNet architecture, shown in Figure 4, which is a high-performing DNN on ShanghaiTech datasets[15].

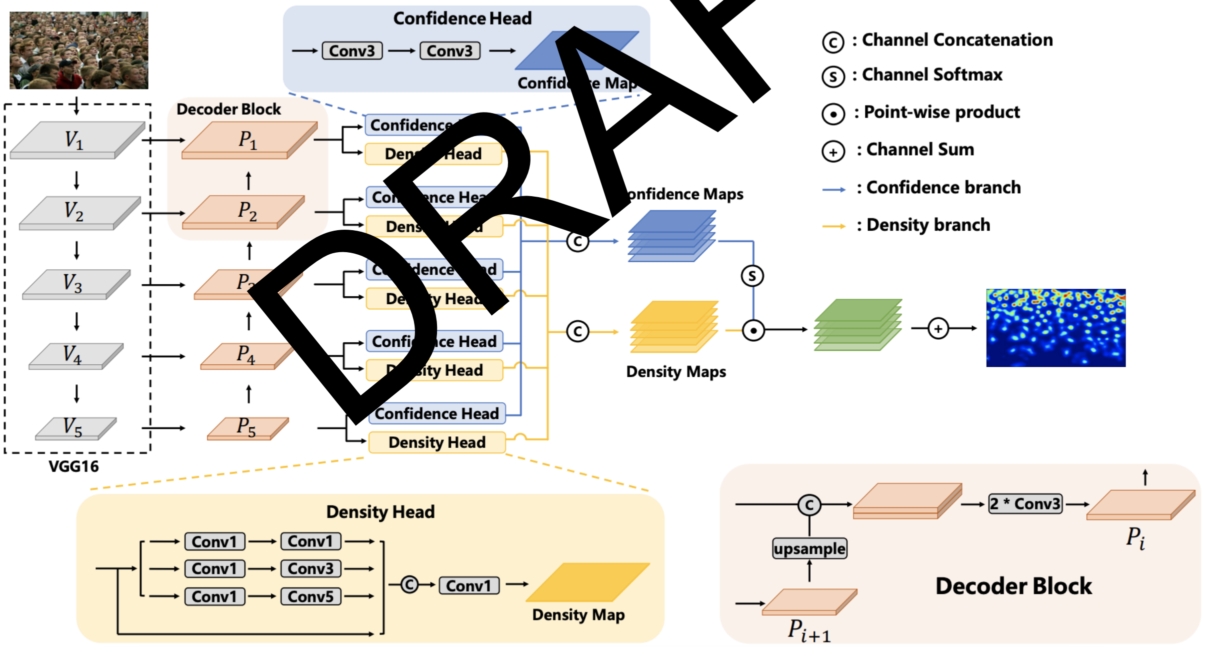


Figure 4: SASNet architecture. Image taken from Song, Qingyu, et al. "To choose or to fuse? scale selection for crowd counting." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 35. No. 3. 2021.

Similar to VCC, the audio-visual crowd counting (AVCC) model is also responsible for counting the total number of people that are present in a given scene. However, AVCC is the multi-modal version of this task where ambient audio from the scene is also available in addition to the image. Thus, alongside the high-resolution RGB image of the scene, the input to AVCC (as mentioned in Section 2) also includes a 1-second raw audio waveform taken from the scene, starting 0.5 seconds before the image was

[15]https://paperswithcode.com/dataset/shanghaitech

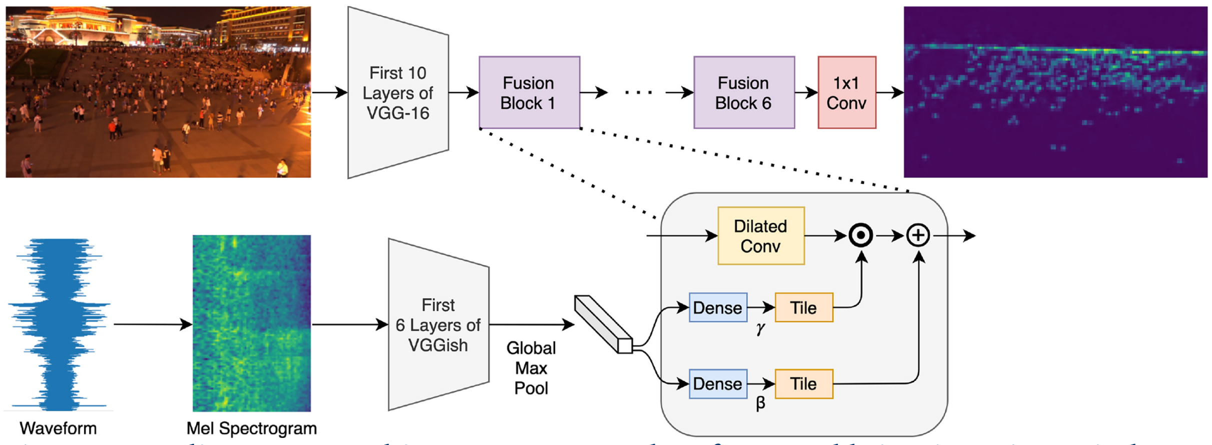


Figure 5: AudioCSRNet architecture. Image taken from: Bakhtiarnia, Arian, Qi Zhang, and Alexandros Iosifidis. "Single-layer vision transformers for more accurate early exits with less overhead." Neural Networks 153 (2022): 461-473.

captured and ending 0.5 seconds afterwards. Similarly, the output of this model is a density map detailing the density of the crowd for every pixel of the input image, which can be summed to obtain the total count. This model is also trained using only visual annotations in the form of head annotations, and the audio is not annotated. Features extracted from the ambient audio of the scene are especially useful in situations where the quality of the image is low, for instance, due to low illumination, occlusion, low resolution or noisy capture device. This model uses the AudioCSRNet architecture, shown in Figure 5, which is a high-performing audio-visual DNN on the DISCO dataset [18]. In this architecture, the raw waveform is not directly used as the input to the DNN; instead, the raw waveform is first converted to the frequency domain to obtain a mel spectrogram, which is then passed on to the DNN. Mel spectrograms are a special type of spectrogram that uses frequency bins that are adjusted based on the human auditory system.

3.2 FedL deployment

The deployment of FedL in MARVEL R1 is done across each of the three MARVEL pilots, and specifically within use cases 1) GRN4 Traffic junction trajectory collection, in the Malta pilot; 2) MT1 Monitoring of crowded areas, in the Trento pilot; and 3) UNS1 Drone experiment, for monitoring large space public events, in the experimental pilot in Novi Sad. For each pilot, the respective FedL client is deployed at the pilot's fog layer, the details of which are provided in Section 3.2.1. The deployment of the FedL clients and the server is achieved through the MARVdash Kubernetes dashboard, the details of which are given in Section 3.2.2. We next discuss details of the infrastructure used for FedL deployment within MARVEL R1.

3.2.1 Infrastructure at MT, GRN, UNS

MT FedL fog layer. The fog tier architecture of MT uses cases, designed and hosted by FBK, implements a peculiar and articulated solution. To account for data access restrictions introduced by the MT DPO, which allows only FBK to access raw data from a centrally managed workstation, the fog tier is split into two separate nodes to dis-

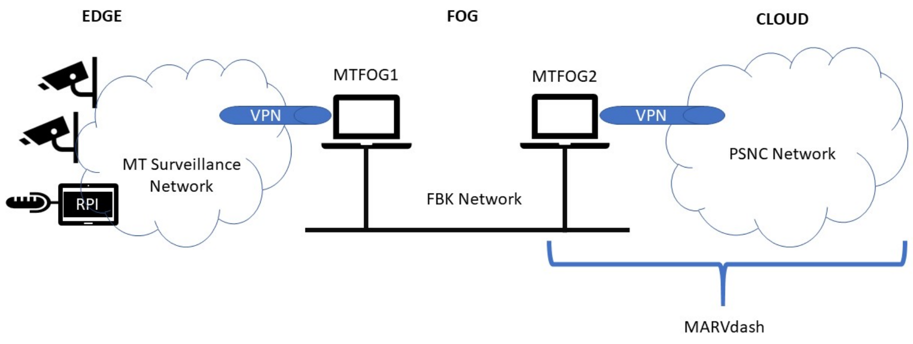


Figure 6: Infrastructure of the MT fog, implemented at FBK premises (taken from D5.4)

entangle the edge layer (which is part of the MT operational network) and the Kubernetes cluster managed via MARVdash. Details about this specific design, the protocol and software implementations, and the characteristic of the two nodes are available in D5.4. As insight of the overall architecture design of the MT use cases, with emphasis on the fog tier, we report here Figure 6 from D5.4. Briefly, MTFOG1 connects via VPN to the cameras and microphones and is not part of MARVdash. MTFOG2 is within the Kubernetes cluster and communicates with MTFOG1 via the local area network. In this way: access to MTFOG1 is restricted and controlled by standard FBK's security protocol; anonymised video and audio streams are made available by MTFOG2 to the nodes in the Kubernetes cluster and that host the components developed by the consortium. Table 1 shows the specifications of the FBK fog layer.

Table 1: Specifications of the FBK fog work-station in the Kubernetes cluster.

Hardware component	Specifications
CPU	Intel Xeon E5-1620
GPU	Tesla K40 11GB
Hard Drive	512GB
RAM	20GB

GRN FedL fog layer. GRN has provided a workstation with GPU as part of the fog layer. The AV streams from the Zejtun Cameras will be processed on the fog layer. Table 2 shows the specifications of the GRN fog layer.

UNS FedL fog layer. For FedL training, a SUPERMICRO SYS-7049A Server SuperWorkstation node is used, with the specifications presented in Table 3. Additionally, there is a local network storage appliance: QNAP TVS-682-8G with 4 $\times$ 3TB SATA3 disks under RAID protection, with NFS/SMB/S3 protocols enabled. All infrastructure components are locally connected with multiple redundant 1G Ethernet links. UNS infrastructure is currently hosted "behind" an HTTP(S) proxy for Internet connectivity.

FedL cloud server. The server for the FedL component is deployed in the cloud layer of the MARVEL architecture. The only requirement for its deployment is a working net-

Table 2: GRN fog work-station specifications.

Hardware component	Specifications
CPU	Intel core I 7 11700K
GPU	RTX3070 8GB
Hard Drive	2Tb
RAM	32Gb

Table 3: UNS fog server specifications.

Hardware component	Specifications
CPU	2 × Intel Xeon Silver 4110 2.1 GHz 8C/16T CPU
GPU	Nvidia TitanXP
Hard Drive	3 × 300GB SSD, 2 × 1TB SSD
RAM	128 GB DDR4

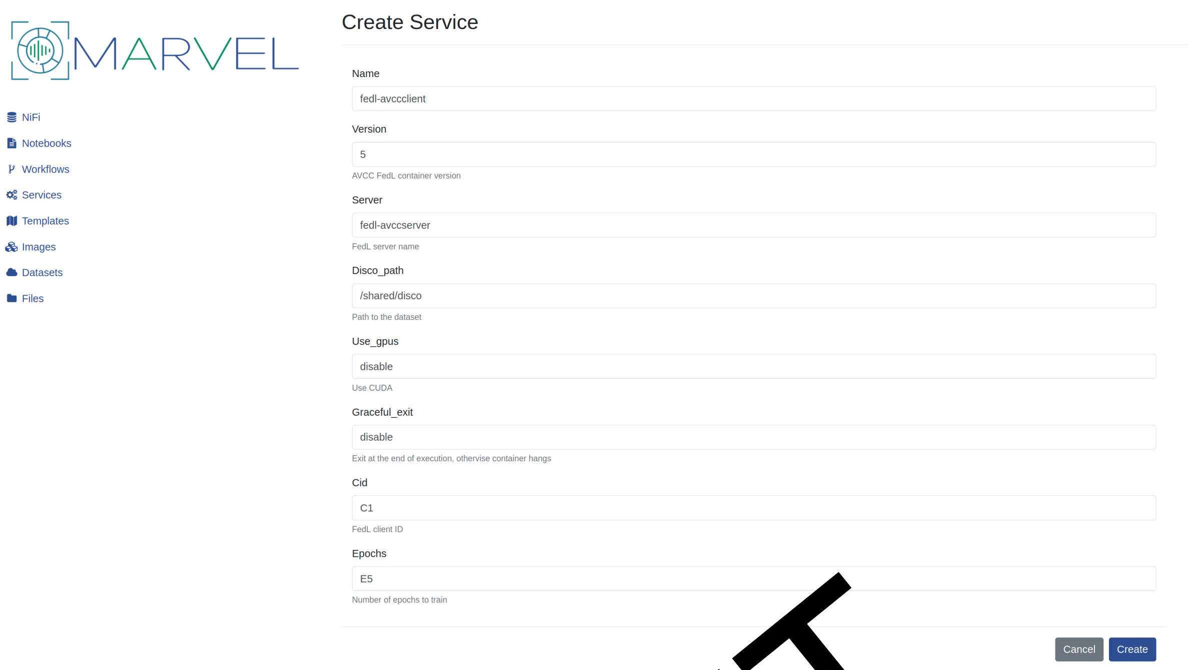
work configuration so that clients can be reached and vice-versa. In our configuration, exactly one FedL server is deployed per each model being currently trained/in use.

3.2.2 MARVdash deployment

Both the FedL client and server components can be run in a containerised environment (Docker/Kubernetes). They were designed from the initial phases to be compatible and suitable for use in the MARVEL system architecture.

Both components are built with specific Dockerfiles through which they can also receive configuration parameters. Specific to MARVEL, we also developed deployment templates for the MARVdash web interface as can be seen in Figures 7 and 8. Many of the parameters (such as data path, number of epochs, GPU usage, and so on) can be configured through the Web interface of MARVdash. We also allow configurable execution of clients in specific parts of the MARVEL architecture (e.g., cloud or specific fog locations). On the server side, we allow for configuring of chosen strategy and number of rounds.

When the services are deployed (see Figure 9), we can monitor and inspect their state through the built-in Kubebox MARVdash service (Figure 10).



Create Service

Name
fedl-avccclient

Version
5
AVCC FedL container version

Server
fedl-avccserver
FedL server name

Disco_path
/shared/disco
Path to the dataset

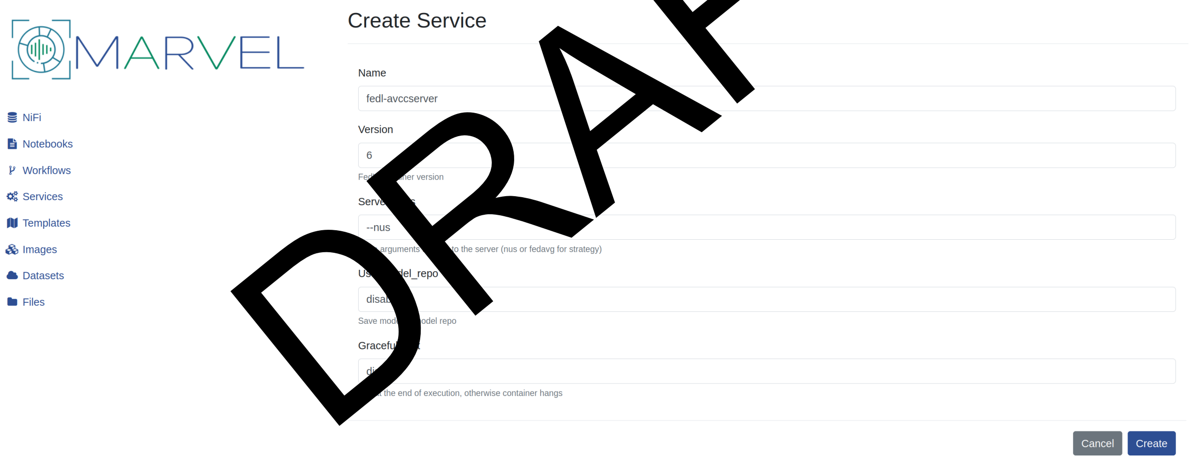
Use_gpus
disable
Use CUDA

Graceful_exit
disable
Exit at the end of execution, otherwise container hangs

Cid
C1
FedL client ID

Epochs
E5
Number of epochs to train

Figure 7: FedL client being set up for deployment at MARVdash platform.



Create Service

Name
fedl-avccserver

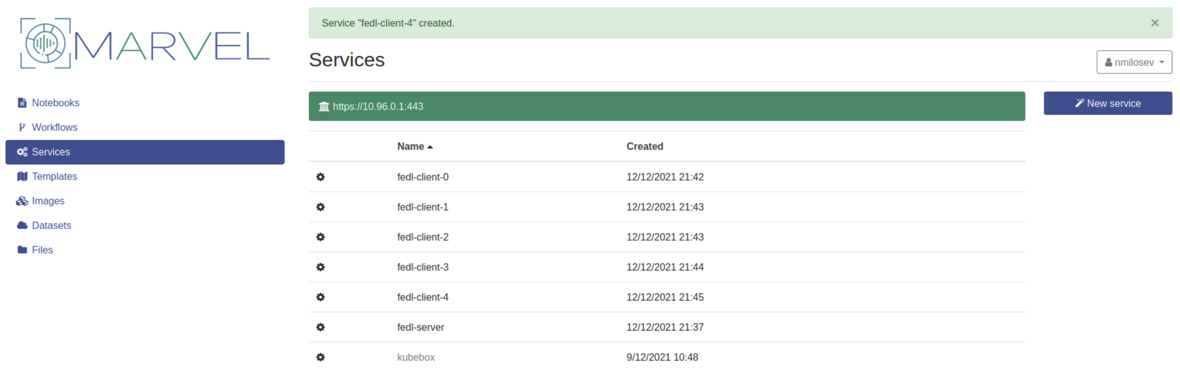
Version
6
Fe[illegible]ner version

Serve[illegible]s
--nus
[illegible] arguments [illegible] to the server (nus or fedavg for strategy)

Us[illegible]del_repo
disa[illegible]
Save mod[illegible]odel repo

Graceful[illegible]
d[illegible]
[illegible] the end of execution, otherwise container hangs

Figure 8: FedL server being set up for deployment at MARVdash platform.

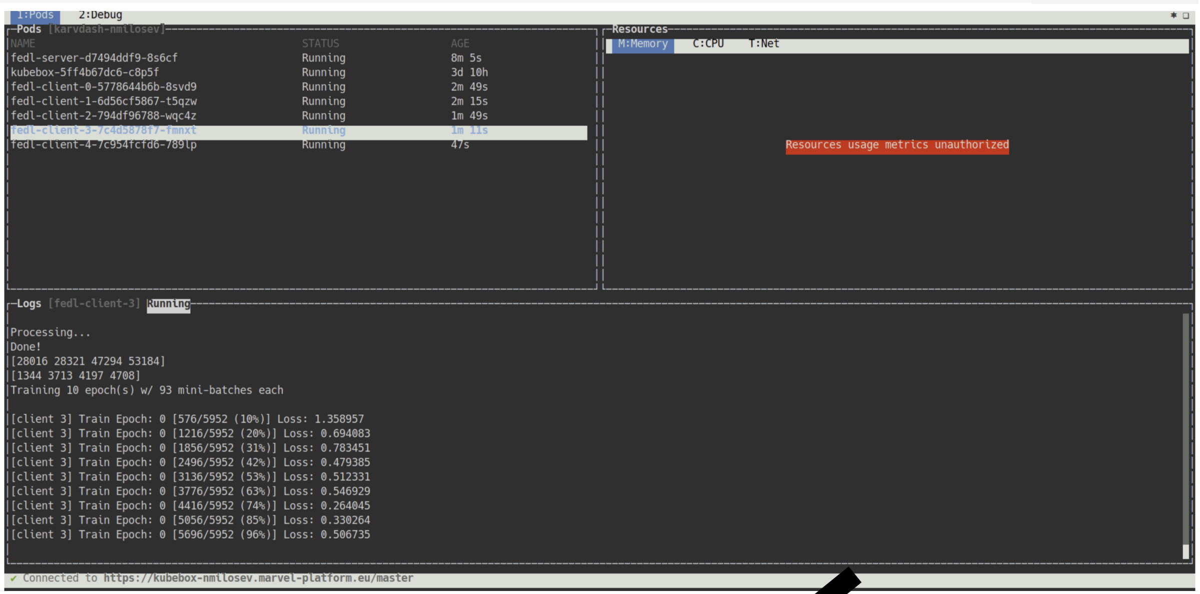


Service "fedl-client-4" created.

Services

	Name	Created
⚙	fedl-client-0	12/12/2021 21:42
⚙	fedl-client-1	12/12/2021 21:43
⚙	fedl-client-2	12/12/2021 21:43
⚙	fedl-client-3	12/12/2021 21:44
⚙	fedl-client-4	12/12/2021 21:45
⚙	fedl-server	12/12/2021 21:37
⚙	kubebox	9/12/2021 10:48

Figure 9: FedL components running in MARVEL cloud, both clients and server deployed as standalone components.



```
 1:Pods     2:Debug
┌Pods [karvdash-nmilosev]--------------------------------------------   ┌Resources-
|NAME                              STATUS          AGE                  | M:Memory   C:CPU   T:Net
|fedl-server-d7494ddf9-8s6cf       Running         8m 5s
|kubebox-5ff4b67dc6-c8p5f          Running         3d 10h
|fedl-client-0-5778644b6b-8svd9    Running         2m 49s
|fedl-client-1-6d56cf5867-t5qzw    Running         2m 15s
|fedl-client-2-794df96788-wqc4z    Running         1m 49s
|fedl-client-3-7c4d5878f7-fmnxt    Running         1m 11s
|fedl-client-4-7c954fcfd6-789lp    Running         47s                               Resources usage metrics unauthorized

┌Logs [fedl-client-3] Running-----------------------------------------
|
|Processing...
|Done!
|[28016 28321 47294 53184]
|[1344 3713 4197 4708]
|Training 10 epoch(s) w/ 93 mini-batches each
|
|[client 3] Train Epoch: 0 [576/5952 (10%)] Loss: 1.358957
|[client 3] Train Epoch: 0 [1216/5952 (20%)] Loss: 0.694083
|[client 3] Train Epoch: 0 [1856/5952 (31%)] Loss: 0.783451
|[client 3] Train Epoch: 0 [2496/5952 (42%)] Loss: 0.479385
|[client 3] Train Epoch: 0 [3136/5952 (53%)] Loss: 0.512331
|[client 3] Train Epoch: 0 [3776/5952 (63%)] Loss: 0.546929
|[client 3] Train Epoch: 0 [4416/5952 (74%)] Loss: 0.264045
|[client 3] Train Epoch: 0 [5056/5952 (85%)] Loss: 0.330264
|[client 3] Train Epoch: 0 [5696/5952 (96%)] Loss: 0.506735
|
✓ Connected to https://kubebox-nmilosev.marvel-platform.eu/master
```

Figure 10: Monitoring of FedL components running in MARVEL cloud, through Kubebox service.

3.3 FedL Performance Evaluation

In this section, we describe benchmarks of our FedL component and more specifically the custom NUS strategy both on standard FL benchmark datasets such as from the LEAF benchmark [2] and the actual MARVEL datasets. For benchmarking, we monitor model performance (loss curves, validation metrics) and data transfers as our strategy aims to minimise data transfer while preserving performance.

3.3.1 Benchmarks with FedAvg baseline

In our first experiments, we test the NUS strategy and compare it to FedAvg strategy on two widely-used image classification datasets: MNIST [3] and FashionMNIST [19]. MNIST dataset contains images of hand-written digits and it is often used nowadays as a deep learning testing dataset, while FashionMNIST contains images of clothing in the same format as the MNIST dataset. Both datasets contain grayscale images of 28 by 28 pixels. While simplistic, these datasets proved useful to validate our code for correctness.

In both use cases, we see similar behaviours with our strategy. Loss values over training epochs have a similar curvature, and while global loss is slightly higher for our NUS strategy, performance remains unaffected. The data savings become larger as the model grows, where one of the main benefits of our strategy comes into light.

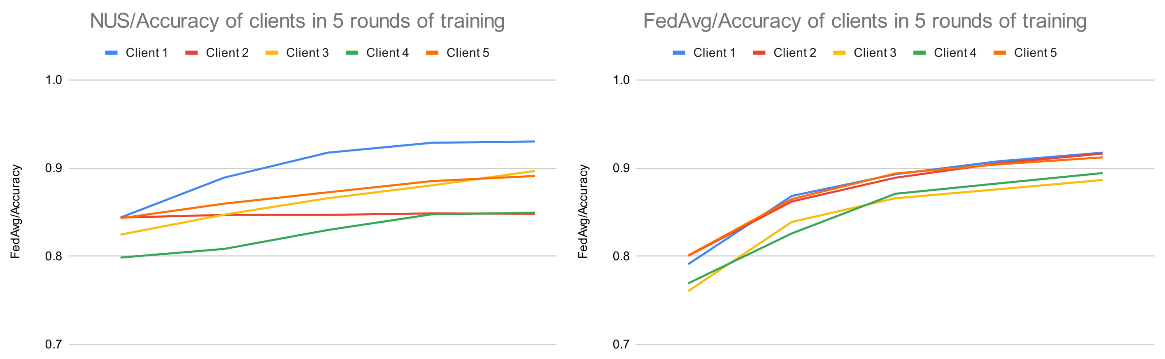


Figure 11: FedL training process with the MNIST dataset.

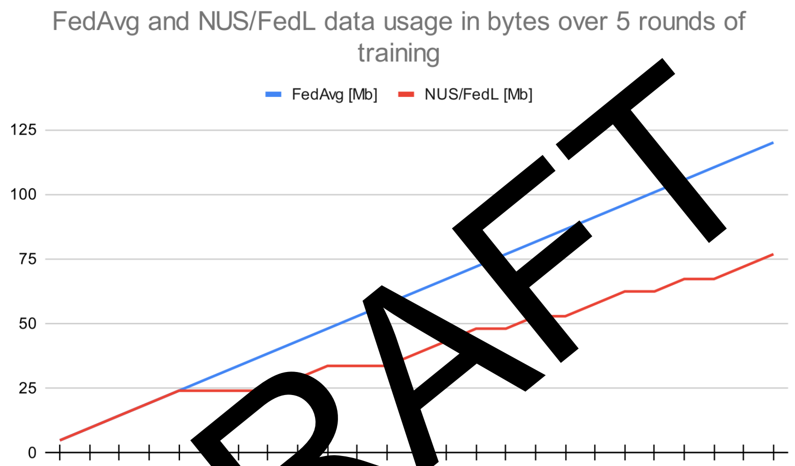


Figure 12: Data transfer overview of FedL training process with the MNIST dataset.

In Figures 11 and 12 we observe training curves on the MNIST dataset, where we see similar global accuracy (close to 0.9) with significant (up to 40%) data transfer savings compared to FedAvg baseline. It is important to note that we simulate a realistic scenario where we do not divide the dataset equally and independently over clients, but rather use random splits and random sampling.

In Figure 13 we see a similar experiment for the Fashion MNIST, FMNIST, dataset[16], where we monitor values of the loss function and the amount of the data being transferred. We again see similar loss curves with significant data savings and we also note similar validation global accuracy values as can be seen in Figure 14. Although this issue has not occurred in the reported simulation, we would like to note that some clients may become starved (if they have received a small partition of the dataset in our random splitting methods) and their performance may stagnate. This can be related to the nature of the dataset a client is employing in the training process. Some clients may work with a smaller dataset that has a higher variance, or with a dataset with a significantly different underlying distribution than the rest of the clients which may cause the server to rarely poll them. This issue is apparent only in smaller datasets, and we will work on correcting this issue in the future.

[16]Data set available at https://github.com/zalandoresearch/fashion-mnist.

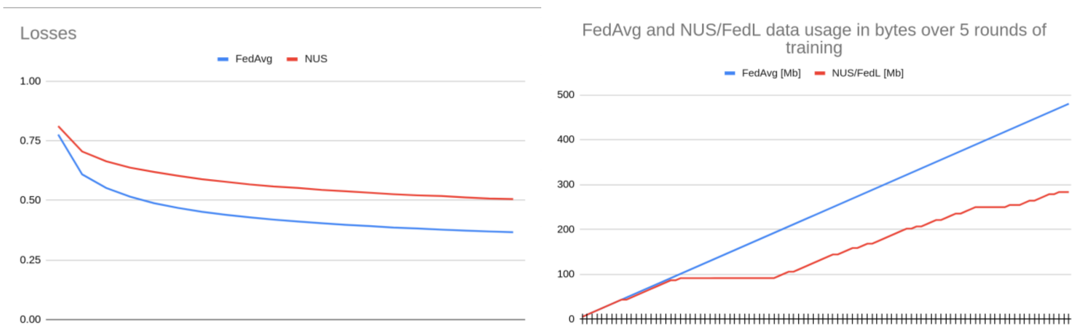


Figure 13: FedL training process with the FMNIST dataset.

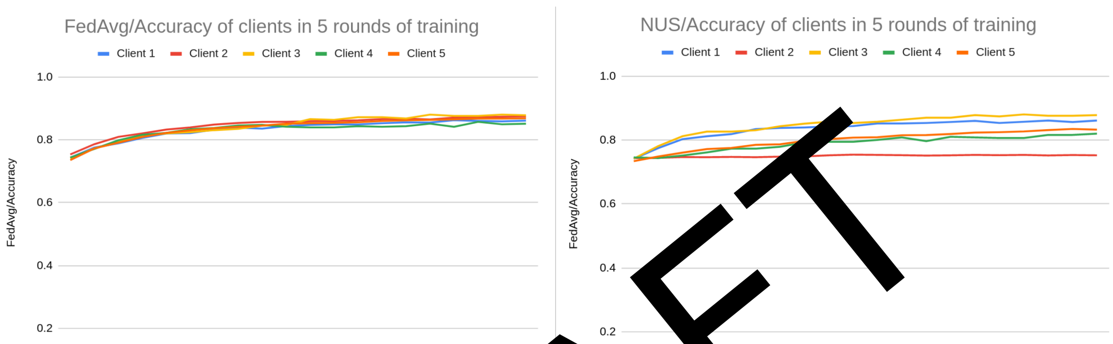


Figure 14: Data transfer overview of FedL training process with the MNIST dataset.

3.3.2 Benchmarks with MARVEL data

After successfully verifying our strategy implementation on smaller datasets, we now benchmark on real MARVEL data. We explain next the details of the datasets employed and the experiments carried out.

VCC task. For these tests we used the crowd counting task which is relevant for each of the MARVEL pilots, specifically for 1) GRN4 Traffic junction trajectory collection, to monitor the number of people at junctions, including bus stops; 2) MT1 Monitoring of crowded areas, to estimate crowd density/number of people; and 3) UNS1 Drone experiment, for monitoring crowd behaviour in large space public events. The model used is a visual crowd-counting neural network model SASNet (as described in earlier sections of this document), where we note that the code was further extended (the client side in particular) to be compatible with FedL.

Datasets. Each of the three pilot sites, namely, GRN, MT and UNS, had a local dataset acquired, anonymised and annotated previously for the VCC training tasks. MT has provided the dataset ”TrentoOutdoor – real recording” (as defined in D2.1[17]) processing the streams of cameras from Piazza Duomo and Piazza Fiera, respectively. MT, in collaboration with FBK, collected and anonymised more than 500 videos, out of which 170 videos were annotated for VCC. The videos were annotated using the CVAT software [20] to provide the number of detected people, with a total of 700 frames. GRN contributed with more than 660 video snippets from GRN static cameras, in the Zejtun and Mgarr locations. All data was subsequently anonymised (faces, licence plates) and

[17]MARVEL D2.1: Collection and analysis of experimental data, 2021. https://doi.org/10.5281/zenodo.5052713

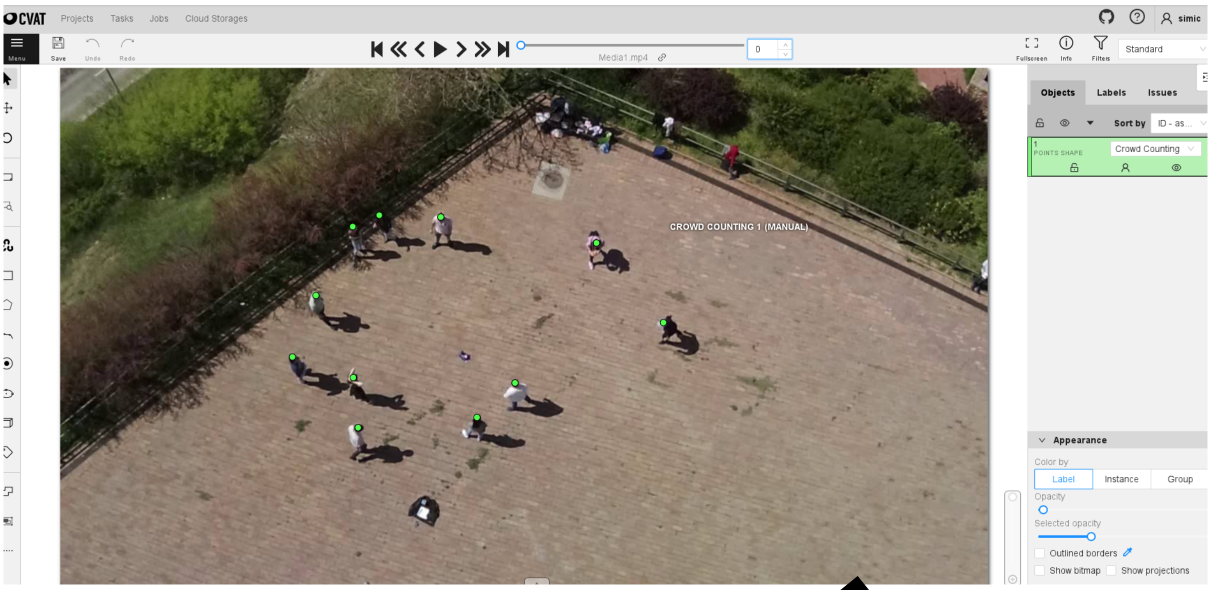


Figure 15: User interface of CVAT showing one annotated frame from UNS drone-based recordings.

annotated using CVAT software. For the VCC task [illegible] frames that contain pedestrians were selected for VCC training. UNS collected the dataset for VCC within the staged recording that was carried out at Petrovaradin fortress, in Novi Sad. For recording purposes, the DJI P4 Multispectral Drone was used. Video capturing was performed from different heights varying between 10-[illegible]0 meters. Videos were recorded using the following configuration: HD resolution (720x1280 pixels), 30fps (frames per second), H.264 codec and MP4 format. No anonymisation was necessary as all participants signed consent for the recording; we also note that in this type of experimental setup, identification of individuals is generally difficult due to the higher and overhead camera position. In total around [illegible]00 annotated frames are provided for VCC training. The figure below shows one annotated frame from the UNS staged recordings in the user interface of CVAT.

Experiments. To evaluate the performance of FedL on MARVEL data, we similarly as before monitor model performance (loss, accuracy metrics) and data transfers. The results are presented in Table 4. Firstly, we experimented with the FedAvg strategy in FedL (Figure 16). In this experiment we see that the global model improves validation metrics (since this is a crowd counting model, we use mean absolute error) for one out of three use cases, the MT use case. For the two other use cases the performance remains similar. We believe this is because the MT use case model benefits greatly from the UNS data which contains similar image data taken from a similar viewpoint. Additional data helps with the model performance, and because inputs are similar we see a performance improvement.

In the second experiment we use our custom NUS strategy (Figure 17) and note similar loss curves (mean and standard deviation values) to both the FedAvg baseline and standard non-federated approach where local training only was used.

Regarding the performance, we see that the NUS approach outperforms both the FedAvg approach and the standard non-federated, local learning approach in two use cases: UNS and MT. The GRN use case has performance degradation, and while monitoring the client selection process we noticed that our strategy rarely selects this client. This is because the distribution of the data samples in MT and UNS use cases are quite similar while for GRN the corresponding distribution is different. For this reason it becomes "starved" of updates in our process. Both the UNS and MT use cases benefit (as a majority) from the global model, while we somewhat degrade the performance of the GRN use case which works better with a specific model trained on its data.

Since the model is large, the data savings are large as well. Over one training process of 100 epochs we see reduction in model parameter transfers of around 23 gigabytes. Despite this, the global model (for two out of three) use cases outperforms the use case-specific models, which was the main goal of our strategy: to retain or increase performance while reducing data transfers.

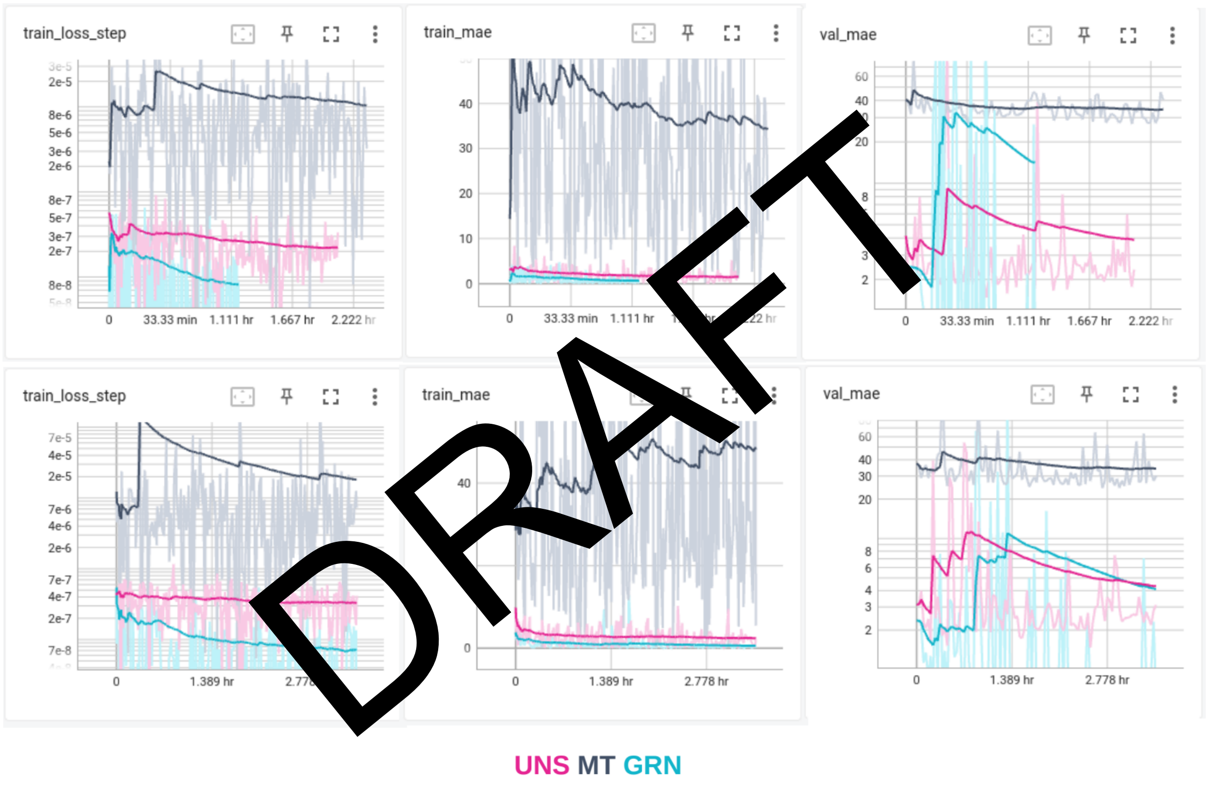


Figure 16: FedL training process with the SASNet Crowd Counting model and MARVEL data. On the top we see loss curves of the non-federated standard approach, and on the bottom the FedL loss curves (FedAvg strategy).

Table 4: Overview of final validation metrics (mean absolute error) for the SASNet model with local training, federated learning with FedAvg, and federated learning with NUS. Significant improvement for the MT use case.

	FedL w/ NUS	FedL w/ FedAvg	Local training
UNS	2.08898305892944	3.10862731933594	2.36223340034485
MT	27.6216239929199	30.0927715301514	40.3407440185547
GRN	9.12610340118408	0.690046668052673	0.626943290233612

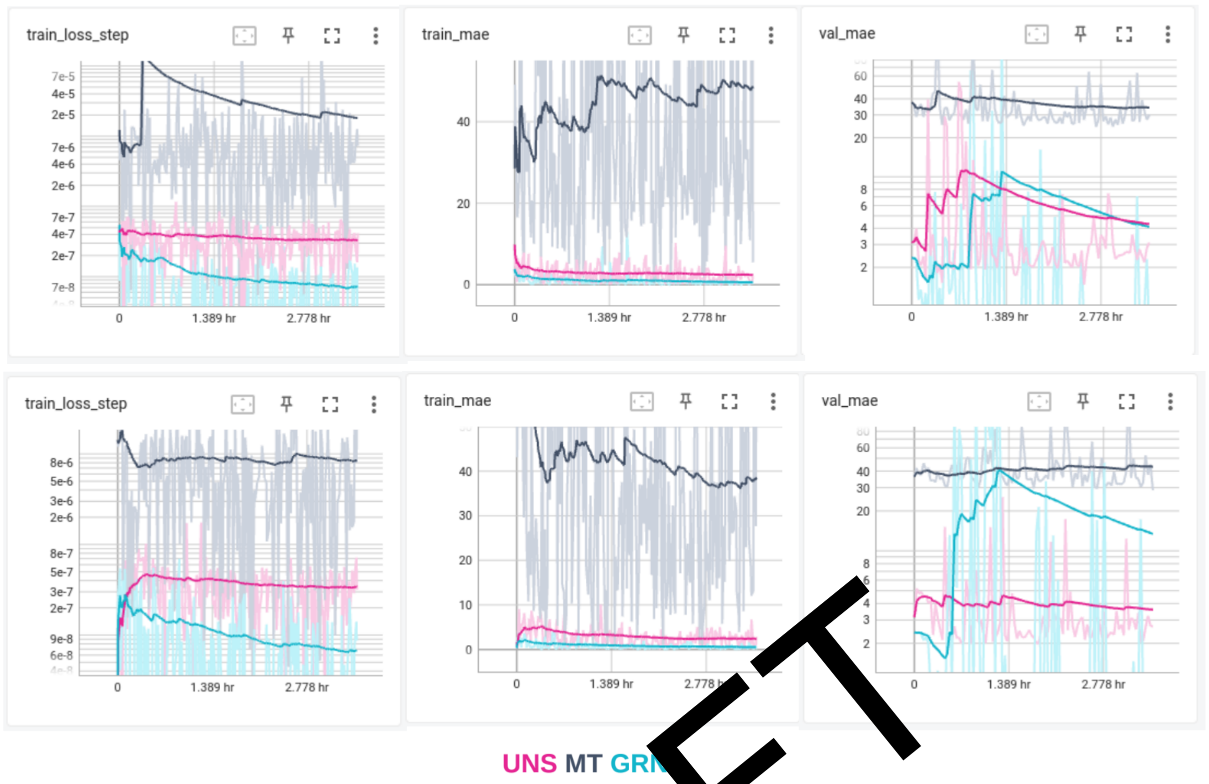


Figure 17: FedL training process, NUS strategy enabled, with the SASNet Crowd Counting model and MARVEL data. Top curves are of FedAvg experiment, bottom are NUS experiment.

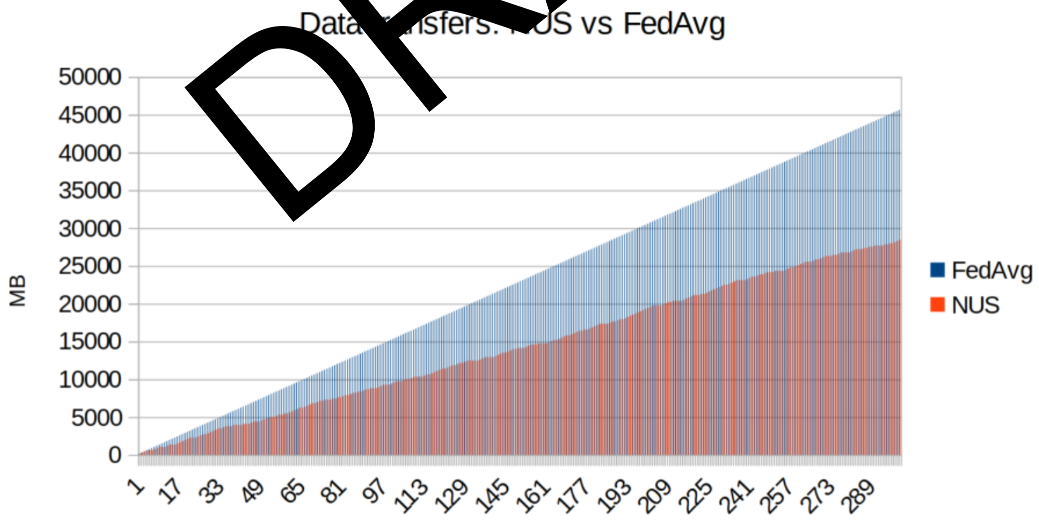


Figure 18: Overview of data transfers for the FedL training process, NUS strategy enabled, with the SASNet Crowd Counting model and MARVEL data.

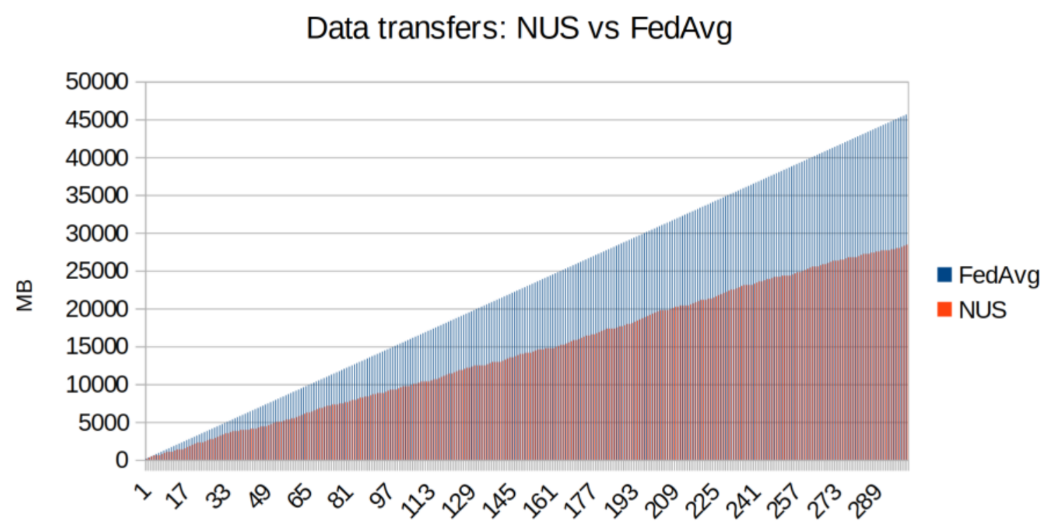


Figure 19: Overview of data transfers for the FedL training process, NUS strategy enabled, with the SASNet Crowd Counting model and MARVEL data.

3.4 FedL conclusions and future work

3.4.1 Contribution to MARVEL KPIs

The delivery of the FedL with the NUS strategy with the achieved performance on both standard benchmarks and real-life MARVEL data, reported in Section 3.3, directly enable fulfillment of the iKPI-3.1 and iKPI-2.1, and contribute to the fulfillment of the KPIs KPI-02-E1-1 and iKPI-1.1, as detailed in Section 1.6. The differential privacy module of FedL contributes to the fulfillment of the KPI KPI-O1-E3-1, details of which can also be found in Section 1.6.

3.4.2 FedL in R2

FedL as a component is feature complete and ready to use. We are still experimenting with various parameters and trying to further improve our client selection process in the custom NUS strategy.

For the MARVEL data, we experimented with the crowd counting use cases. It would be also interesting to experiment with the audio-visual anomaly detection use cases. The risk in this is that anomaly detection data we possess so far are very specific to the location where they are acquired. A global model may not be able to converge on all the data, but this remains to be experimented with in the future.

PART II: Further contributions to personalised federated learning

4 Personalised Federated Learning via Convex Clustering

In this section, we briefly describe the results that have been published in the paper [21], Appendix B, and where MARVEL UNS researchers are two paper co-authors.

4.1 Context and State-of-the-art

Personalised federated learning generally refers to federated learning approaches that, besides generalisation harnessed by accounting for the data from multiple FL users, also enable personalisation. That is, each user in general can learn a distinct model that is fine-tuned to the users' local data. More formally, with personalised FL over N users, we would like to solve the following optimisation problem:

$$\text{minimize}_{x_1,\ldots,x_N\in\mathbb{R}^d}\ \frac{1}{N}\sum_{i=1}^{N} f_i(x_i), \tag{1}$$

subject to appropriately defined constraints. Here, f_i is the loss function associated with i-th user, $i = 1, ..., N$. Various types of constraints can be introduced in (1), giving rise to different personalised FL approaches. They include, for example, multi-task learning [22], [23], fine-tuning [24], [25], knowledge distillation [26], [27], [28], [29] and clustering-based approaches [30], [31], [32]. A notable approach is given in [33], that proposes a formulation which adds in the objective of (1) the following term: $\lambda \sum_{i\neq j} \|x_i - x_j\|^2$, where $\lambda > 0$ is a regularisation parameter, and $\|\cdot\|$ is the Euclidean norm. The role of the regularisation, depending on the value of λ, is to trade-off the generalisation and personalisation abilities of the FL model. A smaller λ means more distinct local models and hence a stronger degree of personalisation, but also a lower generalisation ability in general.

4.2 Description of the method

In paper [21], we have proposed a novel approach to personalised FL. The approach makes possible simultaneous personalisation, generalisation, and model clustering. Specifically, the paper [21] introduces the following novel formulation:

$$\text{minimize}_{x_1,\ldots,x_N\in\mathbb{R}^d} F(x_1,\ldots,x_N) = \frac{1}{N}\sum_{i=1}^{N} f_i(x_i) + \lambda \sum_{j\neq i} \|x_i - x_j\|. \tag{2}$$

Here $\lambda > 0$, is a parameter, and $\|\cdot\|$ denotes the Euclidean norm. Compared with (1), formulation (2) also involves regularisation, controlled with $\lambda > 0$. The regularisation forces that the local solutions for some pairs of nodes be close one to another.

Formulation (2) is related to reference [33]. However, instead of the sum of norms of pairwise distances, [33] uses the sum of squared norms of distances; this leads to several distinctive features of (2). In more detail, the contributions of [21] are the following. First, it gives a new formulation for personalised FL (2). The solution of the formulation exhibits a clustering structure while at the same time preserving generalisation abilities. Furthermore, the reference gives a condition on the penalty λ that leads to clustering of the users' personalised models according to a ground truth unknown beforehand. Next, the method in [21] works without knowing the true number of clusters (distinct models) in advance. Moreover, [21] develops an efficient algorithm for federated learning over conventional FL architectures with the server-users topology that solves (2), using the Parallel Direction Method of Multipliers (PDMM) [34]. Finally, numerical studies corroborate findings and compare the achieved results with alternative state-of-the-art personalised FL approaches in [23].

4.3 Application and relevance to MARVEL

The proposed method is of direct relevance to MARVEL smart city scenarios. For example, for a crowd counting task, it might be expected that different cities exhibit similarities of their “crowd distributions” in the relevant public areas (e.g., main city squares). This may be addressed through the proposed personalisation via clustering approach, which automatically clusters the cities (i.e., models) according to this similarity. This method contributes to the fulfillment of the KPIs KPI-02-E1-1 and iKPI-1.1, as detailed in Section 1.6.

4.4 Evaluation

We next present evaluation results for the method in [21]. The experimental setup is as follows. A supervised binary classification problem is considered. The data has $K = 3$ clusters. For each cluster, for each label/class (± 1), the data is uniformly distributed over a 2D ellipse. There are 200 (training) data points for each cluster, 100 points per class. There are 20 FL users belonging to each cluster. Each user has 10 data points. The data for all users in the same cluster is from the same distribution. We use the squared Hinge loss function; that is, at user i, the loss function is given by:

$$f_i(x) = c\,\frac{\|w\|^2}{2} + \frac{1}{m}\sum_{j=1}^{m} \max\left\{0, 1 - \ell_{ij}(\langle w, a_{ij}\rangle - b)\right\}^2.$$

Here, m represents the number of (local) data points, $(a_{ij}, \ell_{ij})_{j=1}^{m}$, are the data points and class labels at user i, $c > 0$ is a penalty parameter that controls the regularisation, while $x = \left[w, b\right] \in \mathbb{R}^{d+1}$, $d = 2$, is the vector that defines the classifier. We set $c = 10^{-3}$.

In order to benchmark the method in [21], the method is compared with that of [23], the non-personalised (global) FL, and the users' models trained by each user in isolation. In addition, an oracle model is considered that (for an unfair setting) knows beforehand the clustering structure. For the oracle model, for each user i in cluster C_k, the oracle lets user i's classifier vector equal to $x_i = \arg\min_x \sum_{j \in C_k} f_k(x)$. All problem formulations are solved using CVXPY [35], [36].

We measure testing accuracy of the corresponding classifiers. In more detail, consider x_i – a classifier vector for user i obtained through training via any of the considered methods. For each user i, testing accuracy of x_i is evaluated with respect to the full testing data set for the cluster to which user i belongs. Then, average testing accuracy across all users $i = 1, \ldots, N$ is evaluated. The testing data is generated by drawing new samples from the same distributions according to which the training data is generated.

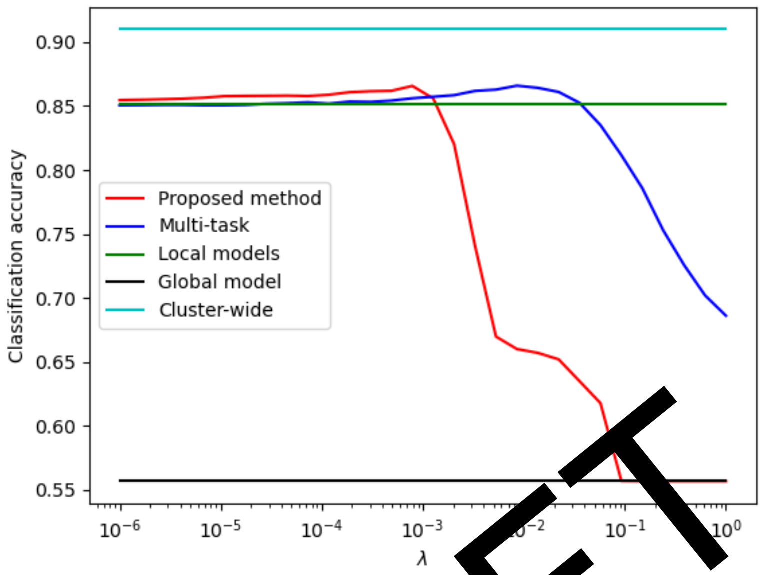


Figure 20: Average classification accuracy across all users and all clusters versus λ.

Figure 20 shows the testing accuracy for all the considered methods. We can see that, clearly, the cluster-wide one that knows the cluster structure beforehand exhibits the best performance. The method in [21] exhibits a peak accuracy that is comparable with a state of the art method [23]. At the same time, the proposed method produces clustered models, unlike the method in [23].

For the performance of the proposed PDMM solver, we refer the reader to [21, 37].

4.5 Conclusions

Reference [21] has proposed a novel formulation for personalised FL. A distinctive feature of the formulation is that it enables users' model clustering, i.e., assigning same users' models to the users that belong to the same cluster. The users' clustering structure is not known beforehand and is determined by the similarity of users' local costs and their data. The paper establishes theoretical guarantees on the cluster structure recovery. It also provides a PDMM-based FL algorithm to solve efficiently for the proposed formulation. Numerical evaluations demonstrate that the proposed personalised FL approach exhibits a comparable accuracy with state-of-the-art, while at the same time it produces clustered users' models.

5 One-shot clustered federated learning

In this section, we briefly describe the results in [38], Appendix B, where MARVEL UNS researchers are two work's co-authors.

5.1 Context and State-of-the-art

Clustered federated learning (CFL) refers to scenarios where FL users can be partitioned into disjoint groups (clusters); users in the same cluster have similar data and require mutually equal (similar) models, while users from different clusters require different models. It is then desirable to ensure collaboration among users from the same cluster, and completely avoid, or utilise in a reduced (controlled) way, the effect of data of users from a different cluster. At the same time, the users' assignment across clusters is not known beforehand.

Several recent works study CFL, e.g., [39], [40], [41], [42], [43]. References [39] and [40] develop methods that iteratively estimate cluster membership and perform model updates. The authors of [41] develop a robust CFL method where adversarial users may be present. Reference [42] develops a method that successively bi-partitions the current set of users based on cosine similarity and does not require prior knowledge of the number of clusters K. When a bi-partitioning of users is done, a full FL training over each partition needs to be carried out. To the best of our knowledge, none of the CFL works (other than [38]) is concerned with developing a CFL method that is communication-efficient, i.e., that utilises only one round of communication between the server and the users.

5.2 Description of the Method

Reference [38] proposes a one-shot method for CFL that has the following features. The method is communication-efficient, i.e., it requires only one round of communication, while it does not need the knowledge of the true number of clusters. For strongly convex cost functions, the method achieves the (order-) optimal mean squared error (MSE) decay in terms of the sample size. That is, the MSE decay with respect to the sample size matches that of a hypothetical oracle that knows (in an unrealistic setting) the cluster structure beforehand and performs empirical loss minimisation over all the data that belongs to the same cluster. In other words, it is shown that the method has optimal MSE guarantees with respect to sample size, equivalent to those of centralised learning. This is achieved when the number of users is above a threshold that is also explicitly characterised. Numerical experiments confirm the established theory. They also demonstrate that the proposed method, in the order-optimality regime, significantly outperforms state-of-the-art methods in terms of communication cost.

In more detail, the method in [38] works as follows. Here, f_i is user i's cost function, and Θ is the compact, convex domain that contains the optimal FL model. As a subroutine, the method makes use of convex clustering, a standard method to cluster data points; see, e.g., [44]. Convex clustering has a penalty parameter λ that needs to be appropriately tuned for a successful clustering.

- Each user i, $i = 1, ..., m$, with m being the number of users, gets its local model $\widehat{\theta}_i$, via
$\widehat{\theta}_i = \arg\min_{\theta_i \in \Theta} f_i(\theta_i)$ and sends $\widehat{\theta}_i$ to the server.

- The server receives $\{\widehat{\theta}_i\}_{i=1}^m$, sets a convex clustering penalty value λ from an appropriately defined interval (see [38] for details), and then it performs convex clustering with the local models as inputs.
- For each obtained cluster $C'_{k'}$, $k' \in [K']$, the server performs within-cluster averaging of local models: $\overline{\theta}_{k'} = \frac{1}{|C'_{k'}|} \sum_{i \in C'_{k'}} \widehat{\theta}_i$.
- The server sends the models to each user according to their cluster assignment, i.e., each user $i \in C'_{k'}$ receives the model $\overline{\theta}_{k'}$.

5.3 Application and relevance to MARVEL

As described in the preceding section, personalisation via clustering is an intuitive and well-defined approach to automatically group data arising in a given smart city application according to the similarity of the underlying data distributions or loss function. An additional desirable property is to be able to identify such "hidden" clusters in a communication-efficient manner in order to minimise the amount of (model) data traffic between the different smart city FL clients during the training process. The proposed method adopts a one-shot strategy to identify cluster structure based on a novel personalised FL formulation from Section 4. This method contributes to the fulfillment of the KPI iKPI-1.1, as detailed in Section 1.6.

5.4 Evaluation

We now present evaluation results for the method in [38]. A linear regression problem is considered. Specifically, for each cluster k, we have that a data pair (a, b) is generated by:

$$b = \langle a, u_k^\star \rangle + \epsilon.$$

Here, $\langle \cdot, \cdot \rangle$ denotes the standard inner product of vectors, ϵ is generated from the standard normal distribution, and we set the number of clusters $K = 4$. The regressors $u_k^\star$ are $d \times 1$ vectors with $d = 20$, where each entry is generated independently from a uniform distribution, as follows: for cluster 1, $U([0, 1])$; for cluster 2, $U([1, 2])$; for cluster 3, $U([-1, 0])$m and for cluster 4, $U([-2, -1])$. Each cluster has $N_k = 100,000$ points, generated in the following way: for each $a \in \mathbb{R}^d$, we choose 5 components in $[d]$ that are drawn from a standard normal distribution, while the other components are set to zero. We measure the error as follows:

$$\ell\left((a, b); u\right) = (b - \langle a, u \rangle)^2.$$

There are $m = 100$ users, where each cluster has 10 users. Each user $i \in C_k$ has n points taken uniformly at random from the corresponding sample N_k. The method proposed in [38] is compared with the following methods:

- Oracle Averaging, i.e., a method that (in an unfair setting) knows the true cluster structure and applies the model averaging on each individual cluster.
- Cluster Oracle, i.e., a method that has access to all data points assigned to the users from the same cluster, and then performing:

$$u_k = \arg\min_u \frac{K}{m} \sum_{i \in C_k} f_i(u),$$

where we recall that the f_i is user i's local cost.

- Local empirical risk minimisers (ERMs), i.e., the method that trains an ERM on each user's data in isolation. Naive averaging, i.e., a method that averages all users' local ERMs obliviously to data heterogeneity.

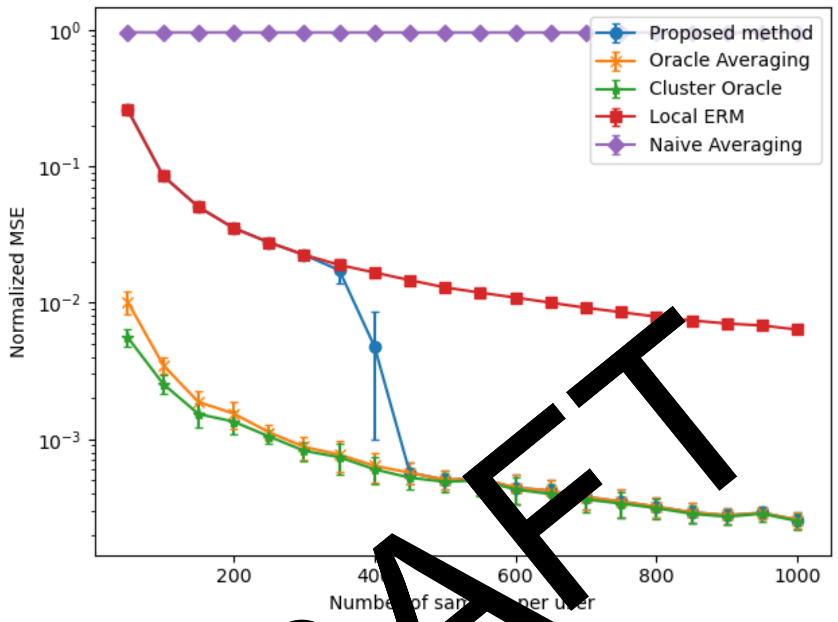


Figure 21: *Left*: Performance of different methods for linear regression, versus the number of samples available per user.

Figure 21 shows performance for all the methods considered. The left figure shows the MSE of different methods. The right figure shows the number of clusters produced by the convex clustering sub-procedure. We can see that the method proposed in [38] performs on par with the oracle methods, i.e., optimally, when the number of samples per user is above a threshold, hence confirming the developed theory.

5.5 Conclusions

In this Section, we briefly reviewed the main results from the paper [38] that proposes a one-shot scheme for communication-efficient clustered federated learning. Future work will investigate performance evaluation for nonconvex, DL models, such as VCC, AVCC, and other models developed within MARVEL.

6 Decentralised and Unsupervised Anomaly Detection

In this section, we present an overview of the paper [45], Appendix C.

6.1 Context and State-of-the-art

Centralised AI assumes that the data generated at the network's edge has to be centrally collected for processing. Although effective, such a paradigm also poses several concerns in terms of data privacy and performance, e.g., it is well known that cloud-based inference might suffer from latency issues. Therefore, the current trend is exploring ways to move part or the entire AI-related process closer to the network's edge and where the data are generated. In this context, new methodologies have been proposed for training models at the edge and performing inference efficiently. Federated Learning is the framework where a set of devices, typically edge servers or edge devices holding a certain amount of local data, collaborate to train an ML model collectively without sharing their data. FL has been extensively investigated assuming that local data is labelled (supervised learning) and the process is centrally coordinated/controlled. However, the vast majority of data at the edge is unlabelled and central coordination is not always possible. In this activity, we explore both aspects: unsupervised and decentralised FL in a heterogeneous environment.

Precisely, we make the following assumptions on the data collection process. First, among all the devices in the system, a subset of them observe the same phenomenon and collect data belonging to the same unknown distribution, e.g., while a set of devices might collect different realisations of a given handwritten letter or symbol, another group might have data belonging to a different letter or symbol. Second, the collected data is unlabelled, i.e., the devices are unaware of which category the collected data belongs to or even if a commonly named category exists. The local data represents only a partial view of the overall distribution, which forces them to find other devices to collaborate with. The computational resources of these devices are limited in that the model trained must be lightweight and cannot learn a wide variety of data patterns (differently from what would be expected from a Deep Network trained in a centralised fashion on a huge amount of data). Finally, the whole system is decentralised: we assume no central coordination during the process, and the devices must coordinate alone.

We consider anomaly detection as a specific application case for this peculiar scenario. In our scenario, the normal data belong to multiple classes (in contrast to the typical AD task involving only a single inlier class). For instance, the methodology proposed, whose output is a set of models each specialised in identifying a single normal pattern, can be further extended with ensemble-based methods to efficiently tackle the multi-class anomaly detection problem, as shown in [46]. The challenge addressed here is the following. In the case of supervised learning, data belonging to each class are labelled, so each node knows which other nodes "see" the same majority class, and therefore forming FL groups is straightforward. In unsupervised cases, each node can detect its majority class from local data but has no direct information to know which other nodes see the same majority class. Therefore, the main objective of our methodology is to identify an effective algorithm for nodes to form consistent groups (i.e., groups that see the same majority class) to run a standard FL process across nodes of the same group.

The literature on decentralised and unsupervised FL is limited compared to the one developed for the supervised learning problem. The first work coping with this problem is [47], where the authors combined federated and unsupervised learning. However, they considered a rather controlled scenario unsuitable for mobile environments. Other solutions considered more challenging conditions like non-IID data distributions and misalignment of representation[48] or coped with domain-specific problems like speech enhancement [49]. Finally, [50] goes in a similar direction w.r.t. ours dealing with anomaly detection in the context of intrusion detection. However, they assume that the local data is labelled.

6.2 Description of the method

We consider a distributed learning system with a set of clients M and a set of data distributions C, such that $|C| \leq |M|$. With data distribution, we refer to a set of identically distributed data representing a specific pattern (e.g., observations of phenomena belonging to the same class of events in case of a classification task). We assume that every client's local dataset is composed of a portion $d \in (0\%, 50\%)$ of samples from a single distribution $C_{out} \in C$, and the remaining $(100 - d)\%$ from $C_{in} \in C$, such that $C_{in} \neq C_{out}$. The sample partitions within each client form the outlier and inlier classes, respectively. This split represents a basic assumption when dealing with AD tasks.

Our methodology is made by two phases:

- **Phase 1: Group identification** to make the clients join a group (i.e. cluster) having the same (or similar) majority class C_{in}.
- **Phase 2. Federated Outlier Detection** to run multiple standard FL sessions among clients in the same group to improve the local models of the devices belonging to the same group.

Phase 1. First, each device trains locally a standard Anomaly Detection model, e.g., One-Class Support Vector Machine. The trained AD model splits the local data between inliers and outliers such that all the devices can discriminate their local anomalies. Second, all the devices share their local AD model with their peers. A device i uses the received models on its local data. The main idea is to test those models on the local data using the performance of their locally trained models as a benchmark. If the performance of the j-model is above some predefined threshold, the client i considers the client j to be in his group. The steps performed by each client are detailed in Algorithm 1.

At the end of algorithm 1, each client has a local view of which other clients should belong to its group. However, due to data locality, different clients in the same group may have different local views regarding the groups' formation. To obtain an overall view of the groups shared by all nodes, we build a graph starting from their local groups. Precisely, since the association of two clients is reciprocal (line 14), we can build an undirected graph from all the resulting groups of candidates for each client. A link between two nodes means that those two nodes mutually "think" to be in the same group. Finally, a community detection algorithm is run on this graph to detect which groups of nodes should be considered part of the same set and thus undergo a standard FL step.

Algorithm 1 Client m_i local training and association

Input: AD Model *Mod*$_\text{i}$, contamination d, association threshold q, set of other clients M

Output: Group G_i of candidate clients similar to m_i

```
 1: procedure LocalAD($Mod_i, d, q, M$)
 2:     $G_i \leftarrow \emptyset$
 3:     $Mod_i = Mod_i.fit(x_i, d)$
 4:     $y_{i,i} = Mod_i.predict(x_i)$
 5:     $in_{i,i} = inlierPercCount(y_{i,i})$
 6:     $send(Mod_i, M)$
 7:     for all $m_j$ in $M$ do
 8:         $Mod_j = receive(m_j)$
 9:         $y_{j,i} = Mod_j.predict(x_i)$
10:         $in_{j,i} = inlierPercCount(y_{j,i})$
11:         $b_{j,i} = in_{i,i} - q \leq in_{j,i} \leq in_{i,i} + q$
12:         $send(b_{j,i}, m_j)$
13:         $b_{i,j} = receive(b_{i,j}, m_j)$
14:         if $b_{j,i}$ $AND$ $b_{i,j}$ then
15:             $G_i \leftarrow m_i$
16:         end if
17:     end for
18:     return $G_i$
19: end procedure
```

Phase 2. The result of Phase 1 is a set of k communities $G_0, \ldots, G_k$. For each of them, we run an FL session where devices train a lightweight autoencoder collaboratively. We selected autoencoders because: (i) being NN-based, they can be trained iteratively as it happens in FL; (ii) they proved suitable for AD tasks. Typically, once trained, it is possible to use the reconstruction error of a given sample to spot potential anomalies. We run k FL sessions in parallel, one for each identified community. In this work, we use the vanilla version of the Federated Averaging (FedAvg) [51]. The final output will be k trained Autoencoders, one for each group highly specialised in the corresponding majority class. These AE can be further used as an ensemble to identify anomalies as shown in [46].

6.3 Relevance for MARVEL

This methodology touches on several aspects of the MARVEL ecosystem. Precisely, this methodology suggests a way for performing anomaly detection using models with limited capacity. In fact, instead of training one model capable to learn different patterns, our idea is to train several lightweight models hyper-specialised on a small number of patterns and exploit their collaboration to perform a complex anomaly detection task (Task 3.5). This fits very well the edge scenario targeted by MARVEL where resource-constrained devices have to perform an anomaly detection task. Moreover, the collaboration between devices is performed under the Federated Learning Framework (Task 3.2) and targets a problem (i.e., federated unsupervised learning) not yet extensively explored. This method contributes to the fulfillment of the KPI iKPI-1.1, providing a tool for solving anomaly detection tasks on distributed and federated edge devices, as detailed in Section 1.6.

6.4 Evaluation

In our evaluation, we use two standard benchmark datasets: MNIST and Fashion-MNIST keeping the original 60000-10000 train-test splits. The data is distributed among several devices using the following procedure: we define a parameter p as the number of clients within the same data distribution (i.e., class), meaning that the train samples of a class C_{in} of the original dataset (e.g., MNIST) are evenly and randomly spread to form the inliers of p clients. Accordingly, the portion of outliers for each client within the same group and characterised by the same C_{in}, is given by the samples of a class different from C_{in}. We ensure that the outlier classes $C \setminus C_{in}$ are equally represented within the group, meaning that for each client of the group, the minority class is "circular" through the set $C \setminus C_{in}$. See an example in Figure 22.

In our system, we used two types of models, one class support vector machine (OC-SVM) [52] for phase 1 and a lightweight autoencoder (AE) for phase 2. OC-SVM requires two parameters to be set: the kernel and the parameter $\nu \in (0, 1]$, which is an upper bound on the fraction of training errors and a lower bound on the fraction of support vectors. The fine-tuning of ν in contaminated data can be challenging without any assumptions on the distribution of the outliers. However, since in our tests we assume to know (only) the contamination value $d = 10\%$ for every dataset, we can set $\nu = 0.1$. Moreover, we use the RBF kernel. The AE has a three-layers topology (64-32-64), ReLU activations on the hidden fully connected layers, and Sigmoid activation on the output layer.

The local association threshold q used by each client to identify its potential group members is set to 0.[illegible]8 (identified empirically). Basically, the local client considers another client as a partner if the model of the latter produces a fraction of normal data on the local dataset equal to the percentage produced by the local node, $\pm q$.

For both the MNIST and the fashion-MNIST datasets, we run four tests varying the value of $p : \{9, 18, 27, 36\}$. In all the tests we use the contamination parameter $d = 10\%$ and we take into account all the available classes, i.e., $|C| = 10$. Let $m_{C_i,j}$ be the j-th client with majority class C_i; we define I_{C_i} as the ideal set of clients having the same majority class C_i, e.g., $I_0 = \{m_{0,0}, \ldots m_{0,p-1}\}$.

In Table 5, we show the results of the community detection phase for the MNIST dataset: we find nine communities, and in most cases, they match with the ideal group of clients. The major exception is given by G_4, that in all four cases is given by the union of I_4 and I_9, meaning that the clients having 4 and 9 as inlier classes join the same community. This is a consequence of the OC-SVM model's inability to distinguish the two digits, and it represents a typical behaviour when dealing with image classification using MNIST.

We compare our methodology with two baselines: (i) **local**, where clients only train on local data; (ii) **ideal**, in which a client $m_{C_i,j}$ uses the model trained through federated learning on the set of clients I_{C_i}, i.e., the set of the clients sharing the same majority class. The test samples for each client are randomly sampled from the MNIST/fashion-MNIST test set, following the same inlier/outlier classes and the ratio of the corresponding client.

Table 5: Community detection for MNIST

(a) $p = 9$

Community ID	Members
G_0	I_0
G_1	I_1
G_2	I_2
G_3	I_3
G_4	$I_4 \cup I_9$
G_5	I_5
G_6	I_6
G_7	I_7
G_8	I_8

(b) $p = 18$

Community ID	Members
G_0	I_0
G_1	I_1
G_2	I_2
G_3	I_3
G_4	$I_4 \cup I_9$
G_5	I_5
G_6	I_6
G_7	I_7
G_8	I_8

Table 6: Test AUC on MNIST. For each p, mean $\pm$ std are computed on $p|C|$ clients

p	Local	Community (ours)	Ideal
9	0.773 ± 0.205	0.836 ± 0.18	0.839 ± 0.185
18	0.769 ± 0.207	0.835 ± 0.18	0.836 ± 0.181
27	0.77 ± 0.208	0.836 ± 0.18	0.84 ± 0.181
36	0.766 ± 0.207	0.819 ± 0.191	0.838 ± 0.182

In Table 6, we show the test AUC score on MNIST by varying the value of p, meaning that for each row we compute the average AUC score of $p|C|$ clients. Our methodology performs almost as the upper bound baseline, represented by the ideal federations of clients. Nevertheless, the results are consistent with the partitioning we obtain in the first step with the community detection that identifies the right groups of clients in most of the cases. Similar results are available for Fashion-MNIST in [45].

Summarising, in most cases the communities found do match with the ideal groups of clients, which are used as an upper bound baseline in the experimental part. When the ideal groups are not found, our methodology merges more ideal groups into one community. We finally test the resulting AD federated models trained by the detected communities in terms of AUC score. The results show a clear advantage over the models locally trained (i.e., the lower baseline), while the performance is comparable with the federated models of ideal communities' partition, even for detected communities in

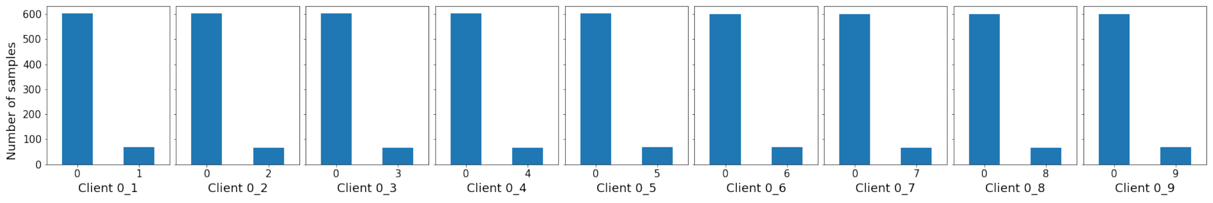


Figure 22: Histograms of training data distribution for the group $C_{in} = 0$ (i.e., 0 is the common inlier class) with $p = 9$. Example of local data distribution between devices. The histograms correspond to the training data distribution for the group $C_{in} = 0$ (i.e., 0 is the common inlier class) with $p = 9$, i.e., the majority (inlier) class is evenly represented among the subset of devices holding it. Conversely, the local outliers are different for each device.

which different majority classes are merged. This indicates that, even though we may not always be able to group clients as in the ideal (supervised) case, still the accuracy of the resulting model is close to optimal and significantly better than using local models trained only on local data.

6.5 Conclusions

In this Section, we presented an overview of the results in [45]. Therein, we defined a new methodology for FL in unsupervised settings, particularly useful for dynamic mobile environments without central coordination. We showed how a set of devices holding non-IID data can collaborate to learn (through low-complexity ML algorithms and models) the existence of new data distributions, going beyond their strictly local knowledge represented by their local and unlabelled data. In this work, we considered a rather simplified data landscape, where the normal data (i.e., inliers) belong to a single data distribution (although different from device to device). In the future, we plan to relax this assumption on the composition of the local datasets by considering that each device holds a multitude of inlier distributions, as it might happen in real-world scenarios.

7 Federated feature selection for efficient data compression and communication

In this section, we present an overview of the paper [53], Appendix D.

7.1 Context and State-of-the-art

In many edge scenarios, edge devices cannot fully process all the data they produce in order to extract knowledge through the training Machine Learning models. To overcome this issue, a typical approach is to transmit data to the closest Edge Server (ES) to process it. However, transferring large quantities of data, especially from resource-limited and power-constrained Edge Devices (ED), might be an issue from the communication standpoint because wireless communications are typically power-hungry and, with high probability, considering a specific learning task, not all the raw data collected is equally informative. In such a case, compressing data before transmission is paramount to save resources and avoid flooding the network with highly redundant data. Beyond the traditional data compression methods, which are not always beneficial from the ML process point of view, a suitable approach would be locally extracting a set of potentially interesting features from raw data directly at the source and using a Feature Selection algorithm (FS) as a tailored compression step before transmission. However, due to the locality of the collection process performed by each EDs, the locally selected features might represent a partial subset of those that actually characterising the informative content. Therefore, in order to keep the information content consistent among all devices collecting data and transmit, FS should be a collaborative process in order to exploit and combine the whole information contained in all the local datasets.

In this activity, we proposed a federated feature selection method where the edge devices collaborate to come up with the minimal set of features selected from their local datasets before transmitting them to the edge server. Our proposal was, to the best of our knowledge, the first of its kind. In fact, although there are works proposing solutions for performing distributed FS, none of them considered federated settings. In [54], authors present a distributed algorithm for FS based on the Intermediate Representation, which aims at preserving the privacy of data, allowing the node to exchange each other the data they hold. Therefore, in this method, FS is performed under the assumption that all data are available to the FS algorithm. Moreover, the method depends on the specific learning model that uses the selected features.

In [55], the authors propose an information-theoretic Federated Feature Selection (FFS) approach called Fed-FiS. Fed-FiS estimates feature-feature mutual information and feature-class mutual information to generate a local feature subset in each user device. Then, a central server ranks each feature and generates a global dominant feature subset using a classification approach. This approach has some commonalities with ours, such as the adopted metric (MI) and the federated settings. However, differently from [55] *(i)* we provide directly the minimum set of relevant features instead of a ranking, *(ii)* we propose an aggregation based on Bayes' theorem that does not rely on any ML scheme to finalise the selection (i.e., no regression or classification methods are adopted in our solution), resulting in a computationally more suitable approach for vehicular scenarios.

7.2 Description of the method

Our method is made of two components, i) an FS algorithm that processes the local data on the devices and ii) an aggregation algorithm executed on the edge server that combines the local estimates transmitted by the devices. Note that the proposed approach shares only the estimates of the most informative local features. Moreover, it guarantees that all the devices reach a consensus on the subset of the most informative features after a finite number of communication rounds between them and the ES.

The FS algorithm is based on the Mutual Information metric [56, 57], and it is designed using the Cross-Entropy method [58]. The formal definition of the FS problems is the following:

Definition 1 (FS Problem) *Given the input data matrix* $\mathbf{X}$ *composed by* n *samples of* m *features (*$\mathbf{X} \in \mathbb{R}^{n\times m}$*), and the target attributes' (or labels) vector* $\mathbf{y} \in \mathbb{R}^n$*, the FS problem is to find a* k*-dimensional subset* $\mathbf{U} \subseteq \mathbf{X}$ *with* $k \leq m$*, by which we can characterise* $\mathbf{y}$.

The method we adopt in the paper [53] performs the FS measuring, through the Mutual Information metric, the amount of information that a subset of features (or attributes) $\mathbf{U}$ expresses regarding a specific target label $\mathbf{y}$. Formally, the MI between random variables can be defined as [59, 60]:

$$\mathbf{I}(\mathbf{U};\mathbf{y}) = \mathbf{H}(\mathbf{y}) - \mathbf{H}(\mathbf{y}|\mathbf{U}), \tag{3}$$

where $\mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_k \mid k \leq m\} \subseteq \mathbf{X}$, and $\mathbf{H}(\mathbf{y}|\mathbf{U})$ is the conditional entropy which measures the amount of information needed to describe $\mathbf{y}$, conditioned by the information carried by $\mathbf{U}$.

In MI-based FS the features to be selected are those that maximise Equation (3) by finding a solution to the following optimisation problem:

$$\begin{gathered}\arg\max_{\mathbf{U}} \mathbf{I}(\mathbf{U};\mathbf{y}) \\ \mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_k \mid k \leq m\} \subseteq \mathbf{X}\end{gathered} \tag{4}$$

Note that the problem (4) belongs to the class of Integer Programming (IP) optimisation problems and finding its optimal solution is NP-hard [61], i.e., the optimal solution $\mathbf{U}$ would be found among all combinations of feature indices of the native set $\mathbf{X}$. The problem (4) becomes computationally tractable if approached through an iterative algorithm which selects and adds to the subset $\mathbf{U}$ one feature at a time. Therefore, instead of solving 4, we address the problem defined in (5):

$$\begin{gathered}\arg\max_{\mathbf{x}_j \in \mathbf{X}\setminus\mathbf{U}} \mathbf{I}(\mathbf{x}_j;\mathbf{y}|\mathbf{U}), \\ \mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_{k-1} \mid k \leq m\} \subseteq \mathbf{X}.\end{gathered} \tag{5}$$

In practice, we solve it through an algorithm based on the Cross-Entropy concept. The CE-based algorithm finds, in a finite number of steps, a solution that well approximates the one found by solving the problem (4), while making negligible the assumption of independence among features introduced in problem (5). In other words, with CE-based FS, instead of selecting one feature at a time, we select a set of features *jointly*. Precisely, we associate each i-th feature with a random variable $z_i \sim \mathrm{Bernoulli}(p_i)$. The

CE-based algorithm identifies which variables z_i, $i = 1, \cdots, m$ must have $p_i \to 1$, so that the objective function $\mathcal{O}(\mathbf{U}(\mathbf{z})) = \mathbf{H}(\mathbf{y}|\mathbf{U})$ gets close to 0. At the end of the local feature selection, each device obtains a vector of the same size of the feature vector in which each element contains the probability that a feature has to be selected. All the mathematical details and algorithms are detailed in the published paper.

The aggregation algorithm is based on a Bayesian approach through which we merge the information sent by the devices to the ES. In our approach, the devices share probability vectors where each element is the estimated probability of selecting a certain feature. The main idea is to merge the local probability vectors through a weighted average where the weights (computed as in Eq. (7)) serve the twofold purpose of *(i)* considering more (or less) those vectors that are computed from larger local datasets and *(ii)* defining common support among all the probability vectors. This second aspect is crucial for the computation's consistency in Eq. (6).

Formally, we assume that each node acquires several independent identically distributed (*i.i.d.*) records n_l to perform the FS, and that the nodes share the same set of features $\mathbf{X}$. The global probability $\mathbf{p}_G$ used for the FS can be written as follows:

$$\mathbf{p}_G = \sum_l \mathbf{p}_l \omega_l, \tag{6}$$

where $\mathbf{p}_l$ is the solution of the local Feature Selection at node l obtained using the Cross-Entropy algorithm, and ω_l weights $\mathbf{p}_l$ *w.r.t.* the other nodes, whose formal definition is:

$$\omega_l = \frac{n_l}{\sum n_l}. \tag{7}$$

we weigh the probability vector of node l proportionally to the size of its local dataset compared to the whole amount of data present in the system.

The proposed algorithm provably converges to a subset of features that effectively compresses the data collected by the devices.

7.3 Relevance for MARVEL

This work explores the possibility of training a data compressor in federated settings. Although developed for vehicular scenarios, such a methodology might also be beneficial in the context targeted by MARVEL when sending data towards the E2F2C infrastructure for processing. Edge devices can collaborate to determine the best set of features to be transmitted to one of the layers mentioned above to avoid generating redundant traffic during data collection. As such, this method directly contributes to the fulfillment of the KPI iKPI-1.1, detailed in Section 1.6.

7.4 Evaluation

We consider a system where a set of autonomous vehicles equipped with edge devices collect data generated by their sensors and collaborate to learn a minimal and most informative set of features from their local datasets. To this end, they execute an in-network data filtering process through our Federated Feature Selection approach to reach a consensus in identifying the most informative feature subset. Finally, the globally shared feature set is used like a compression scheme before transmitting it to ES.

Note that, in this system, EDs are only responsible for finding the best compression scheme applicable to their local data in a collaborative way, based only on the control information they exchange with the ES.

We tested our solution on two benchmark datasets from the literature corresponding to two reference scenarios in the vehicular environment. The first one, called MAV[18] is a publicly available dataset containing both 64×64 images and 6 inertial measurements collected by a drone during a mission in a controlled environment. This dataset refers to a problem of self-localisation in the environment. MAV dataset has been preprocessed as follows. We pre-process the raw images to extract more informative features. In our settings, we extract the HOG features[19], and we assume that the feature extraction is accomplished directly on the drone, which might be possible if equipped with a board of the kind discussed in [62]. The original dataset is unlabeled. Therefore, we labelled it in a way compatible with the positioning context. We associated with each record a label corresponding to the corresponding voxel.[20]

The second dataset, called WEarable Stress and Affect Detection (WESAD), is a collection of data sampled from heterogeneous biophysical sensors: ECG, EDA, EMG, Temperature, Respiration and Inertial Measurements on the three axes. It regards the physiological-state monitoring of a passenger in the automotive domain. The dataset contains readings from two devices, i.e., Respiban and Empatica E4, positioned i) on the chest and ii) on the wrist of human subjects. Each device is equipped with multiple sensors monitoring several physiological parameters. Since the two devices have different operating settings, we focused on the Respiban, whose collection rate is homogeneous for all its sensors. The dataset contains readings collected from 17 human subjects who perform a predetermined protocol to induce the body in one of the following states: 0-baseline, 1-amusement, 2-stress, 3-meditation, and 4-recovery. The data collected for each subject amounts to ∼3.6M records, equivalent to ∼220 MB. A complete description of the dataset is provided in [63].

We evaluate the performance of our solution in terms of (i) compression (i.e., the ratio of features transmitted over the total number of features present in the local dataset), (ii) quality of the selected features (i.e., the accuracy obtained by a model trained on the feature selection obtained by each methodology evaluated) and, (iii) network overhead (i.e., how much additional traffic our solution generates before transmitting the compressed data).

We focus now on the analysis of our FFS method. We compare its performance to those obtained by the Cross-Entropy algorithm executed in centralised settings (CE-CFS).

Table 7 reveals that for MAV dataset, our solution finds a set of features that, although slightly larger than that found by the centralised competitor CE-CFS (24 instead of 18), has the very same informative content, i.e., the accuracy of the NN model trained on both subsets of features are statistically equivalent. The same results hold in the WESAD case too.

We tested the performance of our solution, also varying the size of the local datasets. We aim to understand how it reacts when we decrease the data per device. The results reported in Table 8 show that decreasing the size of data processed at each round does

[18]dataset available at: `https://tinyurl.com/mavmr01`

[19]HOG is a standard feature extraction methodology used in computer vision and image processing to create an image descriptor that captures the spatial relations between different portions of it [62].

[20]A voxel represents a value on a regular grid in three-dimensional space.

Table 7: Comparison between CE-CFS and FFS on MAV and WESAD.

Dataset	Method	Size (# obs.)	FS (#)	C (%)	Accuracy (%±C.I.)
MAV	CE-CFS	2911	18	99	96.7±0.5
	FFS	291	24	99	96.7±0.4
WESAD	CE-CFS	15M	4	50	94.6±0.8
	FFS	3M	4	50	94.6±0.8

not significantly affect the number of communication rounds needed for convergence. Interestingly, the same holds also for the WESAD scenario.

Table 8: Performance of FFS varying the data processed during a communication round. The columns report the size of data used for each update (Size), the number of selected features (FS), the accuracy, the number of communication rounds upon convergence (R_c), the compression obtainable with FFS (C), the network overhead generated by FFS (N_{OH}), and the size of the cache needed to collect the data before starting the data transmission.

Dataset	Size (# obs)	FS (#)	Accuracy (%±C.I.)	R_c (#)	C (%)	N_{OH} (MB)	Cache (MB)
MAV	291	24	96.7±0.4	44	99	16	217
	203	[illegible]	96.5±0.3	37	99	13	128
	145	34	97.0±0.3	55	98	20	134
	87	59	[illegible]±0.3	43	97	15	63
	[illegible]	[illegible]	97.2±0.4	53	98	19	26
WESAD	$3\cdot10^6$	4	93.6±0.8	11	50	0.009	$2\cdot10^3$
	$1\cdot10^6$	5	93.8±0.5	10	38	0.008	$1\cdot10^3$

7.5 Conclusions

In this section, we presented an overview of the results in paper [53]. The results show that our FFS algorithm identifies a minimal subset of informative features without sharing raw data between the devices. FFS is robust to feature redundancy and all the devices can reach a consensus on the FS achieving a compression rate of up to 90x on the selected datasets. Finally, the quality of the feature selection is maintained. The proposed framework is general and modular, i.e., it can be applied to every incremental FS algorithm that associates a probability to each feature. We plan to investigate how to turn it into a framework to include more FS algorithms, enriching the challenges connected to the data distribution on devices (i.e., non-IID-ness) and the potential lack of supervision.

8 Nonlinear stochastic gradient descent

In this section, we briefly describe the results in [64], Appendix E, where MARVEL UNS researchers are three work's co-authors.

8.1 Context and State-of-the-art

For training large-scale machine learning models, stochastic gradient descent (SGD) and its advanced versions have been extensively used, e.g., [65, 66, 67, 68, 69, 70, 71].

More recently, several nonlinear variants of SGD have been introduced and are shown to exhibit a number of favourable properties. More precisely, clipped gradient descent, e.g., [72, 73], the sign gradient, e.g., [74, 75], and (component-wise) quantised gradient, e.g., [76, 77], have been introduced. Introducing a nonlinearity has been shown to speed up models training [73], and improve communication efficiency [75, 74].

On the other hand, several recent studies demonstrate that, when training deep learning models, gradient noise exhibits a heavy-tailed behaviour [78, 79]. A recent reference [80] introduces adaptive clipping nonlinearity into SGD to combat the heavy-tailed gradient noise.

8.2 Description of the method

In paper [64], we have introduced a general framework of nonlinear SGD to combat a heavy-tailed gradient noise. An algorithm within the framework, that minimises a convex function f, works as follows:

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \alpha_t \mathbf{\Psi}(\nabla f(\mathbf{x}^t) + \boldsymbol{\nu}^t). \tag{8}$$

Here, $\mathbf{x}^t$ is the solution estimate at iteration t, $t = 0, 1, ...$; $\mathbf{\Psi} : \mathbb{R}^d \mapsto \mathbb{R}^d$ is a general nonlinear map; $\alpha_t > 0$ is step-size; $\boldsymbol{\nu}^t \in \mathbb{R}^d$ is a zero-mean gradient noise; and $\mathbf{x}^0$ is an arbitrary deterministic initialisation.

The framework allows for very general nonlinearities that include the following popular and commonly used choices: Sign gradient; Component-wise clipping; Component-wise quantisation; Normalised gradient; and Clipped gradient.

The contributions of [64] can be summarised as follows. The paper establishes for the above-defined general nonlinear SGD strong convergence guarantees, namely almost sure convergence, asymptotic normality, and mean-squared error (MSE) convergence rate. When compared with state-of-the-art, the paper's results correspond to a more general class of nonlinearities; existing studies usually focus on a specific nonlinearity such as clipping. In addition, the paper allows for more general gradient noise distributions, specifically allowing for infinite variance gradient noises.

8.3 Application and relevance to MARVEL

Several recent works, e.g., [78, 79], have observed existence of fat-tailed distributions of gradient updates within the training process for some widely used DL models. To improve the speed and therefore communication efficiency of federated training, nonlinearities of the type studied here can be exploited, together with similar analytical tools for convergence/performance analysis, contributing to the KPIs iKPI-1.1 and iKPI-12.1, Section 1.6.

8.4 Evaluation

We present validation of the results in [64], by demonstrating numerically that the nonlinear SGD, with the normalised gradient nonlinearity, converges in the presence of an infinite-variance gradient noise. At the same time, the standard, linear SGD fails to converge under the same setting.

The simulation setting is as follows. A strongly convex quadratic function is considered, $f : \mathbb{R}^d \mapsto \mathbb{R}$, $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b}^\top \mathbf{x}$. Here, $\mathbf{A} \in \mathbb{R}^{d\times d}$ is a symmetric positive definite matrix generated at random, with $d = 16$. Also, vector $\mathbf{b}$ is generated randomly. We consider the method in [64] with the normalised gradient nonlinearity and the linear SGD. The gradient noise has the distribution with the following probability density function (pdf) entry-wise:

$$p(u) = \frac{\alpha - 1}{2(1+|u|)^\alpha}, \tag{9}$$

for $u \in \mathbb{R}$ and $\alpha = 2.05$. The gradient noise is independent of the current iterate $\mathbf{x}^t$. The above noise distribution can be shown to have an infinite variance. We use with both methods the zero initialisation and step-size $\alpha_t = \frac{1}{t+1}$. Figure 23, left, shows a Monte Carlo estimate of MSE across iterations t. We can see that the proposed method (blue) converges in heavy-tailed noise, while the linear SGD (red) fails to converge. Figure 23, right, repeats the experiment, but now at each iteration t, gradient noise is dependent on $\mathbf{x}^t$; more precisely, we have that each entry of the gradient noise at iteration t follows the distribution in (9), with $\alpha = \alpha_t = 2 + \|x^t\|$, where $\|\cdot\|$ denotes the Euclidean norm. We can observe a similar behaviour as the one shown in Figure 23, left.

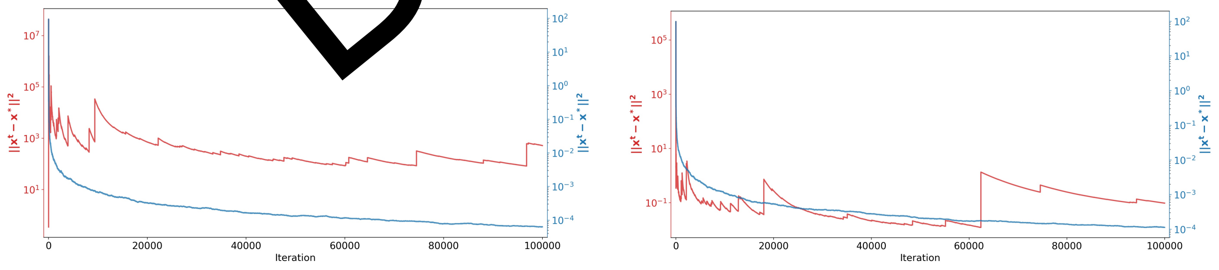


Figure 23: Monte Carlo estimate of MSE versus iterations t for the normalised gradient-based nonlinear SGD (blue) and linear SGD (red). Left: noise is independent of the current iterate; Right: noise is dependent on the current iterate.

8.5 Conclusions

The current work focused on investigating the effects of gradient update nonlinearities for training a model in a centralised fashion. However, the adopted methodology and accompanying analytical tools are of direct interest to federated training as well. Future work in that sense could investigate different nonlinearity points – such as client nonlinearities, server nonlinearity or a combined approach. A problem of particular interest is to identify/optimise the nonlinearity type to be adopted for a given noise distribution of the client's gradients.

9 Analytical benchmarking of stochastic gradient descent based on large deviations rates

In this section, we briefly describe the results in [81], Appendix F, where two co-authors are MARVEL UNS researchers.

9.1 Context and State-of-the-art

Stochastic gradient descent (SGD) is a well-established method for optimising a given objective function, such as an empirical loss that arises with ML/DL model training, or as a subtask of a more complex optimisation or learning method, such as federated learning. SGD finds numerous practical applications, such as training machine learning and deep learning models, e.g., [65, 66, 82]. In order to evaluate SGD performance and compare different alternatives – in terms of the final solution of the "plain" SGD (e.g., last iterate versus suffix averaging) or within stochastic training protocols (e.g., sampling strategy for participating training nodes), typically MSE is the metric of choice. Recent works have shown that high probability metrics with SGD exhibit informativeness and in some cases advantage over the commonly adopted MSE-based ones, e.g., [83, 84, 85, 86, 87].

9.2 Description of the method

We provide here an overview of the results presented in the paper [81]. Reference [81] is interested in applying the large deviations theory to analysing the stochastic gradient descent (SGD) method. More precisely, we consider unconstrained optimisation problems where the goal is to minimize a smooth, strongly convex function $f : \mathbb{R}^d \to \mathbb{R}$, via the SGD method of the form

$$X_{k+1} = X_k - \alpha_k \left(\nabla f(X_k) - Z_k\right). \tag{10}$$

Here, $k = 1, 2, ...$ is the iteration counter, $\alpha_k = a/k$, $a > 0$ is the step size, and Z_k is a zero-mean gradient noise that may depend on X_k. In this context, reference [81] finds for SGD the set-valued function $I(\cdot)$, referred to as the inaccuracy rate, such that the following holds:

$$\mathbb{P}\left(X_k \in C\right) = e^{-k\,\mathbf{I}(C)+o(k)}. \tag{11}$$

Here, $o(k)$ corresponds to terms growing slower than linearly with k, and C is an arbitrary open subset of $\mathbb{R}^d$. Moreover, the reference expressed the inaccuracy rate $\mathbf{I}(C)$ in the form of the so-called *rate function* $I : \mathbb{R}^d \mapsto \mathbb{R}$, according to the following formula [88]:

$$\mathbf{I}(C) = \inf_{x \in C} I(x). \tag{12}$$

Solving for (11) is highly relevant to the analysis of SGD, as it provides estimates of the probabilities of "rare events." For example, when $C = \{x : \|x - x^\star\| > \epsilon\}$, where $x^\star$ is the minimiser of f, (11) quantifies the probability that the solution estimate X_k stays ϵ-away from the solution $x^\star$.

Existing literature only provides non-tight lower bounds on the inaccuracy rate $\mathbf{I}(C)$, for the case when set C is a complement of an Euclidean ball. In contrast, reference [81] provides inaccuracy rate $\mathbf{I}(C)$ for arbitrary open sets C. Furthermore, it provides *exact* asymptotic expressions for $\mathbf{I}(C)$ which are therefore *tight*, for the special case when the objective function f is quadratic.

9.3 Application and relevance to federated learning

As it has been demonstrated in [81], the large deviations metric in (11) can be more informative in several scenarios than the commonly used MSE criterion. Specifically, as shown in [81], the performance of SGD depends not only on the noise variance but also on higher order noise moments. In other words, the SGD subject to a one type of noise (e.g., Gaussian) can perform very differently than the SGD subject to another type of noise (e.g., Laplacian), even when in the two cases, the noise variances are mutually equal. That is, higher-order moments make the difference in the SGD performance, in this case. This higher-order difference is however not captured via an MSE analysis that only "sees" the noise variance. In contrast, the large deviations inaccuracy rate metric in (11) successfully captures this difference in the SGD performance. Therefore, quantification of the inaccuracy rate metric opens a new avenue for optimisation of stochastic algorithms like SGD. Similarly, an inaccuracy rate analysis for stochastic FL methods like, e.g., FedAvg with partial participation can potentially lead to improved FL design, such as, e.g., design of client sampling strategies, step sizes, client transmit power optimisations, etc.. This is not pursued per se here but is subject to future work, and it may be considered in the context of MARVEL use cases implementation tasks T5.4 and T6.2. Due to its intrinsic capability to handle various sources of complexity in distributed and federated systems (non-IID gradient noise, system intermittency and heterogeneity), the tool directly contributes to the iKPI-1.1, Section 1.6.

9.4 Evaluation

We provide a numerical study highlighting the relevance of the large deviations metric in (11). A strongly convex quadratic cost function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is considered, with $f(x) = \frac{1}{2}x^\top Ax + bx$, $d = 10$. Further, $A \in \mathbb{R}^{d \times d}$ is a symmetric matrix and $b \in \mathbb{R}^d$ is a vector generated at random; further details can be found in [81].

The gradient noise is generated in an independent, identically distributed way. We consider two different noise distributions (per gradient noise element), the zero-mean Gaussian and the zero-mean Laplace, while the per-element gradient variance is equal in the two cases, and it equals σ^2. This scenario enables us to assess the effects of higher order moments on the performance of SGD.

Figure 24 plots a numerical, Monte Carlo estimate, of the rare event probability $P\left(\|X_k - x^\star\| > \epsilon\right)$ along iterations $k = 1, 2, \ldots$, for $\epsilon = 0.3$. We can see that, even though for the two cases the gradient noise variance is the same, the performance of SGD for the two scenarios is very different. This is captured well by the inaccuracy rate metric in (11) and is not "visible" via an MSE analysis, as explained above.

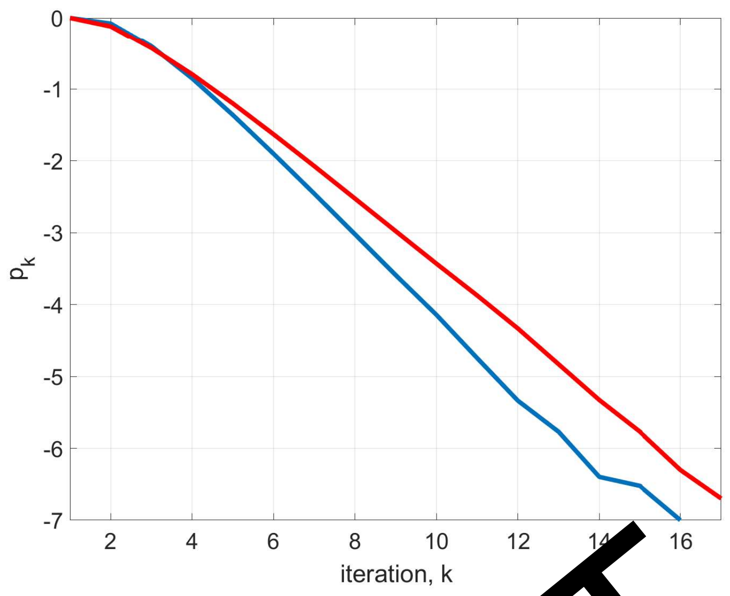


Figure 24: Numerical estimate of the rare event probability $\mathbb{P}(\|X_k - x^\star\| > \epsilon)$ along iterations $k = 1, 2, \ldots$ for SGD with Gaussian (blue line) and Laplacian (red line) noise.

9.5 Conclusions

We provided an overview of the results presented in the paper [81]. We have established in the paper large deviations performance of the stochastic gradient descent method, and we outlined how the newly established large deviations metrics may be used in the design and analysis of FL algorithms.

10 Distributed statistical inference and learning: Performance evaluation

In this section, we briefly describe the results of paper [89], Appendix G, the author of which is a MARVEL UNS researcher.

10.1 Context and State-of-the-art

Recently, there has been significant progress in distributed inference and learning. Well-known methods include consensus+innovations, e.g., [90, 91], diffusion, e.g., [92, 93], and non-Bayesian or social learning, e.g., [94, 95].

Unlike conventional server-client FL, a common setup for these methods assumes a network of nodes (agents) interconnected over a generic network. Each agent holds a local inference vector (e.g., decision variable, belief). Each agent updates their inference vectors at each time by i) assimilating a new observation and ii) weight-averaging their own and their neighbours inference vectors. See [89] for more details on the corresponding mathematical formulation.

There have been a number of works concerned with performance evaluation of distributed learning and inference methods. The authors of [95], [96] consider the problem of distributed M-ary hypothesis testing. Therein, agents perform their local updates by applying a Bayesian update on the vector of prior beliefs.

There have been a few works that are concerned with *inaccuracy rates* of distributed inference and learning methods. We refer to Chapter 9 for the definition and relevance of the large deviations (inaccuracy rate) metric. Large deviations analysis has been performed in [95], assuming a static directed network. The authors show that the large deviation performance is related to the eigenvector-centralities convex combination of the nodes' local rate functions.

To the best of our knowledge, large deviation performance for distributed inference (inaccuracy rates) for random networks have not been sufficiently studied before. The corresponding large deviations rates (detection error exponents) have been studied on random networks [90, 97], but only for a different problem, namely distributed detection.

10.2 Description of the method

We briefly present here the method and results for large deviations analysis (inaccuracy rates calculation) for distributed inference and learning developed in paper [89].

Namely, in contrast with the works in [92]-[98], reference [89] addresses computation of the inaccuracy rate for distributed inference on *random networks*. For each node in the network, it provides a lower and a family of upper bounds on the rate function (inaccuracy rate). This is achieved by carrying out node-specific large deviations analyses. We show that the two bounds match in several cases, such as for pendant nodes and also for nodes in a regular network, hence providing the exact (tight) inaccuracy rate evaluation. Interestingly, the rate function is a convex envelope between the rate function of the full network and the rate function of the respective component, lifted up by the probability of the event that induces the component. This confirms the intuition that the distributed inference behaves in its performance "in-between" 1) a centralised inference engine that has access to all nodes' data; and 2) each node working in isolation. However, it in addition quantifies, for each targeted regime, the benefits of nodes' cooperation (or the loss with respect to the centralised system).

As an application of particular interest to this study, we consider social learning, specifically the form with the geometric average update [95]. We show that, with appropriate transformation of the belief iterates – namely, considering their log-ratios with respect to the belief in the true distribution, the algorithm studied in [95] exhibits full equivalence to the consensus+innovations algorithm that we analyse here. Building on this equivalence, we characterise the rate function of the beliefs in social learning and provide the first proof of the large deviations principle for social learning run over random networks.

In addition, the results of [89] can be applied to distributed detection as well. In that case, the inference vector becomes a scalar local decision statistic at each node, that is being compared with a threshold. In that case, finding, for example, the false alarm probability corresponds to the calculation of the rate function for a specific set, equal to a one-sided interval in the real line.

10.3 Relevance for MARVEL

Distributed inference and learning, e.g., of consensus+innovations type, represents a useful alternative to server-client FL settings (see Section 4), for the scenarios of low nodes-to-server bandwidths and when network capacities between nodes are highly distributed and smaller than that in a data-center [99]. In these scenarios, a server-client training algorithm like FedAvg may be replaced with a consensus+innovations type method. Therein, the inaccuracy rate analysis is a valuable tool for system design, e.g., uplink communication protocols, node activation schedule, etc., contributing in particular to the iKPI [illegible].1, Section 1.6.

10.4 Evaluation

We illustrate the relevance of the rate function metric on a distributed detection problem, where we consider the distributed detection algorithm in [90]. We consider the same "baseline" distributed detection algorithm under two different scenarios. In the first scenario, nodes utilise a randomised communication, such that the communication probabilities increase over iterations up to a certain limiting value p. In the second scenario, the communication probabilities at each node are kept constant along iterations, equal to p. By the large deviations theory, the two different scenarios exhibit the same large deviations (rate function) performance asymptotically, as the number of iterations grows. Therefore, in terms of the transmit power, intuitively, the scheme with increasing communications should perform better as far as the overall communication cost (number of transmissions) is concerned.

Figure 25 plots the Monte Carlo-estimated probability of detection error versus the total number of transmissions (communication cost). We can see that the scenario with increasing probabilities of communications leads to an improved detection performance, as predicted by the theory.

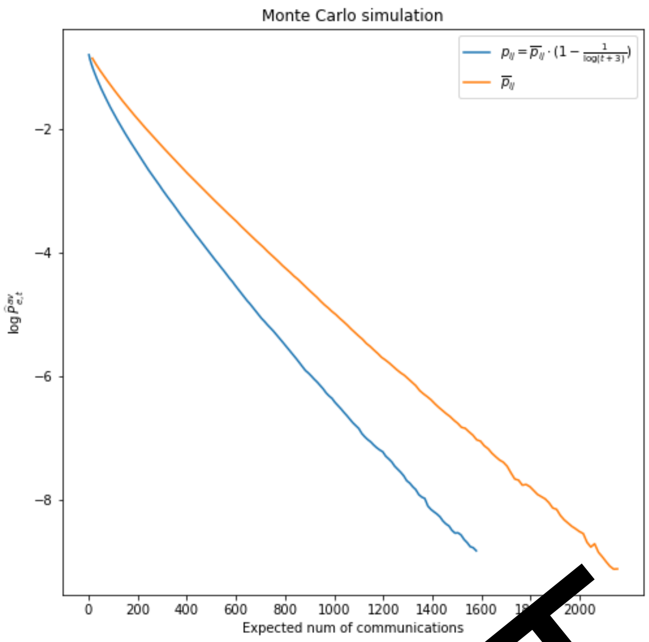


Figure 25: Estimated probability of error (in the log scale) with constant activation probabilities (orange), time-varying activation probabilities (blue), versus the expected number of communications.

10.5 Conclusions

We provided an overview of [89] that establishes large deviations (inaccuracy rates) performance of a class of distributed inference and learning algorithms over random networks. We explained the results achieved, as well as their relevance for MARVEL.

11 Conclusions and plans for the 2nd release of the MARVEL integrated framework

In this deliverable, we presented the final research, innovation, and development results achieved in the context of Task 3.2 of the MARVEL project. These results are along several important dimensions. First, the task developed a novel open-source tool for personalised federated learning. Second, it implemented, deployed, and validated the developed tool on several real-world use cases defined in the MARVEL project, including highly relevant deep learning tasks such as crowd counting. Third, it described several strategies for data privacy preservation and enhancements, as well as for edge-fog-cloud deployments of personalised federated learning methods. Finally, the work in the context of this task has resulted in several state-of-the-art advances in the personalised federated and distributed learning domain. The deliverable has described each of those papers, highlighting their positioning and advances beyond state-of-the-art, as well as their relevance to MARVEL and MARVEL's application domain. Future work will include the deployment and integration of the task outputs into the MARVEL R2 period's framework and use cases, as well as consider exploitation opportunities for the task's outputs achieved.

References

[1] Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[2] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[3] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[4] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The NumPy array: a structure for efficient numerical computation. *Computing in science & engineering*, 13(2):22–30, 2011.

[5] Wes McKinney et al. pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, 14(9):1–9, 2011.

[6] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[7] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[8] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, Pedro PB de Gusmão, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

[9] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[10] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[11] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, et al. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021.

[12] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[15] Qingyu Song, Changan Wang, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Jian Wu, and Jiayi Ma. To choose or to fuse? scale selection for crowd counting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2576–2583, 2021.

[16] Di Hu, Lichao Mou, Qingzhong Wang, Junyu Gao, Yuansheng Hua, Dejing Dou, and Xiao Xiang Zhu. Ambient sound helps: Audiovisual crowd counting in extreme conditions. *arXiv preprint arXiv:2005.07097*, 2020.

[17] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE, 2017.

[18] Di Hu, Lichao Mou, Qingzhong Wang, Junyu Gao, Yuansheng Hua, Dejing Dou, and Xiao Xiang Zhu. Ambient sound helps: Audiovisual crowd counting in extreme conditions, 2020.

[19] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[20] Boris Sekachev, Nikita Manovich, Maxim Zhiltsov, Andrey Zhavoronkov, Dmitry Kalinin, Ben Hoff, TOsmanov, Dmitry Kruchinin, Artyom Zankevich, DmitriySidnev, Maksim Markelov, Johannes222, Mathis Chenuet, a andre, telenachos, Aleksandr Melnikov, Jijoong Kim, Liron Ilouz, Nikita Glazov, Priya4607, Rush Tehrani, Seungwon Jeong, Vladimir Skubriev, Sebastian Yonekura, vugia truong, zliang7, lizhming, and Tritin Truong. opencv/cvat: v1.1.0, August 2020.

[21] A. Armacki, D. Bajvovic, D. Jakovetic, and S. Kar. Personalized federated learning via convex clustering. In *2022 IEEE International Smart Cities Conference (ISC2)*, Pafos, Cyprus, 2022.

[22] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[23] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models, 2021.

[24] Alireza Fallah, Aryan Mokhtari, and Asuman E. Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan,

and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[25] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated evaluation of on-device personalization, 2019.

[26] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[27] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.

[28] Ilai Bistritz, Ariana Mann, and Nicholas Bambos. Distributed distillation for on-device learning. In *NeurIPS*, 2020.

[29] Yae Jee Cho, Jianyu Wang, Tarun Chiruvolu, and Gauri Joshi. Personalized federated learning for heterogeneous clients with clustered knowledge transfer. *ArXiv*, abs/2109.08119, 2021.

[30] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning, 2021.

[31] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 32(8):3710–3722, 2021.

[32] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *ArXiv*, https://arxiv.org/abs/2002.10619, 2020.

[33] Filip Hanzely, Slavomir Hanzely, Samuel Horvath, and Peter Richtarik. Lower bounds and optimal algorithms for personalized federated learning. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020.

[34] Huahua Wang, Arindam Banerjee, and Zhi-Quan Luo. Parallel direction method of multipliers. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[35] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[36] Akshay Agrawal, Robin Verschueren, Steven Diamond, and Stephen Boyd. A rewriting system for convex optimization problems. *Journal of Control and Decision*, 5(1):42–60, 2018.

[37] A. Armacki, D. Bajovic, D. Jakovetic, and S. Kar. Personalized federated learning via convex clustering. *arXiv preprint, available at: https://arxiv.org/abs/2202.00718*, 2022.

[38] A. Armacki, D. Bajovic, D. Jakovetic, and S. Kar. One-shot federated learning for model clustering and learning in heterogeneous environments. 2022. arxiv preprint, https://arxiv.org/abs/2209.10866.

[39] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[40] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *IEEE Transactions on Information Theory*, pages 1–1, 2022.

[41] Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. Robust federated learning in a heterogeneous environment. *arXiv preprint arXiv:1906.06629*, 2019.

[42] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 32(8):3710–3722, 2021.

[43] Aleksandar Armacki, Dragana Bajovic, Dusan Jakovetic, and Soummya Kar. Personalized federated learning via convex clustering. *arXiv preprint arXiv:2202.00718*, 2022.

[44] Toby Dylan Hocking, Armand Joulin, Francis Bach, and Jean-Philippe Vert. Clusterpath An Algorithm for Clustering using Convex Fusion Penalties. In *28th international conference on machine learning*, page 1, United States, June 2011.

[45] Mirko Nardi, Lorenzo Valerio, and Andrea Passarella. Anomaly detection through unsupervised federated learning. In IEEE, editor, *Proceedings of the 38th IEEE International Conference on Mobile Sensing and Networking (MSN)*, 2022.

[46] Mirko Nardi, Lorenzo Valerio, and Andrea Passarella. Centralised vs decentralised anomaly detection: when local and imbalanced data are beneficial. In Nuno Moniz, Paula Branco, Luis Torgo, Nathalie Japkowicz, Michał Woźniak, and Shuo Wang, editors, *Proceedings of the Third International Workshop on Learning with Imbalanced Domains: Theory and Applications*, volume 154 of *Proceedings of Machine Learning Research*, pages 7–20. PMLR, 17 Sep 2021.

[47] Bram van Berlo, Aaqib Saeed, and Tanir Ozcelebi. Towards federated unsupervised representation learning. In *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, pages 31–36, 2020.

[48] Fengda Zhang, Kun Kuang, Zhaoyang You, Tao Shen, Jun Xiao, Yin Zhang, Chao Wu, Yueting Zhuang, and Xiaolin Li. Federated unsupervised representation learning. *arXiv preprint arXiv:2010.08982*, 2020.

[49] Efthymios Tzinis, Jonah Casebeer, Zhepei Wang, and Paris Smaragdis. Separate but together: Unsupervised federated learning for speech enhancement from non-iid data. *arXiv preprint arXiv:2105.04727*, 2021.

[50] Viraaji Mothukuri, Prachi Khare, Reza M Parizi, Seyedamin Pouriyeh, Ali Dehghantanha, and Gautam Srivastava. Federated-learning-based anomaly detection for iot security attacks. *IEEE Internet of Things Journal*, 9(4):2545–2554, 2021.

[51] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

[52] B. Scholkopf et al. Support vector method for novelty detection. In *Advances in neural information processing systems*, 1999.

[53] Pietro Cassará, Alberto Gotta, and Lorenzo Valerio. Federated feature selection for cyber-physical systems of systems. *IEEE Transactions on Vehicular Technology*, 71(9):9937–9950, 2022.

[54] X. Ye, H. Li, A. Imakura, and T. Sakurai. Distributed Collaborative Feature Selection Based on Intermediate Representation. In *Int. Conf. IJCAI 2019*, pages 4242–4149, 2019,August.

[55] Sourasekhar Banerjee, Erik Elmroth, and Monowar Bhuyan. Fed-fis: a novel information-theoretic federated feature selection for learning stability. In Teddy Mantoro, Minho Lee, Media Anugerah Ayu, Kok Wai Wong, and Achmad Nizar Hidayanto, editors, *Neural Information Processing*, pages 480–487, Cham, 2021. Springer International Publishing.

[56] G. Brown. A new perspective for information theoretic feature selection. In *Int. Conf. on Artificial Intelligence and Statistics*, Clearwater Beach, Fl, USA, April 2009.

[57] X.V. Nguyen, J. Chan, S. Romano, and J. Bailey. Effective Global Approaches for Mutual Information Based Feature Selection. In *Int. Conf. ACM on Knowledge Discovery and Data Mining*, pages 1–10, New York City, NY, USA, August 2014.

[58] D.P. Rubinstein, R.Y. Kroese. *The Cross-Entropy Method: a Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, Machine Learning*. Springer, 2004.

[59] R.J. McEliece. *In The Theory of Information and Coding: A Mathematical Framework for Communication*, volume Vol. 3 of *Encyclopedia of Mathematics and Its Applications*. Addison-Wesley Publishing Company: Reading, MA, USA, 1977.

[60] J.A. Cover, T.M. Thomas. *Elements of Information Theory*. John Wiley and Sons, Inc, New York, NY, USA, 1991.

[61] A.W. Chaovalitwongse, I.P. Androulakis, and P.M. Pardalos. Quadratic Integer Programming: Complexity and Equivalent Form. *Springer Encyclopedia of Optimization*, pages 3153–3159, September 2009.

[62] P. Chen, C. Huang, C. Lien, and Y. Tsai. An efficient hardware implementation of hog feature extraction for human detection. *IEEE Transactions on Intelligent Transportation Systems*, 15(2):656–662, 2014.

[63] P. Schmidt, A. Reiss, R. Duerichen, C. Marberger, and K. Van Laerhoven. Introducing WESAD, a multimodal dataset for Wearable Stress and Affect Detection. In *Int. Conf. ACM ICMI*, pages 1–9. ACM, 2019, October.

[64] D. Jakovetic, D. Bajovic, A. Kumar Sahu, S. Kar, N. Milosevic, and D. Stamenkovic. Nonlinear gradient mappings and stochastic optimization: A general framework with applications to heavy-tail noise. *SIAM J. Opt.*, 2022. to appear.

[65] Feng Niu, Benjamin Recht, Christopher Ré, and Stephen J Wright. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *arXiv preprint arXiv:1106.5730*, 2011.

[66] Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. A unified theory of sgd: Variance reduction, sampling, quantization and coordinate descent. In *International Conference on Artificial Intelligence and Statistics*, pages 680–690. PMLR, 2020.

[67] Richard H Byrd, Gillian M Chin, Will Neveitt, and Jorge Nocedal. On the use of stochastic hessian information in optimization methods for machine learning. *SIAM Journal on Optimization*, 21(3):977–995, 2011.

[68] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[69] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[70] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczynski. *Lectures on stochastic programming: modeling and theory*. SIAM, 2021.

[71] Volkan Cevher, Stephen Becker, and Mark Schmidt. Convex optimization for big data: Scalable, randomized, and parallel algorithms for big data analytics. *IEEE Signal Processing Magazine*, 31(5):32–43, Sept. 2014.

[72] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318. PMLR, 2013.

[73] Jingzhao Zhang, Tianxing He, Suvrit Sra, and Ali Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. *arXiv preprint arXiv:1905.11881*, 2019.

[74] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

[75] Lukas Balles, Fabian Pedregosa, and Nicolas Le Roux. The geometry of sign gradient descent. *arXiv preprint arXiv:2002.08056*, 2020.

[76] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic. QSGD: Communicationefficient sgd via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pages 1709–1720, 2017.

[77] Samuel Horváth, Dmitry Kovalev, Konstantin Mishchenko, Sebastian Stich, and Peter Richtárik. Stochastic distributed learning with gradient quantization and variance reduction. *arXiv preprint arXiv:1904.05115*, 2019.

[78] Umut Simsekli, Mert Gürbüzbalaban, Thanh Huy Nguyen, Gaël Richard, and Levent Sagun. On the heavy-tailed theory of stochastic gradient descent for deep neural networks. *arXiv preprint arXiv:1912.00018*, 2019.

[79] Mert Gurbuzbalaban, Umut Simsekli, and Lingjiong Zhu. The heavy-tail phenomenon in sgd. In *International Conference on Machine Learning*, pages 3964–3975. PMLR, 2021.

[80] Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank J Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? *arXiv preprint arXiv:1912.03194*, 2019.

[81] D. Bajovic, D. Jakovetic, and S. Kar. Large deviations rates for stochastic gradient descent with strongly convex functions. 2022. available at: https://arxiv.org/abs/2211.00969.

[82] Lihua Lei and Michael I Jordan. On the adaptivity of stochastic gradient-based optimization. *SIAM Journal on Optimization*, 30(2):1473–1500, 2020.

[83] S. Ghadimi and G. Lan. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization I: A generic algorithmic framework. *SIAM J. Optim.*, 22(4):1469–1492, 2012.

[84] S. Ghadimi and G. Lan. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization, II: Shrinking procedures and optimal algorithms. *SIAM J. Optim.*, 23(4):2061–2089, 2013.

[85] A. Juditsky, A. Nazin, A. Nemirovsky, and A. Tsybakov. Algorithms of robust stochastic optimization based on mirror descent method. *arXiv:1907.02707*, 2019.

[86] Eduard Gorbunov, Marina Danilova, and Alexander Gasnikov. Stochastic optimization with heavy-tailed noise via accelerated gradient clipping. *arXiv preprint arXiv:2005.10785*, 2020.

[87] Damek Davis, Dmitriy Drusvyatskiy, Lin Xiao, and Junyu Zhang. From low probability to high confidence in stochastic convex optimization. *J. Mach. Learn. Res.*, 22:49–1, 2021.

[88] R. R. Bahadur. On the asymptotic efficiency of tests and estimates. *Sankhya: The Indian Journal of Statistics, 1933-1960*, 22(3/4):229–252, 1960.

[89] D. Bajovic. Inaccuracy rates for distributed inference over random networks with applications to social learning. 2022. available at: https://arxiv.org/abs/2208.05236.

[90] D. Bajović, D. Jakovetić, J. Xavier, B. Sinopoli, and J. M. F. Moura. Distributed detection via Gaussian running consensus: Large deviations asymptotic analysis. *IEEE Transactions on Signal Processing*, 59(9):4381–4396, 9 2011.

[91] D. Bajović, J. M. F. Moura, J. Xavier, and B. Sinopoli. Distributed inference over directed networks: Performance limits and optimal design. *IEEE Transactions on Signal Processing*, 64(13):3308–3323, 7 2016.

[92] V. Matta, P. Braca, S. Marano, and A. H. Sayed. Diffusion-based adaptive distributed detection: Steady-state performance in the slow adaptation regime. *IEEE Trans. Information Theory*, 62(8):4710–4732, 8 2016.

[93] S. Marano and A. H. Sayed. Detection under one-bit messaging over adaptive networks. *IEEE Trans. Information Theory*, 65(10):6519–6538, 10 2019.

[94] Ali Jadbabaie, Pooya Molavi, Alvaro Sandroni, and Alireza Tahbaz-Salehi. Non-Bayesian social learning. *Games and Economic Behavior*, 76(1):210 – 225, 2012.

[95] A. Lalitha, T. Javidi, and A. D. Sarwate. Social learning and distributed hypothesis testing. *IEEE Transactions on Information Theory*, 64(9):6161–6179, 2018.

[96] S. Shahrampour, A. Rakhlin, and A. Jadbabaie. Distributed detection: Finite-time analysis and impact of network topology. *IEEE Transactions on Automatic Control*, 61(11):3256–3268, 2016.

[97] D. Bajović, D. Jakovetić, J. M. F. Moura, J. Xavier, and B. Sinopoli. Large deviations performance of consensus+innovations distributed detection with non-Gaussian observations. *IEEE Transactions on Signal Processing*, 60(11):5987–6002, 11 2012.

[98] Aritra Mitra, John A. Richards, and Shreyas Sundaram. A new approach to distributed hypothesis testing and non-bayesian learning: Improved learning rate and byzantine resilience. *IEEE Transactions on Automatic Control*, 66(9):4084–4100, 2021.

[99] Chenghao Hu, Jingyan Jiang, and Zhi Wang. Decentralized federated learning: A segmented gossip approach. 2022. available at: https://arxiv.org/abs/1908.07782.

Appendices

A Personalized Federated Learning via Convex Clustering

The appended paper follows.

Personalized Federated Learning via Convex Clustering

Aleksandar Armacki[1], Dragana Bajovic[2], Dusan Jakovetic[3], and Soummya Kar[1]

[1]Carnegie Mellon University, Pittsburgh, PA
Email: {aarmacki, soummyak}@andrew.cmu.edu

[2]Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia
Email: dbajovic@uns.ac.rs

[3]Faculty of Sciences, University of Novi Sad, Novi Sad, Serbia
Email: dusan.jakovetic@dmi.uns.ac.rs

***Abstract*—We propose a parametric family of algorithms for personalized federated learning with locally convex user costs. The proposed framework is based on a generalization of convex clustering in which the differences between different users' models are penalized via a sum-of-norms penalty, weighted by a penalty parameter λ. The proposed approach enables "automatic" model clustering, without prior knowledge of the hidden cluster structure, nor the number of clusters. Analytical bounds on the weight parameter, that lead to simultaneous personalization, generalization and automatic model clustering are provided. The solution to the formulated problem enables personalization, by providing different models across different clusters, and generalization, by providing models different than the per-user models computed in isolation. We then provide an efficient algorithm based on the Parallel Direction Method of Multipliers (PDMM) to solve the proposed formulation in a federated server-users setting. Numerical experiments corroborate our findings. As an interesting byproduct, our results provide several generalizations to convex clustering.**

I. Introduction

Federated learning (FL) is a paradigm in which many users collaborate, with the goal of learning a joint model [1]. Each user has a local dataset, with private and possibly sensitive data. The data distribution across users is typically highly heterogeneous.

A federated learning system can have a huge amount of users, wherein each user contributes with a proportionally small local dataset. Therefore, the federation provides users with the benefit of training on the joint data, effectively offering broader knowledge and better generalization. However, due to the highly heterogeneous nature of the data, it is nontrivial to design a FL system where individual users achieve better performance though the federation when compared with models trained on their own local data. In fact, the authors in [2] show that in many tasks, users may actually not benefit from participating in federated learning. The globally trained model underperforms on their local data, compared to the model solely trained on the local data. Moreover, applying privacy preserving techniques further deteriorates the performance. On the other hand, users with very small datasets suffer from overfitting and poor generalization of models trained only on their local data.

To remedy these problems and reap the benefits of both worlds - the abundance of data and better learning that the federation offers, as well as adapting the models to perform well on the local data, *personalized federated learning* is introduced. Unlike the standard federated learning, the goal of personalized federated learning is to learn multiple models. In particular, let N be the number of participating users, with $f : \mathbb{R}^d \mapsto \mathbb{R}$ a given cost function. Then, the goal of standard FL is to solve

$$\arg\min_{x \in \mathbb{R}^d} F_{\text{global}}(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x), \tag{1}$$

where $f_i(x)$ is the cost function f evaluated on the local dataset of the i-th user. Contrary to this approach, the (broad) goal of personalized FL is to learn N models, by solving

$$\arg\min_{x_1,\ldots,x_N \in \mathbb{R}^d} F_{\text{local}}(x_1,\ldots,x_N) = \frac{1}{N}\sum_{i=1}^{N} f_i(x_i), \tag{2}$$

subject to appropriately defined constraints. Depending on the constraints imposed and the approach taken to solving (2), the literature on personalized FL adopts different approaches to personalization, including multi-task learning [3], [4], fine-tuning [5], [6], knowledge distillation [7], [8], [9], [10] and clustering-based approaches [11], [12], [13].

In this paper, we propose a novel approach to personalized federated learning that enables simultaneous personalization, generalization, and model clustering. The approach is based on the following novel personalized FL (convex problem) formulation:

$$\arg\min_{x_1,\ldots,x_N \in \mathbb{R}^d} F(x_1,\ldots,x_N) = \frac{1}{N}\sum_{i=1}^{N} f_i(x_i) + \lambda \sum_{j \neq i} \|x_i - x_j\|, \tag{3}$$

The work of D. Bajovic is supported by the European Union's Horizon 2020 Research and Innovation program under grant agreement No 957337. The work of A. Armacki and S. Kar was partially supported by the National Science Foundation under grant CNS-1837607. This paper reflects only the authors' views and the European Commission cannot be held responsible for any use which may be made of the information contained therein.

where $\lambda > 0$ is a penalty parameter, and $\|\cdot\|$ stands for the Euclidean norm. Compared with (2), the formulation (3) has a regularization term controlled with $\lambda > 0$, that penalizes the differences between the local nodes' solutions via a sum-of-norms penalty.

Formulation (3) may be seen as a generalization of convex clustering, e.g., [14], where, instead of the (quadratic) distance functions, which penalize the departure from a local data point, we use arbitrary convex local losses.

Problem (3) is also related with the personalized FL formulation in [15], which, instead of the sum of norms of pairwise distances uses the sum of *squared* norms of distances of local solutions to their average. As we show in the paper, although the two formulations are resembling, the solution structures of the two formulations are qualitatively very different.

Specifically, as we show in the paper, the solution to (3) has several interesting properties, which, to the best of our knowledge, are not (jointly) exhibited with any of the previous personalized FL formulations. Namely, a solution to (3) has a clustered structure: depending on the penalty parameter λ and similarity of the local functions, local solutions $x_i^\star$ of (3) are equal across certain users' groups (clusters). The number of groups (clusters) K is automatically determined as part of the solution. To further illustrate benefits of this feature, suppose that the users exhibit an (unknown) clustered structure such that each user's data within a cluster comes from the same distribution, while the distributions that correspond to different clusters are mutually different. If the different clusters' distributions are sufficiently far apart, the proposed method (3) uncovers the unknown cluster structure and hence allocates the same models to all users within the same cluster. This allows within-cluster generalization, i.e., users to effectively enlarge their training data by harnessing data of all users from the same cluster. In addition, depending on the distance between the different clusters' distributions, the method (3) allows for a controlled across-clusters generalization; intuitively, it allows a user to harness data from another cluster's distribution, but with a different (reduced) "weight" when compared to within-cluster data. If, at an extreme, the difference between different clusters' distributions is negligible (but this information is unknown), then users should clearly use the global model (1). This is again captured by (3), because (as shown in the paper) it matches (1) for λ above a threshold.

In summary, our contributions are the following: First, we propose a novel formulation for personalized federated learning (3), the solution of which has a clustering structure while at the same time preserving generalization abilities.

Second, we provide a condition on the penalty parameter λ, with theoretical guarantees, for discovering the "hidden structure" underlying the models; this condition is expressed in terms of the well-established diversity of the local functions, hence making a strong connection and justifying analytically the use of this quantity.

Third, the proposed solution "automatically" determines the number of models K, i.e., K need not be known in advance.

Fourth, we provide an efficient algorithm to solve the novel personalized learning formulation (3) in a federated server-client setting that is based upon the Parallel Direction Method of Multipliers (PDMM) [16].

Finally, we demonstrate by simulation examples, on a supervized binary classification problem, that the proposed solution exhibits 1) generalization, i.e., improves testing accuracy with respect to the users models trained in isolation; 2) personalization, i.e., improves testing accuracy with respect to the global FL model (1); and 3) achieves a comparable (or better) generalization and personalization (in the sense of 1) and 2)) than [4], while at the same time uncovering cluster structure, hence reducing the number of distinct models from N to K.

With respect to existing personalized FL approaches discussed above, the works [11], [12], [13], [10] also account for users' clustering in a certain way, but very differently from our approach. Most notably, existing approaches aim to uncover "cluster identities" first and subsequently provide loss minimizations across cluster groups in isolation from other groups. This within-clusters isolation may reduce overall generalization ability of the models. In contrast, the proposed approach allows also for across-clusters generalization that is further controlled by the penalty parameter λ. It is worth noting that reference [17] introduces formulation similar to (3), but in a different context of distributed consensus optimization. Most importantly, they are only concerned with the question when (3) matches (1), i.e., when (3) leads to a global consensus across local models; they are not concerned, nor they study personalization (clustering) abilities of (3).

Our results are also of direct interest to convex clustering, e.g., [14], as they provide recovery guarantees for generalized convex clustering, when the squared quadratic loss $f_i(x) := \|x_i - a_i\|^2$ per data point $a_i \in \mathbb{R}^d$ is replaced with an arbitrary differentiable convex loss, e.g., the Huber loss.

Paper organization. The rest of the paper is organized as follows. Section II describes the problem of interest and outlines the assumptions used in the analysis. Section III presents the recovery guarantees of the method. Section IV outlines an efficient algorithm for solving the proposed problem in the federated setting. Section V presents numerical experiments, and Section VI concludes the paper. The next paragraph introduces the notation used throughout the paper.

Notation. The set of real numbers is denoted by $\mathbb{R}$, while $\mathbb{R}^d$ denotes the corresponding d-dimensional vector space; $\|\cdot\|$ represents the standard Euclidean norm. $\langle \cdot, \cdot \rangle$ represents the standard vector product over the space of real vectors. $[N]$ denotes the set of integers up to and including N, i.e., $[N] = \{1, \ldots, N\}$.

II. Problem formulation

Consider a collection of N users, $i = 1, \ldots, N$, that participate in a federated learning activity. Each user i holds a function $f_i : \mathbb{R}^d \to \mathbb{R}$. Function f_i may correspond, e.g., to an empirical loss with respect to the local data set available at user i. We make the following assumptions throughout the paper.

Assumption 1. *For each $i = 1, \dots, N$, function $f_i : \mathbb{R}^d \to \mathbb{R}$ is convex and coercive, i.e., $f_i(x) \to +\infty$ whenever $\|x\| \to +\infty$.*

Assumption 2. *For each $i = 1, \dots, N$, function $f_i : \mathbb{R}^d \to \mathbb{R}$ has Lipschitz continuous gradients, i.e. the following holds*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \le L\|x - y\|, \text{ for all } x, y \in \mathbb{R}^d.$$

Note that, under the above assumptions, problems (1) and (3) are solvable. We denote by $y^\star \in \mathbb{R}^d$ a solution to (1) and by $\{x_i^\star\}$, $i = 1, \dots, N$, $x_i^\star \in \mathbb{R}^d$, a solution to (3).

There are many machine learning models that satisfy Assumptions 1 and 2, such as supervized binary classification problems studied in Section V.

The high-level goal in personalized federated learning is that each user i finds a local model, say $x_i^\bullet \in \mathbb{R}^d$, that performs well on the local data (i.e., the value $f_i(x_i^\bullet)$ is low), but that also exhibits a generalization ability with respect to data available at other users $j \neq i$. In addition, a desirable feature of personalized federated learning is that the users are able to classify other users into two categories. The first category corresponds to those users $j \in \{1, 2, \dots, N\}$ that have similar data (similar f_j's) to their own; the second category corresponds to those users whose local data is "sufficiently different" from theirs. With this classification in place, each user i can fully harness the data from "similar users" for an improved personalization while avoiding overfitting; e.g., when user i has a very few data points of its own, it effectively enlarges its data set while preserving personalization. On the other hand, the data from "sufficiently different users" should still be harnessed in a controlled way to further improve generalization abilities.

To account for the effects above, we provide a novel personalized learning formulation, where each user i wants to obtain the local model $x_i^\star \in \mathbb{R}^d$ such that $x^\star := ((x_1^\star)^\top, \dots, (x_N^\star)^\top)^\top \in \mathbb{R}^{N\,d}$ is a minimizer of (3), where $\lambda > 0$ is a tuning parameter. Intuitively, the term $\sum_{i=1}^N f_i(x_i)$ forces the local models x_i's to behave well with respect to local costs f_i's; the term $\lambda \sum_{j \neq i} \|x_i - x_j\|$ makes the local models be mutually close, hence enabling generalization. The penalization term $\sum_{j \neq i} \|x_i - x_j\|$ is known to enforce sparsity in other contexts, in the sense that it forces many of the x_i's to be mutually equal at a solution of (3).

It is interesting to compare our novel formulation (3) with the personalized federated learning formulation in [4]:

$$\operatorname*{arg\,min}_{x_1, \dots, x_N \in \mathbb{R}^d} \frac{1}{N} \sum_{i=1}^N f_i(x_i) + \gamma \sum_{j \neq i} \|x_i - x_j\|^2, \qquad (4)$$

where $\gamma > 0$ is a penalty parameter.

The difference of (4) with respect to (3) is that, in (4), the differences of local models x_i and x_j are penalized via the squared Euclidean norm, while with our formulation, the 2-norm appears without squares. There are several important implications of this difference with respect to the resulting personalized learning models. Most importantly, in contrast with (3), formulation (4) in general does not lead to model clustering for any $\lambda > 0$. In addition, as a side comment, the solutions to (1) and (4) are in general mutually different for any $\lambda > 0$ (in the sense that the solution $\{y_i^\star\}$ to (4) does not obey $y_i^\star = y_j^\star$, for all i, j, irrespective of the choice of λ). In contrast, with (3), we recover global model learning as in (1) for $\lambda > \widehat{\lambda}$.

We also connect (3) with convex clustering. Convex clustering, e.g. [14], is an appealing method to cluster N data points $a_i \in \mathbb{R}^d$, $i = 1, \dots, N$. The method corresponds to solving problem (3) with $f_i(x) = \frac{1}{2}\|x - a_i\|^2$. Intuitively, to each data point i, we associate a candidate cluster center x_i, and then we enforce a (soft) constraint that many x_i's should be mutually equal. There are several efficient algorithms and cluster recovery guarantees results available for convex clustering, but only when $f_i(x) = \frac{1}{2}\|x - a_i\|^2$. Our results make a direct generalization of convex clustering to other loss metrics, e.g., the "distance" of a candidate cluster x_i from data point a_i may be measured through the Huber loss.

III. Theoretical guarantees for optimal cluster recovery

In this section, we state and prove our main results on characterization of solutions to (3).

We start by defining the following auxiliary optimization problem associated to a certain (predefined) partition of users $C_1, \dots, C_K$, $\cup_{k=1}^K C_k = [N]$ and $C_k \cap C_j = \emptyset$:

$$\operatorname*{arg\,min}_{w_1, \dots, w_N \in \mathbb{R}^d} \frac{1}{N} \sum_{k=1}^K n_k g_k(w_k) + \lambda \sum_{l \neq k} n_k n_l \|w_k - w_l\|, \quad (5)$$

where $g_k(w) := 1/n_k \sum_{i \in C_k} f_i(w)$, for $w \in \mathbb{R}^d$, and $n_k = |C_k|$ is the number of elements in C_k, for $k = 1, \dots, K$. Let $w_k^\star = w_k^\star(\lambda)$, $k = 1, \dots, K$, denote a solution to (5). Note that problem (5) is solvable by Assumption 1.

Theorem 1. *Consider problem* (3)*. Assume that, for some node partition $C_1, C_2, \dots, C_K$, and parameter λ, there holds*

$$\lambda \ge \max_{k=1,\dots,K} \max_{i,j \in C_k} \frac{\|\nabla f_i(w_k^\star) - \nabla f_j(w_k^\star)\|}{n_k}. \qquad (6)$$

Next, let $\{x_i^\star\}$, $i = 1, \dots, N$, be defined as follows: for each $i \in C_k$, we let $x_i^\star = w_k^\star$, for $k = 1, \dots, K$, where $\{w_k^\star = w_k^\star(\lambda)\}$, $k = 1, \dots, K$, is a solution to (5)*, defined for the same partition $C_1, \dots, C_K$ that verifies* (6)*. Then, $\{x_i^\star\}$, $i = 1, \dots, N$, is a solution of* (3)*.*

Remark 1. *Note that Theorem 1 guarantees that at least one solution of* (3) *exhibits the clustered structure with respect to partition $C_1, \dots, C_K$, while it does not preclude a scenario that there might be another solution of* (3) *that may not exhibit this cluster structure. However, when each of the f_i's is in addition assumed to be strictly convex, then $\{x_i^\star\}$ is unique, and it necessarily has the clustered structure.*

We next prove Theorem 1.

This can be easily seen based on Theorem 1 in [17].

Proof. The proof is in spirit similar to Theorem 1 in [18]. From the first order optimality conditions for (5), we obtain that, for each $k = 1, \ldots, K$, there must hold:

$$\nabla g_k(w_k^\star) + \lambda \sum_{l \neq k} n_l r_{kl}^\star = 0, \tag{7}$$

where $r_{kl}^\star$ is a subgradient of $\|w_k - w_l\|$ with respect to w_k, computed at the solution. For each $k, l = 1, \ldots, K$, $r_{kl}^\star$ must satisfy:

$$r_{kl}^\star = \begin{cases} \frac{w_k^\star - w_l^\star}{\|w_k^\star - w_l^\star\|}, & \text{for}\, w_k^\star \neq w_l^\star \\ \text{a vector}\, r \in \mathbb{R}^d\, \text{s.t.}\, \|r\| \leq 1, & \text{otherwise} \end{cases} \tag{8}$$

We now turn to first order optimality conditions for the original problem (3):

$$\nabla f_i(x_i) + \lambda \sum_{j \neq i} s_{ij} = 0, \tag{9}$$

where s_{ij} is a subgradient of $\|x_i - x_j\|$ computed with respect to x_i. Similarly as in the above, at the solution, s_{ij} must satisfy:

$$s_{ij} = \begin{cases} \frac{x_i - x_j}{\|x_i - x_j\|}, & \text{for}\, x_i \neq x_j \\ \text{a vector}\, s \in \mathbb{R}^d\, \text{s.t.}\, \|s\| \leq 1, & \text{otherwise} \end{cases} \tag{10}$$

It can be verified that, when condition (6) is fulfilled, then the following choice of $x_i^\star$ and $s_{ij}^\star$ satisfy the first order optimality conditions in (9) and (10)

$$x_i^\star = w_k^\star,\, i \in C_k, \tag{11}$$

$$s_{ij}^\star = \begin{cases} r_{kl}^\star, & j \in C_l, k \neq l \\ \frac{\nabla f_j(w_k^\star) - \nabla f_i(w_k^\star)}{\lambda n_k}, & j \in C_k \end{cases}, \tag{12}$$

hence proving the result. □

Theorem 1 guarantees the existence of a solution of (3) that exhibits the desired clustering structure. However, if the parameter λ is chosen too large, it can actually coarsen the clustering structure $C_1, \ldots, C_K$ and provide a solution with $1 \leq M < K$ groups (clusters). The following theorem ensure the correct clustering structure is recovered.

Theorem 2. *Consider problem* (5). *If for some node partition $C_1, C_2, \ldots, C_K$ and parameter $\lambda > 0$ there holds*

$$\lambda < \frac{\min_{k,l \in [K], k \neq l} \|\nabla g_k(w_k^\star) - \nabla g_l(w_k^\star)\|}{2 \max_{k \in [K]} \sum_{l \neq k} n_l}, \tag{13}$$

then for each $k, l \in [K], k \neq l$, we have $w_k^\star \neq w_l^\star$, where $w_k^\star = w_k^\star(\lambda)$, $k = 1, \ldots, K$, is a solution to (5), *defined for the same partition $C_1, \ldots, C_K$, that verifies* (13).

We note that in practice, the bounds (6) and (13) might not be easy to obtain, as $w_k^\star$'s depend on λ. In Appendix A we provide several considerations regarding selection of the penalty parameter λ in (3), in practice.

We next prove Theorem 2.

Proof. Denote by $s_k = \lambda \sum_{l \neq k} r_{kl}^*$. From (7), we have

$$\begin{aligned} \|s_l - s_k\| &= \|\nabla g_k(w_k^*) - \nabla g_l(w_l^*)\| \\ &\leq \|\nabla g_k(w_k^*) - g_l(w_k^*)\| + \|\nabla g_l(w_k^*) - \nabla g_l(w_l^*)\| \\ &\leq \|\nabla g_k(w_k^*) - g_l(w_k^*)\| + L\|w_k^* - w_l^*\|, \end{aligned}$$

where we used Assumption 2 in the second inequality. Rearranging, we get

$$\|w_k^* - w_l^*\| \geq \frac{1}{L}\|\nabla g_k(w_k^*) - g_l(w_k^*)\| - \frac{1}{L}(\|s_l\| + \|s_k\|).$$

Next, note that

$$\|s_k\| \leq \lambda \sum_{l \neq k} n_l \|r_{kl}^*\| \leq \lambda \sum_{l \neq k} n_l.$$

Plugging in the equation above, we get

$$\|w_k^* - w_l^*\| \geq \frac{1}{L}\|\nabla g_k(w_k^*) - g_l(w_k^*)\| - \frac{2\lambda}{L} \max_{k \in [K]} \sum_{l \neq k} n_l.$$

It can be readily checked that the choice of λ satisfying (13) results in

$$\|w_k^* - w_l^*\| > 0, \quad k \neq l,$$

hence showing the claim. □

IV. Algorithm for Personalized Federated Learning

In this section, we introduce an algorithm to solve (3) in a federated server-users setting. The algorithm is adapted from the parallel direction method of multipliers (PDMM) in [16].

We start by reformulating problem (3) as follows:

$$\begin{aligned} \min \sum_{i=1}^{N} f_i(x_i) + \lambda \sum_{j \neq i} \|z_{ij}\| \\ \text{s.t. } x_i - x_j = z_{ij},\ i \neq j. \end{aligned} \tag{14}$$

That is, each of the $N(N-1)$ terms $\|x_i - x_j\|$ in (3) are replaced with $\|z_{ij}\|$, where $z_{ij} \in \mathbb{R}^d$ is an auxiliary (primal) variable. Then, for equivalence of (3) and (14), we add for each ordered pair (i, j), $i \neq j$, the constraint $x_i - x_j = z_{ij}$. Next, introduce the augmented Lagrangian $L : \mathbb{R}^{N\, d} \times \mathbb{R}^{N(N-1)d} \times \mathbb{R}^{N(N-1)d} \to \mathbb{R}$, defined by:

$$\begin{aligned} L(\{x_i\}, \{z_{ij}\}, \{\mu_{ij}\}) &= \sum_{i=1}^{N} f_i(x_i) + \lambda \sum_{j \neq i} \|z_{ij}\| \\ + \quad \sum_{j \neq i} \mu_{ij}^\top (x_i - x_j - z_{ij}) &+ \frac{\rho}{2} \sum_{j \neq i} \|x_i - x_j - z_{ij}\|^2, \end{aligned} \tag{15}$$

where $\{x_i\}$, $i = 1, \ldots, N$, and $\{z_{ij}\}$, $i = 1, \ldots, N$, $j = 1, \ldots, N$, $i \neq j$, are the primal variables, and $\{\mu_{ij}\}$, $i = 1, \ldots, N$, $j = 1, \ldots, N$, $i \neq j$, are the dual variables, and $\rho > 0$ is a penalty parameter. Abstracting details, PDMM proceeds as follows. First, it updates at each iteration $t = 0, 1, \ldots$, a randomly selected subset of primal variables $\{x_i, z_{ij}\}$ by

It is possible to halve the number of constraints in (14) by imposing the constraint $x_i - x_j = z_{ij}$ only for $i < j$. This approach reduces the number of variables at the cost of additional coordination of users on the server's side. We present here the approach with the larger number of variables and less coordination required.

minimizing a surrogate of L with the rest of primal and dual variables fixed. Then, a randomly selected subset of dual variables $\{\mu_{ij}\}$ is updated, while also bookkeeping a set of auxiliary dual variables $\{\widehat{\mu}_{ij}\}$. See equations (29)–(31) in [19] for a detailed definition of the generic PDMM.

Here, we apply and adapt PDMM to solve (14), and hence, solve (3), in the federated server-users setting. To facilitate presentation of the algorithm, we enumerate all primal variables x_i's and z_{ij}'s through a common index set $\mathcal{S}_P$ with N^2 elements, such that the i-th element of $\mathcal{S}_P$, $i = 1, \ldots, N$, corresponds to x_i, and the remaining $N(N-1)$ subsequent elements correspond to z_{ij}'s, where the ordered pairs $\ell \sim (i,j)$ are positioned lexicographically in $\mathcal{S}_P$. For example, $(N+1)$-th element of $\mathcal{S}_P$ corresponds to variable z_{12}, $(N+2)$-nd element of $\mathcal{S}_P$ corresponds to z_{13}, etc. Similarly, we let $\mathcal{S}_D$ be the $N(N-1)$-sized index set, such that its ℓ-th element corresponds to the dual variable μ_{ij}, $\ell \sim (i,j)$, $\ell = 1, \ldots, N(N-1)$. The PDMM-based personalized FL method is shown in Algorithm 1.

Functions $B_i(\cdot,\cdot)$ in (16) and $B_{ij}(\cdot,\cdot)$ in (17) are instances of Bregman divergence, e.g., [16]; for example, they can be taken as $B(u,v) = \frac{1}{2}\|u-v\|^2$. The choice of functions $B_i(\cdot,\cdot)$ and $B_{ij}(\cdot,\cdot)$ also affect the computational cost of updates (16) and (17), respectively. For example, for $B_{ij}(u,v) = \frac{1}{2}\|u-v\|^2$, update (17) corresponds to evaluating a proximal operator of the 2-norm that is done via block soft-thresholding. See also subsection 2.1 in [16] for the choices of $B_i(\cdot,\cdot)$ that make update (16) computationally cheap. The positive parameters η_i in (16) and η_{ij} in (17) weigh the Bregman divergence terms; the larger η_i is, the closer $x_i^{(t+1)}$ is to $x_i^{(t)}$, i.e., the smaller steps the algorithm makes. A similar effect is achieved with η_{ij} in (17). Quantity $\widehat{\mu}_{ij}^{(t)}$ in (19) is an auxiliary dual variable associated with the dual variable $\mu_{ij}^{(t)}$. The update step (19) is a backward dual step that is introduced for improving the algorithm's stability that may otherwise be violated due to the parallel and randomized nature of primal variable updates; see [16] for details. Similarly, parameters $\tau > 0$ and $\nu > 0$ in (18) and (19), respectively, are "damping" factors in dual variable updates, that are again used to stabilize the algorithm trajectory.

With Algorithm 1's initialization, we can set the $z_{ij}^{(0)}$'s, $\mu_{ij}^{(0)}$'s, and and $\widehat{\mu}_{ij}^{(0)}$'s arbitrarily. For example, they can be all set to zero. For the initialization of the x_i's, we need that $x_j^{(0)}$, $j \neq i$, is available at each user i. This can be achieved by, e.g., setting $x_i^{(0)} = 0$, for all i, or by letting the server send a common initial point $y^{(0)} \in \mathbb{R}^d$ to all users prior to the algorithm start, so that each user i sets $x_i^{(0)} = y^{(0)}$. The initial point $y^{(0)}$ may also be obtained by (approximately) solving (1) via a (non-personalized, standard) FL algorithm, e.g., FedAvg.

With Algorithm 1, the server maintains and updates the $N(N-1)$ $d \times 1$-sized dual variables $\mu_{ij}^{(t+1)}$, $i,j = 1, \ldots, N$, $i \neq j$. Each user i maintains and updates the $d \times 1$-sized primal variable $x_i^{(t)}$; $N-1$ $d \times 1$-sized primal variables $z_{ij}^{(t)}$, $j \neq i$; and $N-1$ $d \times 1$-sized auxiliary dual variables $\widehat{\mu}_{ij}^{(t)}$, $j \neq i$.

With Algorithm 1, communication from users to the server ("uplink") takes place at steps (S5) and (S6). Note that, at

Algorithm 1 A PDMM-based algorithm for personalized FL and model clustering

```
for $t = 0, \ldots, T-1$ do
    (S1) The server randomly selects a subset $\mathcal{S}_P^{(t)} \subset \mathcal{S}_P$ of $S_P$, $S_P < N^2$, primal variables;
    (S2) Each user $i \in \{1, \ldots, N\}$, such that $i \in \mathcal{S}_P^{(t)}$, performs the update of $x_i^{(t)}$ as follows:
        $x_i^{(t+1)} = \arg\min_{x_i \in \mathbb{R}^d} f_i(x_i) + \sum_{j \neq i} (\widehat{\mu}_{ij}^{(t)})^\top x_i + \frac{\rho}{2} \sum_{j \neq i} \|x_i - x_j^{(t)} - z_{ij}^{(t)}\|^2 + \eta_i B_i(x_i, x_i^{(t)});$   (16)
    (S3) Each user $i \in \{1, \ldots, N\}$, such that $\ell \in \mathcal{S}_P^{(t)}$, $\ell \sim (i,j)$, performs the update of $z_{ij}^{(t)}$ as follows:
        $z_{ij}^{(t+1)} = \arg\min_{z_{ij} \in \mathbb{R}^d} \lambda \|z_{ij}\| - (\widehat{\mu}_{ij}^{(t)})^\top z_{ij} + \frac{\rho}{2} \|x_i^{(t)} - x_j^{(t)} - z_{ij}\|^2 + \eta_{ij} B_{ij}(z_{ij}, z_{ij}^{(t)});$   (17)
    (S4) For each $s \in \{1, \ldots, N\}$, and $\ell$, $\ell \sim (i,j)$, such that $s, \ell \notin \mathcal{S}_P^{(t)}$, set $x_s^{(t+1)} = x_s^{(t)}$, and $z_{ij}^{(t+1)} = z_{ij}^{(t)}$;
    (S5) Each user $i \in \{1, \ldots, N\}$, such that $i \in \mathcal{S}_P^{(t)}$, sends $x_i^{(t+1)}$ to the server;
    (S6) Each user $i \in \{1, \ldots, N\}$, such that $\ell \in \mathcal{S}_P^{(t)}$, $\ell \sim (i,j)$, sends $z_{ij}^{(t+1)}$ to the server;
    (S7) The server collects $\{x_i^{(t+1)}\}$, $i \in \mathcal{S}_P^{(t)}$, and broadcasts this $(S_P\, d) \times 1$ vector to all users $i \notin \mathcal{S}_P^{(t)}$;
    (S8) The server picks a random subset $\mathcal{S}_D^{(t)}$, $\mathcal{S}_D^{(t)} \subset \mathcal{S}_D$, of $S_D$ dual variables, and performs the following update for $\ell \in \mathcal{S}_D$, $\ell \sim (i,j)$:
        $\mu_{ij}^{(t+1)} = \mu_{ij}^{(t)} + \tau\, \rho \left( x_i^{(t+1)} - x_j^{(t+1)} - z_{ij}^{(t+1)} \right);$   (18)
    (S9) The server sets $\mu_{ij}^{(t+1)} = \mu_{ij}^{(t)}$, for $\ell \notin \mathcal{S}_D^{(t)}$, $\ell \sim (i,j)$;
    (S10) For each $\ell \in \mathcal{S}_D^{(t)}$, $\ell \sim (i,j)$, the server sends $\mu_{ij}^{(t+1)}$ to user $i$;
    (S11) Each user $i$, such that $(i,j) \sim \ell$, $\ell \in \mathcal{S}_D^{(t)}$, performs the following update:
        $\widehat{\mu}_{ij}^{(t+1)} = \mu_{ij}^{(t+1)} - \nu\, \rho \left( x_i^{(t+1)} - x_j^{(t+1)} - z_{ij}^{(t+1)} \right);$   (19)
end for
```

each t, during steps (S5) and (S6), the server receives exactly S_P $d\times 1$-sized (real vectors) messages. Here, S_P is the design parameter that can be taken to be much smaller than N, hence the uplink communication does not incur high overhead. Communication from the server to users ("downlink") takes place at steps (S7) and (S10). At step (S7), the server brodcasts S_D $d\times 1$-sized messages. At step (S10), the server transmits a total of S_D $d \times 1$-sized messages to different users. (Specifically, variable $\mu_{ij}^{(t+1)}$ is sent to user i, for $\ell \sim (i,j)$, $\ell \in \mathcal{S}_D^{(t)}$.) Therefore, the downlink communication involves a total of $(S_P + S_D)$ $d \times 1$-sized messages per iteration t. Quantity S_D is also a design parameter that can be set to be much smaller than N; hence, the downlink communication does not incur a significant communication overhead.

By applying Theorem 3 in [16] it can be shown that, under appropriately chosen tuning parameters, $\mathbb{E}\left[F(x^{(t)}) - F^\star\right] = O(1/t)$, where we recall that F is the objective function defined in (3), $F^\star = \inf_x F(x)$, and $x^{(t)} = ((x_1^{(t)})^\top, \ldots, (x_N^{(t)})^\top)^\top$ is generated by Algorithm 1.

V. Numerical results

We now present numerical simulations. In the first set of experiments, we evaluate the cluster recovery abilities, as well as personalization and generalization abilities of the proposed formulation (3) and compare it with alternatives.

We now describe the first set of experiments. We consider a supervized binary classification problem. The generated data contains $K = 3$ clusters. For each cluster, for each given label/class (± 1), the data comes from a uniform distribution over an ellipse in $\mathbb{R}^2$. The two ellipses that correspond to different classes for a given cluster overlap, so that the data in each cluster is not linearly separable. We generate 200 (training) data points from the distributions from each cluster, 100 points per class. Then, we associate to each cluster 20 FL users. Each FL user samples 10 data points out of the 200 data points available in its cluster. Hence, the data for all users within a cluster comes from the same distribution. Figure 1 illustrates the data, where different colors corresponds to different clusters, while the dashed lines represent optimal separators for each cluster, computed using the squared Hinge loss, i.e. the separators that minimize the squared Hinge loss over the full training data, for each cluster. The squared Hinge loss is used throughout the simulations, as the local loss function of each user i, and is given by

$$f_i(x) = \frac{c\|w\|^2}{2} + \frac{1}{m}\sum_{j=1}^{m} \max\left\{0, 1 - \ell_{ij}(\langle w, a_{ij}\rangle - b)\right\}^2,$$

where m represents the number of (local) data points, $(a_{ij}, \ell_{ij})_{j=1}^m$, represent the data points and class labels at user i, $c > 0$ is a penalty parameter that controls the regularization, while $x = \left[w, b\right] \in \mathbb{R}^{d+1}$, $d = 2$, represents the vector that defines the classifier. That is, the classifier based on vector $x = \left[w, b\right]$ takes a feature vector $a \in \mathbb{R}^2$ as input and predicts its label as $\ell = \mathrm{sign}(\langle w, a\rangle - b)$. Throughout the experiments, we set the parameter $c = 10^{-3}$, in order to put more weight on the classification performance of the method.

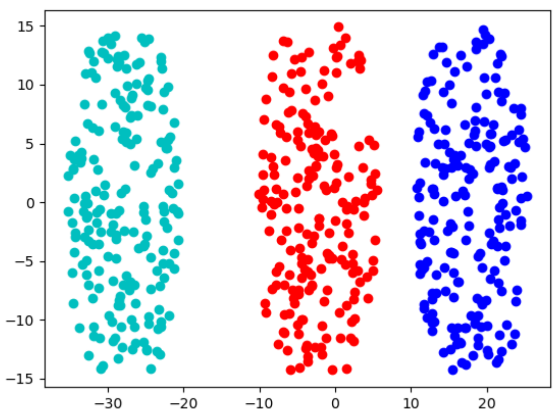


Fig. 1: Training data, generated by the process described in Section V. The dashed lines represent optimal separators of the ellipses, within a cluster.

We compare the proposed formulation (3) with the alternatives in (1), (2) and (4), referred to here as the global model, local model and squared penalty, respectively. Formulation (1) corresponds to a standard, non-personalized FL solution. That is, the classifier vectors x_i's with (1) are equal for all users, i.e., $x_i = y^*$, where we recall that y^* is the solution to (1). With formulation (2), each user i's classifier vector equals $x_i = y_i^* = \arg\min f_i(x)$. In addition, we compare the proposed formulation (3) with an oracle model that knows beforehand the clustering structure of the users; then, for each user i within a cluster C_k, the oracle lets user i's classifier vector be $x_i = \arg\min_x \sum_{j \in C_k} f_k(x)$ We expect that the oracle model performs best in the considered setup among all methods, as it has an unfair advantage of knowing the cluster structure beforehand, and the data distributions of different clusters are very different, so data from a different cluster confuses another cluster's classifier. To evaluate solutions (1)-(4), we used CVXPY [20], [21].

In order to evaluate generalization and personalization abilities of different methods, we evaluate testing accuracy of the corresponding classifiers with respect to a newly generated test data. More precisely, let x_i be a classifier vector for user i obtained through training via any of the methods (1)-(4). For each user i, we then evaluate the testing accuracy of the classifier x_i with respect to the full testing data set for the cluster to which user i belongs to. We then average the testing accuracy across all users $i = 1, \ldots, N$. For each cluster, the testing data is generated by drawing new samples (new with respect to training data) from the same distributions according to which the training data is generated. Methods (3) or (4) then exhibit generalization if the average testing accuracy is above the average testing accuracy of local models; they exhibit personalization if their average testing accuracy is above that of the global model. The performance of the models, for different values of λ, is presented in Figure 2. Additionally, we present the average Euclidean distance between the classifier vectors x_i's belonging to the same cluster. The results are summarized in Figure 3.

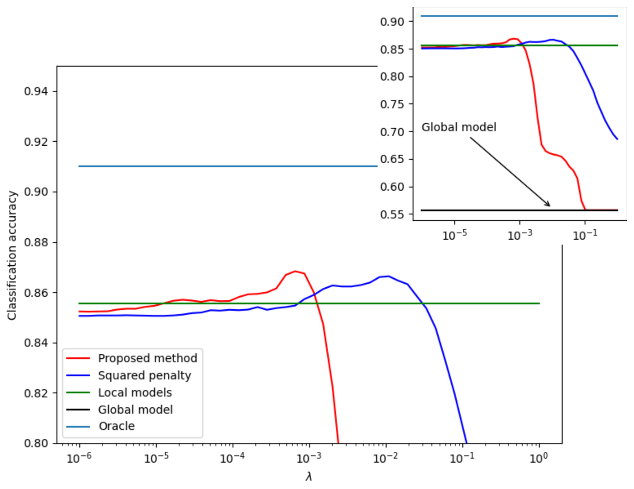


Fig. 2: The average classification accuracy across all users and all clusters. We can see that the two personalization methods achieve both personalization, as they outperform the global model, as well as generalization, as they outperform the strictly local model, for certain values of λ.

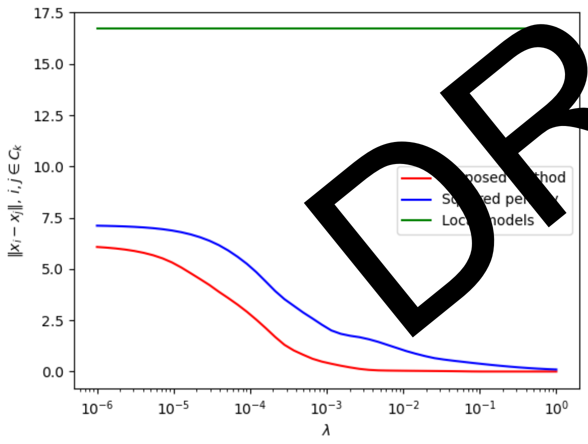


Fig. 3: The average distance between models across all users and all clusters. We can see that distance between models increases, as λ decreases, which is to be expected, as the models fit better to their local data. However, the proposed method results in more compact solutions within clusters, compared to the squared penalty one.

Figures 2 and 3 show the following. For λ sufficiently large, our method enforces consensus, and effectively performs as the standard FL method (1). For λ sufficiently small, the proposed method achieves both personalization, as it significantly outperforms the global model, as well as generalization, as the performance on the the complete cluster data is better than the strictly local models. Compared to the squared penalty model (4), we note that our method recovers the global model for sufficiently large λ, while the squared penalty method can recover the global model only asymptotically, as λ tends to infinity. The highest average accuracy is achieved by our method, being at 86.8%, compared to the highest average accuracy of the squared penalty method, being at 86.6%. Figure 3 shows that our method constantly produces more compact clusters, i.e. the average distance between solutions within clusters is constantly smaller than the one produced by the squared penalty method .

Hence, we can see that the proposed formulation (3) achieves a comparable or slightly better peak accuracy with respect to (4), while producing more compact models, i.e., significantly reducing the number of distinct models that need to be kept in the overall FL system. We report that the proposed model (3) exactly recovers the cluster structure (produces equal user models within clusters and finds 3 clusters) for $\lambda \in (0.0892, 0.0919)$.

Finally, we evaluated the performance of PDMM for solving (3). This result, as well as some additional numerical simulations, can be found in Appendix C.

VI. Conclusion

We proposed a novel approach to personalized federated learning that, in addition to personalization and generalization, allows for clustering of users' local models. The approach is based on a novel formulation of personalized FL wherein we minimize the sum of local users' costs with respect to their local models, subject to a penalization term that penalizes the local models' differences via a sum-of-norms penalty. We prove exact cluster recovery guarantees for a general class of local users' costs, assuming that the penalty parameter λ that weighs the sum-of-norms penalty falls within an appropriately defined range. We further explicitly characterize this range in terms of within-clusters and across-clusters heterogeneity of local users' costs (models). As an interesting byproduct, these results represent a direct generalization of convex clustering recovery guarantees for more general per-data point losses. Next, we propose an efficient algorithm based on the Parallel Direction Method of Multipliers (PDMM) to solve the proposed formulation in a federated server-users setting. Numerical experiments illustrate and corroborate the results.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings

Note that the considered simulation setup is such that the inter cluster generalization does not help. That is, data distributions across different clusters are so far apart, that the oracle, that ignores the data from other clusters, works best. Clearly, if the data distributions across different clusters are very close, another oracle that exploits inter-cluster generalization may be considered. Consider the extreme case when the data distributions across the three clusters are mutually (almost) equal, i.e., the distribution difference is negligible, but this is not known beforehand. Then, clearly, the global model (1) becomes best-performing and an "oracle" model, in the sense that it implicitly utilizes the unknown cluster structure. Our formulation (3) can uncover this and actually match the global model for λ above a threshold, and hence perform optimally. On the other hand, (4) gets closer to the global model as λ increases, but never matches it. This illustration presents a scenario when there is clear merit in across-cluster generalization that the proposed formulation (3) is able to harness.

of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[2] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," *arXiv preprint arXiv:2002.04758*, 2020.

[3] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[4] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," 2021.

[5] A. Fallah, A. Mokhtari, and A. E. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[6] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage, "Federated evaluation of on-device personalization," 2019.

[7] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[8] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, "Deep mutual learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4320–4328.

[9] I. Bistritz, A. Mann, and N. Bambos, "Distributed distillation for on-device learning," in *NeurIPS*, 2020.

[10] Y. J. Cho, J. Wang, T. Chiruvolu, and G. Joshi, "Personalized federated learning for heterogeneous clients with clustered knowledge transfer," *ArXiv*, vol. abs/2109.08119, 2021.

[11] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," 2021.

[12] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3710–3722, 2021.

[13] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *ArXiv*, vol. https://arxiv.org/abs/2002.10619, 2020.

[14] D. Sun, K.-C. Toh, and Y. Yuan, "Convex clustering: Model, theoretical guarantee and efficient algorithm," *Journal of Machine Learning Research*, vol. 22, 2021.

[15] F. Hanzely, S. Hanzely, S. Horvath, and P. Richtarik, "Lower bounds and optimal algorithms for personalized federated learning," in *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020.

[16] H. Wang, A. Banerjee, and Z.-Q. Luo, "Parallel direction method of multipliers," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014.

[17] W. Ben-Ameur, P. Bianchi, and J. Jakubowicz, "Robust distributed consensus using total variation," *IEEE Trans. Aut. Contr.*, vol. 61, no. 6, 2016.

[18] A. Panahi, D. Dubhashi, F. D. Johansson, and C. Bhattacharyya, "Clustering by sum of norms: Stochastic incremental algorithm, convergence and cluster recovery," in *Proceedings of the 34th International Conference on Machine Learning, PMLR 70*, 2017, pp. 2769–2777.

[19] M. Hong, M. Razaviyayn, Z.-Q. Luo, and J.-S. Pang, "A unified algorithmic framework for block-structured optimization involving big data: With applications in machine learning and signal processing," *IEEE Sig. Proc. Mag.*, vol. 33, no. 1, pp. 57–77, 2016.

[20] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[21] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A rewriting system for convex optimization problems," *Journal of Control and Decision*, vol. 5, no. 1, pp. 42–60, 2018.

[22] T. D. Hocking, A. Joulin, F. Bach, and J.-P. Vert, "Clusterpath An Algorithm for Clustering using Convex Fusion Penalties," in *28th international conference on machine learning*, United States, Jun. 2011, p. 1.

Appendix A
Practical considerations

In this section we discuss some practical aspects regarding the penalty parameter λ of the formulation (3). Namely, we present upper and lower bounds alternative to (6)-(13), that still uncover the clustering structure. We also discuss how to select λ in practice.

Remark 2. *Note that, in* (6)*, the right hand side depends on λ through the $w_k^\star(\lambda)$'s. We can replace condition* (6) *with a more conservative condition as follows. Denote by $G : \mathbb{R}^{N\,K} \to \mathbb{R}$, the following function: $G(w_1,\dots,w_K) = \frac{1}{N}\sum_{k=1}^K n_k\, g_k(w_k)$. By Assumption 1, G is coercive and hence it has compact sub-level sets. Also, let $g^\star$ be the optimal value of* (5)*, i.e.:*

$$g^\star = \frac{1}{N}\sum_{k=1}^{K} n_k g_k(w_k^\star) + \lambda \sum_{k\neq l} n_k n_l \|w_k^\star - w_l^\star\|. \tag{20}$$

Then, clearly, $G(w_1^\star,\dots,w_K^\star) \le g^\star$. On the other hand, we have

$$g^\star \le \frac{1}{N}\sum_{k=1} n_k g_k(y) + \sum_{l\neq k} n_k n_l \|y^\star - y^\star\| = \frac{1}{N}\sum_{i=1}^{N} f_i(y^\star) = f^\star, \tag{21}$$

where we recall that $y^\star$ is a solution to (1)*. Therefore, we conclude that, for any $\lambda \ge 0$, $\{w_k^\star(\lambda)\}$ belongs to the compact set*

$$\mathcal{W} := \{w_1,\dots,w_K : G(w_1,\dots,w_K) \le f^\star\}. \tag{22}$$

Therefore, Theorem 1 continues to hold if we replace (6) *with the following more stringent requirement:*

$$\lambda \ge \underline{\lambda} := \max_{k=1,\dots,K}\ \max_{i,j\in C_k}\ \max_{\{w_k\}\in\mathcal{W}} \frac{\|\nabla f_i(w_k) - \nabla f_j(w_k)\|}{n_k}, \tag{23}$$

Clearly, set $\mathcal{W}$ in (23) *can be replaced with a larger compact set $\{w_1\dots,w_K : G(w_1,\dots,w_K) \le \frac{1}{N}\sum_{i=1}^N f_i(x^\bullet)\}$, with arbitrary $x^\bullet \in \mathbb{R}^d$, e.g., $x^\bullet = 0$.*

Remark 3. *Note that, in* (13)*, the right hand side also depends on λ through the $w_k^\star(\lambda)$. We can replace* (13) *with the following more conservative condition:*

$$\lambda < \overline{\lambda} := \frac{\min_{k,l\in[K],k\neq l}\min_{\{w_k\}\in\mathcal{W}} \|\nabla g_k(w_k) - \nabla g_l(w_k)\|}{2\max_{k\in[K]}\sum_{l\neq k} n_l}, \tag{24}$$

where set $\mathcal{W} \subset \mathbb{R}^{d\,K}$ is defined in (22)*.*

Remark 4. *Note that, if $\underline{\lambda} < \overline{\lambda}$, then, for any $\lambda \in (\underline{\lambda}, \overline{\lambda})$, both Theorems 1 and 2 hold, i.e., formulation* (3) *perfectly recovers the f_i's cluster structure $C_1,\dots,C_K$, and moreover the models $w_k^\star$'s, $k = 1,\dots,K$, that correspond to different clusters, are mutually distinct. Intuitively, condition $\underline{\lambda} < \overline{\lambda}$ requires that the (appropriately scaled) within-clusters function heterogeneity is smaller than the (appropriately scaled) between-clusters function heterogeneity.*

In practice, we may not know quantities $\underline{\lambda}$ and $\overline{\lambda}$. Similarly to convex clustering approaches, e.g. [22], we can solve (3) for a set of values of the penalty parameter λ, $\{\lambda^{(r)}\}$, $r = 1,\dots,R$, i.e., we can generate a solution path. In more detail, we can set $\lambda^{(r+1)} = c\,\lambda^{(r)}$, $r = 1,2,\dots,$ for a small positive $\lambda^{(1)}$, where $c > 1$ is a constant, and R is the smallest index r such that the number of distinct vectors $x_i^\star(\lambda^{(R)})$,

$i = 1, \dots, N$, is one. When solving (3) for $\lambda = \lambda^{(r+1)}$, the numerical solver of (3) (e.g., see ahead Algorithm 1) can use warm start, i.e., it can be initialized with $\{x_i^\star(\lambda^{(r)})\}$. The resulting solution path $\{x_i^\star(\lambda^{(r)})\}$, $r = 1, \dots, R$, will typically have a non-increasing number of mutually distinct models x_i's, with N distinct models for $\lambda^{(1)}$ and a sufficiently small $\lambda^{(1)}$, and one distinct model for $\lambda^{(R)}$.

If, for a certain value r (or for a range of values r), we have that $\lambda^{(r)} \in (\underline{\lambda}, \overline{\lambda})$ and there holds $\underline{\lambda} < \overline{\lambda}$ (a hidden cluster structure exists), then solution $\{x_i^\star(\lambda^{(r)})\}$ satisfies Theorems 1 and 2, i.e., the hidden cluster structure is uncovered. For a fixed $\lambda^{(r)}$, we can check whether the hidden cluster structure is uncovered in an "a posteriori" way as follows. We first fix the clustering $C_1 = C_1(\lambda^{(r)}), \dots, C_K = C_K(\lambda^{(r)})$ induced by $\{x_i^\star(\lambda^{(r)})\}$, and then we check whether conditions (23) and (24) hold.

Even when we are not directly concerned with uncovering the hidden cluster structure in the sense of Theorems 1 and 2, there are several merits of computing the solution path, similarly to the convex clustering scenario, e.g., [22]. For example, depending on the requirements of a given application, we can select the index r, i.e., the model $\{x_i^\star(\lambda^{(r)})\}$ for which the number of distinct x_i's (closely) matches the need of the current application. Furthermore, increasing λ translates into improving the degree of generalization and reducing the degree of personalization, so one can select the appropriate $\{x_i^\star(\lambda^{(r)})\}$ according to the current application needs.

Appendix B
Further insights through a special case

In this section, we provide further insight into the method by analyzing a special case. We consider an explicit clustering structure, given by the following assumption.

Assumption 3. *There exists a node partition $C_1, C_2, \dots, C_K$, and parameters $\epsilon_k, \delta_{kl} > 0$, $k, l = 1, \dots, K$, $k \neq l$, such that the following holds:*

$$\sup_{x \in \mathbb{R}^d} \|\nabla f_i(x) - \nabla f_j(x)\| \leq \epsilon_k, \ \forall i, j \in C_k, \tag{25}$$

$$\inf_{x \in R^d} \|\nabla f_i(x) - \nabla f_j(x)\| \geq \delta_{kl}, \ \forall i \in C_k, \ \forall j \in C_l, \ k \neq l. \tag{26}$$

We then have the following result.

Theorem 3. *Let Assumptions 1, 2 and 3 hold. Then, for any λ satisfying*

$$\lambda \in \left[\max_{k \in [K]} \frac{\epsilon_k}{n_k}, \frac{\min_{k \neq l} \left[\delta_{kl} - (\epsilon_k + \epsilon_l) \right]}{2 \max_{k \in [K]} \sum_{l \neq k} n_l} \right], \tag{27}$$

the conditions of Theorems 1 and 2 are satisfied.

Remark 5. *Note that conditions (25) and (26) can be interpreted as measures of within-cluster homogeneity and between-cluster heterogeneity, respectively. In particular, let $x_i^\dagger \in \mathbb{R}^d$ denote a optima of $f_i(x)$, $i \in [N]$. Then, per (25) and (26), we have*

$$\|\nabla f_j(x_i^\dagger)\| \leq \epsilon_k, \forall i, j \in C_k,$$
$$\|\nabla f_j(x_i^\dagger)\| \geq \delta_{kl}, \forall i \in C_k, \forall j \in C_l, k \neq l.$$

From convexity of f_i's (Assumption 1) for any $k \in [K]$, $i, j \in C_k$, if ϵ_k small, we can expect $f_j(x_i^\dagger)$ to be a good approximation to $f_j(x_j^\dagger)$. On the other hand, for any $k \neq l$ and $i \in C_k$, $j \in C_l$, using Lipschitz continuity of the gradients of f_i's, we have

$$f_j(x_i^\dagger) - f_j(x_j^\dagger) \geq \frac{1}{2L} \|\nabla f_j(x_i^\dagger)\|^2 \geq \frac{\delta_{kl}^2}{2L}.$$

Hence, for δ_{kl} large, $f_j(x_i^\dagger)$ can be an arbitrarily bad approximation to $f_j(x_j^\dagger)$. Therefore, ϵ_k and δ_{kl} can be interpreted as natural measures of within-cluster homogeneity and between-cluster heterogeneity, respectively.

Remark 6. *Theorem 3 states that the clustering structure of the solution is maintained, for any choice of λ satisfying (27). Compared to the results from Theorems 1 and 2, the resulting interval (if existent) is smaller, but the lower and upper bounds are independent of λ, and of the optimal solutions of problem (5).*

Remark 7. *Compared with (23) and (24), while possibly smaller, the interval (27) provides a more natural interpretation: if the between-cluster heterogeneity is sufficiently larger than the within-cluster homogeneity, so the interval (27) is non-empty, a (strong) clustering structure among users exists, and can be recovered by (3).*

We now prove Theorem 3.

Proof. From Assumption 3 it directly follows that

$$\max_{k \in [K]} \max_{i, j \in C_k} \frac{\|\nabla f_i(w_k^\star) - \nabla f_j(w_k^\star)\|}{n_k} \leq \max_{k \in [K]} \frac{\epsilon_k}{n_k}.$$

Next, for any $k \in [K]$, and $i \in C_k$, we have

$$\begin{aligned} \|\nabla g_k(x) - \nabla f_i(x)\| &= \left\| \frac{1}{n_k} \sum_{j \in C_k} \Big(\nabla f_j(x) - \nabla f_i(x) \Big) \right\| \\ &\leq \frac{1}{n_k} \sum_{j \in C_k \setminus \{i\}} \|\nabla f_j(x) - \nabla f_i(x)\| \\ &\leq \frac{n_k - 1}{n_k} \epsilon_k < \epsilon_k. \end{aligned}$$

For any $k, l \in [K]$, $k \neq l$, and any $i \in C_k$, $j \in C_l$, we then get

$$\begin{aligned} \|\nabla g_k(x) - \nabla g_l(x)\| &\geq \|\nabla f_i(x) - \nabla f_j(x)\| \\ &\quad - \|\nabla g_k(x) - \nabla f_i(x)\| - \|\nabla f_j(x) - \nabla g_l(x)\| \\ &> \delta_{kl} - \epsilon_k - \epsilon_l. \end{aligned}$$

Plugging in w_k^* and taking the min with respect to $k \neq l$ gives

$$\min_{k \neq l} \|\nabla g_k(w_k^*) - \nabla g_l(w_k^*)\| > \min_{k \neq l} (\delta_{kl} - \epsilon_k - \epsilon_l).$$

Finally, dividing both sides by $2 \max_{k \in [K]} \sum_{l \neq k} n_l$ gives the desired result. $\square$

Note that we know that, for $\lambda \geq \widehat{\lambda}$, for some $\widehat{\lambda} > 0$, all the $x_i^\star(\lambda)$'s coincide [17]. Hence, all the $x_i^\star(\lambda)$'s are necessarily mutually equal for a certain $\lambda^{(R)}$, for some finite $R > 0$.

Moreover, assuming strong convexity, Assumption 3 implies a clustering structure among the local solutions, as shown by the following result.

Theorem 4. *Let each $f_i(x)$, $i = 1, \dots, N$, be μ-strongly convex, and let Assumptions 2, 3 hold. Denote by $x_i^\dagger$ the (global) optima of $f_i(x)$, $i = 1, \dots, N$. Then, the following holds:*

$$\|x_i^\dagger - x_j^\dagger\| \le \frac{\epsilon_k}{\mu}, \forall i, j \in C_k, \tag{28}$$

$$\|x_i^\dagger - x_j^\dagger\| \ge \frac{\delta_{kl}}{L}, \forall i \in C_k, \ \forall j \in C_l, \ k \neq l. \tag{29}$$

Remark 8. *Theorem 4 shows that optimal models of users belonging to the same clusters are at least $\tilde{\epsilon}$ close, with optimal models of users belonging to different clusters being at least $\tilde{\delta}$ apart. Here, $\tilde{\epsilon} = \max_{k \in [K]} \frac{\epsilon_k}{\mu}$, and $\tilde{\delta} = \min_{k \neq l} \frac{\delta_{kl}}{L}$. In the case that $\tilde{\epsilon} < \tilde{\delta}$, the clustering structure implied by Theorem 4 is strong, in the sense that the local optima corresponding to different clusters are well separated.*

Remark 9. *In the case that $\tilde{\epsilon} < \tilde{\delta}$ and the interval given by (27) is non-empty, Theorems 3 and 4 show that a natural clustering structure among the user's costs exists. In addition, the proposed formulation (3) uncovers the said structure, for any λ in a range that is independent of the optimal solutions of problem (5).*

We now prove Theorem 4

Proof. From Assumption 3, for $i \in C_k$, $j \in C_l$, $k \neq l$, and for all $x \in \mathbb{R}^d$, we have

$$\delta_{kl} \le \|\nabla f_i(x) - \nabla f_j(x)\|.$$

In particular, for $x = x_i^\dagger$, we have

$$\delta_{kl} \le \|\nabla f_j(x_i^\dagger)\| = \|\nabla f_j(x_i^\dagger) - \nabla f_j(x_j^\dagger)\| \le L\|x_i^\dagger - x_j^\dagger\|,$$

implying (29). Similarly, for $i, j \in C_k$, and for all $x \in \mathbb{R}^d$, we have

$$\|\nabla f_i(x) - \nabla f_j(x)\| \le \epsilon_k.$$

In particular, for $x = x_i^\dagger$, we have

$$\epsilon_k^2 \ge \|\nabla f_j(x_i^\dagger)\|^2 \ge 2\mu(f_j(x_i^\dagger) - f_j(x_j^\dagger)), \tag{30}$$

where we used the Polyak-Lojasiewitz inequality in the second step. From strong convexity of f_j, we have

$$f_j(x_i^\dagger) \ge f_j(x_j^\dagger) + \frac{\mu}{2}\|x_i^\dagger - x_j^\dagger\|^2.$$

Rearranging and plugging into (30), we get

$$\epsilon_k^2 \ge \mu^2\|x_i^\dagger - x_j^\dagger\|^2,$$

which implies (28). □

Appendix C
Additional Experiments

Here, we present some additional experiments. First, we evaluate the clustering recovery of (3) and compare it with (4). The data is generated using the same methodology described in Section V, with each ellipse containing 100 data points, hence each cluster contains a total of 200 data points. The dataset is shown in Figure 4. We generate $n = 10$ users per cluster, and each user samples 85% of points from the cluster it belongs to, ensuring the resulting models will be similar for users within clusters. The performance of (3) and (4) is evaluated for different values of the penalty parameter λ, belonging to $\{1000, 1, 0.12, 0.08, 0.05, 0.03, 0.02, 0.01\}$. The dashed lines in the figures correspond to optimal separators for each cluster, evaluated on the full data. The full lines represent model estimates corresponding to (3) and (4). The results are presented in Figure 6.

We can see that Figure 6 corroborates the results from Theorems 1 and 2. In particular, for λ sufficiently large, our method produces only one model across all users. For λ higher than, but close to the theoretical upper bound from Theorem 2, the method produces two models across all the users (second row, left image). Finally, for λ within the theoretical range, our method produces exactly three models, uncovering the clustering structure. We remark that, while the produced models are not optimal for the training data, they can still be used as a guideline: if a clustering solution is obtained, and the parameter λ falls within the theoretical bounds suggested by Theorems 1 and 2, an innate clustering structure is uncovered, and the users that are selected in the clusters can focus on training their models within the cluster of similar users. Moreover, the left image in the third row suggests that the bounds obtained by Theorems 1 and 2 are somewhat conservative - the proposed method produced 3 models across users, for λ that is lower than the theoretical lower bound, meaning that in practice, the clustering structure can be recovered for a wider range of λ than predicted by theory.

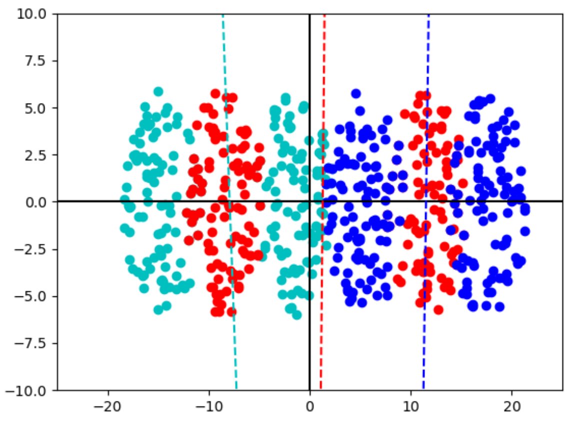


Fig. 4: Data used for evaluating clustering recovery. Each ellipse contains 100 datapoints.

In the second experiment, we present the convergence results of PDMM for solving the proposed method in federated settings. The dataset and the parameters are the same as was described in the setting above. For PDMM in the federated settings, in each iteration we randomly select a subset of

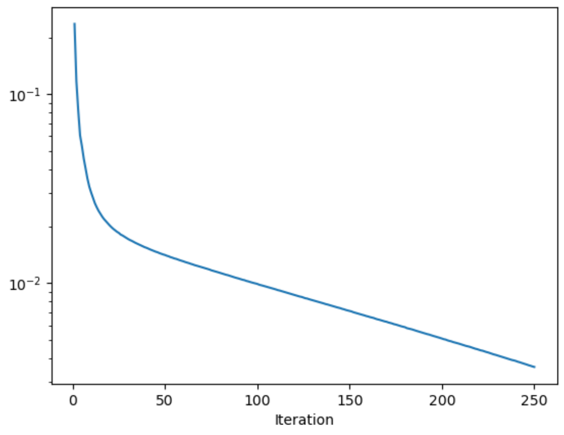


Fig. 5: PDMM convergence for solving (3) in federated settings.

users, being 40% of the total users. The PDMM tuning parameters are selected as $\rho = 10$, $\eta = 10$, $\tau = \frac{4}{5}$, $\nu = \frac{1}{5}$. The Bregman divergence in PDMM is set to be the squared Euclidean distance. All variables in PDMM (primal and dual) are initialized to zero vectors.

We evaluate the cost function in (3) achieved by PDMM at each iteration t. We also evaluate the optimal value of the cost in (3) via CVXPY. We then plot the difference (optimality gap) along iterations. To evaluate the performance of PDMM for (3), we used the primal variables x_i's. In particular, we plot $F(x^{(t)}) - F^{*}$, where F is the objective defined in (3) and F^{*} denotes the optimal value of F, computed via CVXPY. The parameter λ is set to 0.02. The results are presented in Figure 5.

While this is not pursued here, we remark that it is possible to reduce Algorithm 1 complexity by dynamically (over iterations) exploiting the clustered structure of a solution to (3). Namely, note that, for a solution $\{x_i^\star\}$ of (3), if $x_i^\star = x_j^\star$ for some $i \neq j$, then $z_{ij}^\star = 0$. This may be exploited in the algorithm implementation as follows: once user i detects that $z_{ij}^{(t)}$ is close to zero over a range of consecutive iterations, it can cease updating $z_{ij}^{(t)}$ and cease communicating it to the server, and it can also replace locally $x_j^{(t)}$ with $x_i^{(t)}$ in its subsequent updates. This also translates into reducing the communication cost at the server side, as $x_j^{(t)}$ no longer needs to be communicated to user i. In this way, storage, computational, and communication costs may be dynamically reduced as the algorithm evolves.

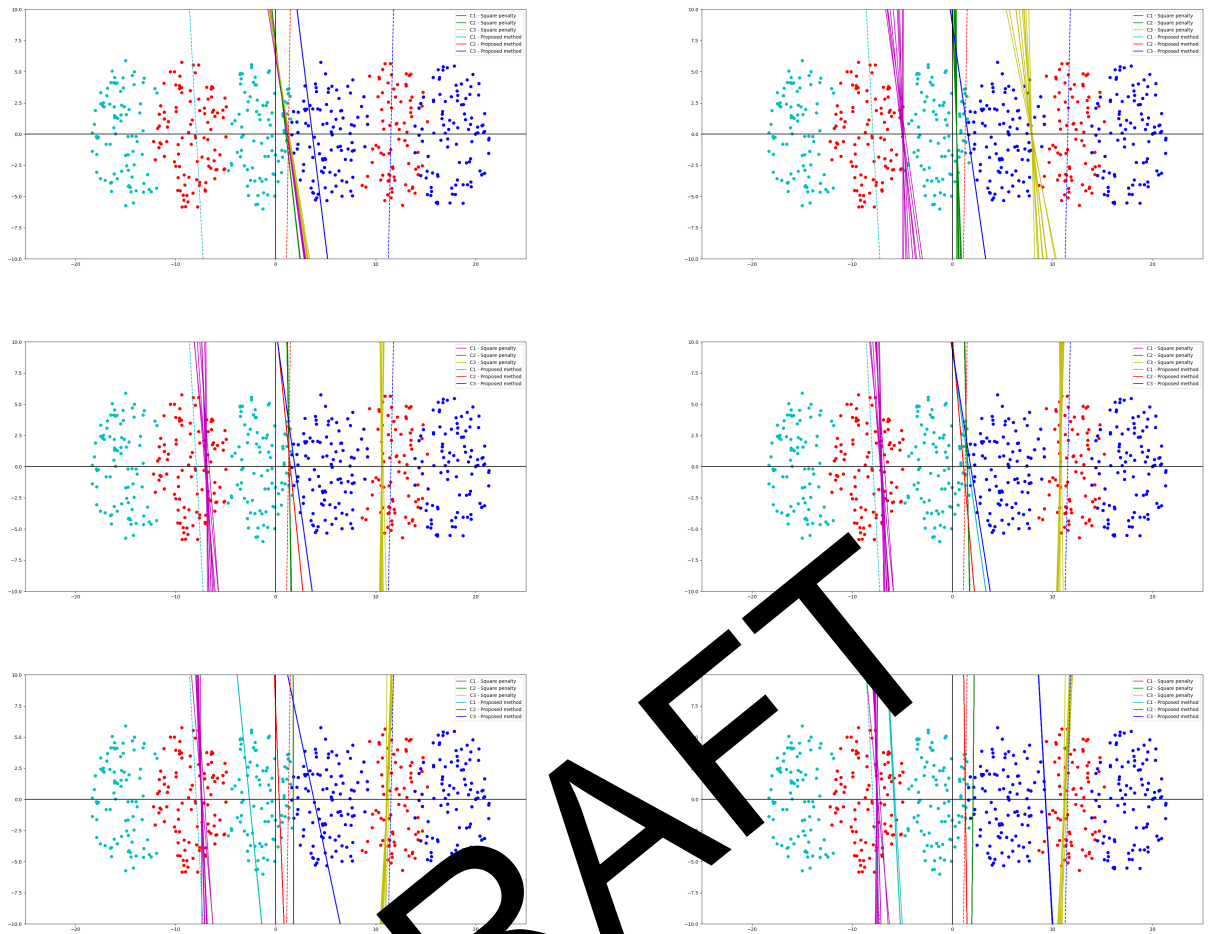


Fig. 6: Clustering structure of the models. The plots show how the number of distinct models changes with the penalty parameter λ. In particular, the values of λ correspond to $\{1000, 1, 0.12, 0.08, 0.05, 0.03, 0.02, 0.01\}$ from left to right, top to bottom, respectively. Note that the squared penalty method does not achieve consensus even for very high values of λ.

B One-Shot Clustered Federated Learning

The appended preprint follows.

One-Shot Federated Learning for Model Clustering and Learning in Heterogeneous Environments

Aleksandar Armacki
Carnegie Mellon University, Pittsburgh, PA
aarmacki@andrew.cmu.edu

Dragana Bajovic
Faculty of Technical Sciences, University of Novi Sad, Novi Sad
dbajovic@uns.ac.rs

Dusan Jakovetic
Faculty of Sciences, University of Novi Sad, Novi Sad
dusan.jakovetic@dmi.uns.ac.rs

Soummya Kar
Carnegie Mellon University, Pittsburgh, PA
soummyak@andrew.cmu.edu [*]

Abstract

We propose a communication efficient approach for federated learning in heterogeneous environments. The system heterogeneity is reflected in the presence of K different data distributions, with each user sampling data from only one of K distributions. The proposed approach requires only one communication round between the users and server, thus significantly reducing the communication cost. Moreover, the proposed method provides strong learning guarantees in heterogeneous environments, by achieving the optimal mean-squared error (MSE) rates in terms of the sample size, i.e., matching the MSE guarantees achieved by learning on all data points belonging to users with the same data distribution, provided that the number of data points per user is above a threshold that we explicitly characterize in terms

[*]This work is supported by the European Union's Horizon 2020 Research and Innovation program under grant agreement No 957337. The paper reflects only the view of the authors and the Commission is not responsible for any use that may be made of the information it contains.

of system parameters. Remarkably, this is achieved without requiring any knowledge of the underlying distributions, or even the true number of distributions K. Numerical experiments illustrate our findings and underline the performance of the proposed method.

1 Introduction

Federated learning (FL) is a paradigm where many users collaborate, with the aim of jointly training a model [1]. Formally, the goal is to solve the problem

$$\arg\min_{\theta \in \Theta} F(\theta) = \frac{1}{m} \sum_{i=1}^{m} F_i(\theta), \tag{1}$$

where $\Theta \subset \mathbb{R}^d$ is the parameter space, $m \in \mathbb{N}$ represents the number of users, while $F_i : \mathbb{R}^d \mapsto \mathbb{R}$, $i = 1, \dots, m$ is the loss of user i.

Unlike in centralized learning, where a single user has access to all the data, in FL each user stores its data locally. This is an important feature, as typically huge amounts of users participate in the process and generate enormous amounts of data, therefore imposing significant storage cost for a single user. Additionally, the nature of the data is often sensitive, which incentivizes the users to keep their data private. The training process is coordinated by a central server, which typically includes updating and sharing the global model with the users.

While such an approach helps alleviate the storage and computation burden for any single user, it imposes significant communication costs on the system as a whole [2]. To tackle this issue, different approaches have been proposed, such as quantization [3], [4], [5], gradient sparsification [6], [7], specialized user sampling [8], [9], [10], local methods [11], [12], [13] and one-shot methods [14], [15], [16], [17], [18], to name a few. Another issue associated with training a global model comes from system heterogeneity. Users that participate in FL often contain datasets generated by different distributions, making the system as a whole highly heterogeneous. A global model can therefore be very bad for an individual user [19], [20].

One way to alleviate the issues stemming from training a global model is for each user to train their own model. Formally, the goal of such an approach is to solve the problem

$$\arg\min_{\theta_1, \dots, \theta_m \in \Theta} F_L(\theta_1, \dots, \theta_m) = \frac{1}{m} \sum_{i=1}^{m} F_i(\theta_i).$$

Such an approach leads to models that can be trained locally and eliminates the need for any communication. However, it is completely oblivious to any underlying similarities that might exist between users. Moreover, a strictly local approach often suffers from data imbalance - while some users generate abundance of data, the majority of users generate only a few data samples. This results in the majority of users being unable to learn useful models on their own [20].

Many different approaches that address the shortcomings of the global and purely local models have been proposed. One such approach is personalized federated learning (PFL). The goal of PFL is to learn models that fit each individual user, while utilizing the federation to produce models that generalize better. Many approaches to personalization have been proposed, such as multi-task learning [21], [22], [23], meta-learning [24], [25], fine-tuning [26], [27].

Another closely related approach is clustered federated learning (CFL). The underlying assumption in clustered federated learning is the presence of K different data distributions $\mathcal{D}_k$, $k \in [K]$, with each user sampling their data from only one of K distributions. This leads to a natural clustering of users, i.e., we can define clusters $\{C_k\}_{k\in[K]}$ given by

$$C_k = \{i \in [m] : \text{user } i\text{'s data follows distribution } \mathcal{D}_k\}.$$

Since each user contains data from exactly one of K distributions, the clusters form a disjoint partition of the set of all users, i.e., we have

$$\cup_{k\in[K]} C_k = [m] \text{ and } C_k \cap C_j = \emptyset,\ \forall k \neq j.$$

In such a scenario the goal is to learn K models associated with the underlying clusters, so that users belonging to the same clusters have the same models. This is somewhat different from the classical PFL approaches, where the goal is to produce m models, one for each user. Allowing for $1 \leq K \leq m$, clustered federated learning can again be seen as an intermediary between the global and local learning, with $K = 1$ resulting in a global model, while $K = m$ resulting in purely local models. There are many works assuming a clustered structure among users, such as [28], [29], [30], [31], [32].

While existing works in CFL focus on dealing with heterogeneity and personalization aspects, none of them focus on communication efficiency. The aim of this paper is to provide a method for CFL that maintains the benefits of standard clustering-based approaches, while simultaneously achieving communication efficiency. This is achieved by developing a one-shot federated learning method, that requires only one round of communication.

Literature review. We next review the related literature, in particular, one-shot methods in FL and methods for CFL.

One-shot methods in the context of FL have been studied in [14], [15], [16], [17], [18]. [14], study one-shot averaging methods. Each user trains a model on the local data and the server produces the final model by averaging all the users' models. The authors show that, for strongly convex loss f_i, the methods can achieve the same MSE guarantees as centralized learning, i.e., order-optimal rates in terms of sample size, provided that the number of samples available to each user is higher than a threshold. [15] propose to train $K \leq m$ ensemble based methods for supervised and semi-supervised problems. [16] propose a one-shot distillation method, wherein the users send a distilled version of their local dataset to the server, which then performs the global model training. [17] study one-shot methods in federated settings under constraints on the communication budget. The proposed method is, under certain regimes, order-optimal up to logarithmic factors, while simultaneously relaxing the higher-order smoothness assumptions made in [14]. [18] introduce a one-shot FL method in heterogeneous settings, designed for data clustering. The methods [14] and [17] provide strong theoretical guarantees[1], however, they assume that the data across all users follows a single distribution $\mathcal{D}$. Therefore, they focus on training a single global model to be used for all users. In modern FL systems the data across different users typically comes from different distributions, hence violating the IID assumptions made in prior works. Moreover, the user heterogeneity stemming from this phenomena is known to hamper the global model [19]. The methods [15] and [16] consider heterogeneous settings, but provide no theoretical analysis of their methods. To the best of our knowledge, no theoretical results for one-shot methods are established under the presence of heterogeneity, i.e., multiple data distributions in the system.

CFL has been studied in [28], [29], [30], [31], [32]. [28] and [29] propose similar methods, that iteratively estimate cluster membership and perform model updates. [29] show an exponential convergence rate up to an error floor that is order-optimal up to a logarithmic factor, in the number of samples and users. [30] propose a robust algorithm for CFL, under the presence of adversarial users. If there are no adversarial users (the setting that we consider in this paper), the method is order-optimal up to a logarithmic factor. [31] propose a method for CFL that can be applied to any standard FL method, as a fine-tuning step. The method is based on successive bi-

[1]The method [18] provides a theoretical analysis under heterogeneous settings. However, the method is not a general learning method, but a method designed for clustering.

partitioning of the current set of users, based on cosine similarity and does not require prior knowledge of the number of clusters K. However, when a bi-partitioning is performed, each partition needs to do a full FL training on the newly formed partition/federation, potentially requiring multiple rounds of model re-training and communication. [32] propose a method that aims to simultaneously infer the clustering of users and train models. The proposed method does not require knowledge of K and establishes explicit conditions under which the true clustering can be recovered. All of the methods require potentially many rounds of communication and model training. The methods [28], [29] and [30] require multiple communication rounds and prior knowledge of the number of clusters K. While the methods in [31] and [32] do not require knowledge of K, they require many rounds of communication.

Contributions. In this paper we propose a one-shot method for CFL, that is able to deal with system heterogeneity. We study the convergence guarantees of the method, in terms of the MSE with respect to the number of samples, as in [14], [29] and [30]. Our contributions can be summarized as follows:

1. We propose a one-shot method for CFL under the presence of multiple data distributions (i.e., system heterogeneity). The method is communication efficient, by only requiring one round of communication. Moreover, the proposed method does not require knowledge of the true number of clusters K.

2. We show that, for strongly convex costs, the method achieves the order-optimal MSE guarantees in terms of sample size, i.e., it matches the order-optimal MSE guarantees of centralized learning, provided that users have sufficient number of samples. This establishes regimes in which communication beyond the first round is not necessary for achieving order-optimality.

3. We show that, compared to existing methods, our algorithm reduces communication cost by a factor of $\mathcal{O}\left(\frac{\kappa}{p}\log\left(\frac{2D}{\varepsilon}\right)\right)$, while improving the rates by a factor logarithmic in the number of samples and users.

4. We explicitly derive the expression for the requirements on the number of data points for the users to achieve the order-optimal MSE rate and show how it depends on various system parameters, like the size of clusters, difficulty of the clustering problem, as well as problem related parameters (e.g., strong convexity constant).

5. We propose a method for discovering the underlying clustering structure of the users and establish conditions under which the method recovers the true cluster membership. The proposed method does not require prior knowledge of the true number of underlying distributions K.

6. We verify our theoretical findings via numerical experiments on linear regression problems, showing the proposed method achieves the order-optimal MSE rate and matches the performance of oracle methods that know the true cluster membership beforehand.

Paper organization. The rest of the paper is organized as follows. Section 2 introduces the relevant background and formally states the problem. Section 3 describes the proposed method. Section 4 presents the main results of the paper. Section 5 presents numerical results. Finally, Section 6 concludes the paper. The reminder of the section introduces the notation used throughout the paper.

Notation. The set of real numbers is denoted by $\mathbb{R}$, while $\mathbb{R}^d$ denotes the corresponding d-dimensional vector space. $\mathbb{N}$ denotes the set of positive integers. $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product and $\| \cdot \|$ denotes the induced norm. In a slight abuse of notation, we will also use $\|\cdot\|$ to denote the corresponding matrix norm. $[m]$ denotes the set of positive integers up to and including $m \in \mathbb{N}$, i.e., $[m] = \{1, 2, \ldots, m\}$. For a collection of sets $\{S_k\}_{k\in[K]}$, we use $S_{(k)}$ to denote the k-th largest set. The notation $\mathcal{O}(\cdot)$, $\Omega(\cdot)$ refers to the standard "big O" and "big Omega" notation, respectively, i.e., for two non-negative sequences $\{a_n\}_{n\in\mathbb{N}}$ and $\{b_n\}_{n\in\mathbb{N}}$, the relation $a_n = \mathcal{O}(b_n)$ implies the existence of a global constant $C_1 > 0$ and $n_1 \in \mathbb{N}$, such that $a_n \leq C_1 b_n$, for all $n \geq n_1$, while $a_n = \Omega(b_n)$ implies the existence of a global constant $C_2 > 0$ and $n_2 \in \mathbb{N}$, such that $a_n \geq C_2 b_n$, for all $n \geq n_2$.

2 Problem formulation and preliminaries

In this section, we begin by formally stating the problem of interest. We introduce some assumptions and discuss their implications. In Subsections 2.1 and 2.2 we introduce the method from [14] and convex clustering, respectively. We begin by introducing the notions of population and empirical loss.

Consider m users, $i = 1, \ldots, m$, that participate in a federated learning system. The goal of standard FL approach is to train a shared model, by solving (1), where $F_i : \Theta \mapsto \mathbb{R}$ is the *population loss* of user i, given by

$$F_i(\theta) = \mathbb{E}_{X_i \sim \mathcal{D}_i}[\ell(\theta; X_i)]. \tag{2}$$

Here, $\Theta \subset \mathbb{R}^d$ is the parameter space, $\mathcal{D}_i$ is the data distribution of user i, $X_i \in \mathcal{X}$ is the data generating random variable distributed according to $\mathcal{D}_i$, $\mathcal{X} \subset \mathbb{R}^{d'}$ is the data space, while $\ell : \Theta \times \mathcal{X} \mapsto \mathbb{R}$ is a loss function.

In practice, users only have access to a finite data sample, hence the aim of federated learning systems is to solve

$$\arg\min_{\theta \in \Theta} f(\theta) = \frac{1}{m}\sum_{i=1}^{m} f_i(\theta), \tag{3}$$

where $f_i : \Theta \mapsto \mathbb{R}$ is the *empirical loss* of user i, given by

$$f_i(\theta) = \frac{1}{n_i}\sum_{j=1}^{n_i} \ell(\theta; x_{ij}). \tag{4}$$

Here, $n_i \in \mathbb{N}$ represents the number of data samples available to user i, while $x_{ij} \in \mathcal{X}$, $j = 1, \ldots, n_i$, represents independent, identically distributed (IID) samples available to user i, sampled from the population X_i. However, as we proceed to argue in the remainder of the section, solving (1)-(3) is not an optimal approach under the presence of strong user heterogeneity. To that end, we formally state the main assumption used throughout the paper.

Assumption 1. *There exist K different data distributions in the system, with $1 < K < m$, such that each user samples their data from only one of the distributions, i.e., for each $i \in [m]$, we have $\mathcal{D}_i = \mathcal{D}_k$, for some $k \in [K]$. Moreover, the population optimal models of each cluster $\theta_k^\star := \arg\min_{\theta_k \in \Theta} F_k(\theta_k)$, $k \in [K]$, satisfy*

$$\min_{k \neq l} \|\theta_k^\star - \theta_l^\star\| > 0.$$

Denote by D the minimal distance between population optima of different distributions, i.e.,

$$D = \min_{k,l \in [K],\, k \neq l} \|\theta_k^\star - \theta_l^\star\| > 0.$$

Assumption 1 provides a natural partitioning of the set of all users $[m]$, given by

$$C_k = \{i \in [m] : \text{ user } i\text{'s data follows distribution } \mathcal{D}_k\}.$$

We then have $\cup_{k \in [K]} C_k = [m]$ and $C_k \cap C_l = \emptyset$, for all $k \neq l$. We will denote the resulting partition of $[m]$ by $\mathcal{C}$, i.e., $\mathcal{C} = \{C_k\}_{k \in [K]}$.

Remark 1. *Assumption 1 can be interpreted as a measure of distance between different distributions. Intuitively, it states that the optimal model corresponding to one of the K different populations will not be a good model for any other population. In general, Assumption 1 can be relaxed, to allow for existence of m different distributions, one corresponding to each user, while requiring that some of them are sufficiently close. We refer the reader to Lemma 7 in the Appendix, for a formal result of this argument.*

Remark 2. *Assumption 1 also gives a lower bound on the error caused by using models coming from different clusters. To see this, let $i, j \in [m]$ be two users such that $i \in C_k$, $j \in C_l$, $k \neq l$ and $n_i \gg n_j$. If user j, due to a lack of available samples, decides to use the model trained by user i, given by $\widehat{\theta}_i = \arg\min_{\theta_i \in \Theta} f_i(\theta_i)$, then the error stemming from such an approximation is*

$$\|\widehat{\theta}_i - \theta_l^\star\| \geq \|\theta_k^\star - \theta_l^\star\| - \|\theta_k^\star - \widehat{\theta}_i\|. \tag{5}$$

By Assumption 1, the first term on the right-hand side of (5) is lower bounded by D. For the second term, under certain regularity conditions (to be formalized below), we can apply results from learning theory, e.g., [33], to get

$$\|\widehat{\theta}_i - \theta_l^\star\| \geq D - \Omega\left(\frac{1}{\sqrt{n_i}}\right).$$

The above equation tells us that the error floor of using a model from a different cluster grows with both D and the number of samples available to user i. For example, as we will show in Corollary 1 ahead, for $D > 2\sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}} \geq 2$ that allows our method to achieve order-optimal MSE rates, the error floor in the ideal case of balanced cluster sizes becomes

$$\|\widehat{\theta}_i - \theta_l^\star\| > 2 - \Omega\left(\frac{1}{\sqrt{n_i}}\right) = \Omega(1),$$

thus showing that the error of using a model coming from a different cluster has a constant error floor.

Under Assumption 1, the population loss in (1) can be rewritten as

$$F(\theta) = \sum_{k=1}^{K} \frac{|C_k|}{m} F_k(\theta). \tag{6}$$

Similarly, the empirical loss in (3) can be cast as

$$f(\theta) = \sum_{k=1}^{K} \frac{|C_k|}{m} g_k(\theta), \tag{7}$$

where $g_k : \Theta \mapsto \mathbb{R}$ is the cluster-wise loss

$$g_k(\theta) = \frac{1}{|C_k|}\sum_{i\in C_k} f_i(\theta).$$

Note that both (6) and (7) have a clear clustering structure. These formulations suggest a natural approach model training, where the goal is to find K different models, one corresponding to each cluster. Thereafter, each user from the cluster is assigned the cluster-wide optimal model. Formally, the goal is to find K models, by solving

$$\operatorname*{arg\,min}_{\theta_1,\ldots,\theta_K\in\Theta} \sum_{k=1}^{K} \frac{|C_k|}{m} g_k(\theta_k). \tag{8}$$

To see why such an approach is optimal recall that, for a user $i \in C_k$, the optimal population model is given by $\theta_k^\star = \arg\min_{\theta_k\in\Theta} F_k(\theta_k)$. If we denote the empirical risk minimizers (ERMs) as $\widehat{\theta}_i = \arg\min_{\theta_i\in\Theta} f_i(\theta_i)$, under some regularity conditions (to be formalized below), we know from, e.g., [33], that the following MSE guarantee holds

$$\mathbb{E}\|\theta_k^\star - \widehat{\theta}_i\|^2 = \mathcal{O}\left(\frac{1}{n_i}\right),$$

where n_i is the number of samples available to user i. Let $\widehat{\theta}_k$ be minimizer of the empirical cluster-wise cost, i.e., $\widehat{\theta}_k = \arg\min_{\theta_k\in\Theta} g_k(\theta_k)$. We then have the following MSE guarantee

$$\mathbb{E}\|\theta_k^\star - \widehat{\theta}_k\|^2 = \mathcal{O}\left(\frac{1}{\sum_{i\in C_k} n_i}\right),$$

which shows the clear benefits of clustered learning, as $\frac{1}{\sum_{i\in C_k} n_i} < \frac{1}{n_i}$, for all $i \in C_k$. On the other hand, by Assumption 1 and cluster design, the benefits of further merging (and/or modifying) the clusters, in terms of sample size, can potentially be significantly outweighted by the distribution skew between two different clusters (recall Remark 2).

We now state the rest of the assumptions used throughout the paper.

Assumption 2. *The parameter space $\Theta \subset \mathbb{R}^d$ is a compact, convex set, with $\theta_k^\star \in int\,\Theta$, for all $k \in [K]$.*

Remark 3. *Assumption 2 is a standard assumption on the parameter space in statistical learning literature, e.g., [14], [33], [34].*

Remark 4. *Assumption 2 implies the existence of a global constant $R > 0$ such that, for all $\theta \in \Theta$*

$$\|\theta\| \leq R.$$

Assumption 3. *For any fixed $x \in \mathcal{X}$, the loss function $\ell(\cdot; x) : \Theta \mapsto \mathbb{R}$ is:*

1. *Nonnegative, i.e., for any $\theta \in \Theta$, $\ell(\theta; x) \geq 0$.*
2. *Convex, i.e., for any $\theta, \theta' \in \Theta$, we have*

$$\ell(\theta'; x) \geq \ell(\theta; x) + \langle \nabla \ell(\theta; x), \theta' - \theta \rangle.$$

3. *Smooth, i.e., there exists a constant $L > 0$ such that, for any $\theta, \theta' \in \Theta$, we have*

$$\ell(\theta'; x) \leq \ell(\theta; x) + \left\langle \nabla \ell(\theta; x), \theta' - \theta \right\rangle + \frac{L}{2}\|\theta' - \theta\|^2.$$

Remark 5. *Assumption 3 requires ℓ to be non-negative, convex and smooth. It is a straightforward exercise to show that this in turns implies non-negativity, convexity and smoothness of all of $\mathcal{L}$, f, F_k and f_k, $k \in [K]$ and f_i, $i \in [m]$.*

Remark 6. *Note that the constant L for the smoothness condition is independent of the choice of x, i.e., L is a global constant that holds for any choice of $x \in \mathcal{X}$.*

Remark 7. *Recall that, under convexity of ℓ, the smoothness condition is equivalent to Lipschitz continuous gradient of ℓ, i.e., for any fixed $x \in \mathcal{X}$ and any $\theta, \theta' \in \Theta$, we have*

$$\|\nabla \ell(\theta; x) - \nabla \ell(\theta'; x)\| \leq L\|\theta - \theta'\|.$$

From Remark 7, we can see that, for each fixed $x \in \mathcal{X}$, the gradients of ℓ are continuous. Using the compactness of Θ, we can conclude that ℓ has bounded gradients over Θ, for any fixed $x \in \mathcal{X}$. Denote by S the global gradient bound, i.e., $S = \sup_{x \in \mathcal{X},\, \theta \in \Theta} \|\nabla \ell(\theta; x)\|$.

Next, from Remark 5 and compactness of Θ, we can conclude that each F_k, $k \in [K]$ and each $f_i, i \in [m]$, have bounded gradients on Θ. Denote the corresponding gradient bounds by G_{F_k} and G_{f_i}, respectively, i.e., $G_{F_k} \coloneqq \max_{\theta \in \Theta} \|\nabla F_k(\theta)\|$, $k \in [K]$ and $G_{f_i} \coloneqq \max_{\theta \in \Theta} \|\nabla f_i(\theta)\|$, $i \in [m]$. Appealing to the mean value theorem, we can conclude that F_k is Lipschitz continuous, with constant G_{F_k}, $k \in [K]$.

The next assumption considers the behaviour of the cluster population losses F_k, $k \in [K]$.

Assumption 4. *For each $k \in [K]$, the population loss $F_k(\theta) = \mathbb{E}_{X_k \sim \mathcal{D}_k}[\ell(\theta; X_k)]$ is strongly convex, i.e., there exists a constant $\mu_{F_k} > 0$, such that, for all $\theta, \theta' \in \Theta$, we have*

$$F_k(\theta') \geq F_k(\theta) + \langle \nabla F_k(\theta), \theta' - \theta \rangle + \frac{\mu_{F_k}}{2} \|\theta - \theta'\|^2.$$

Remark 8. *In addition to requirements of Assumption 3, that implies convexity of each F_k, we require them to be even "better" behaved, i.e., we require them to be strongly convex. This will facilitate the rest of the analysis and allow for stronger bounds to be obtained.*

Assumption 5. *For each $k \in [K]$, there exists a neighborhood $U_k = \{\theta \in \Theta : \|\theta - \theta_k^\star\| \leq \rho_k\}$, where $\theta_k^\star = \arg\min_{\theta_k \in \Theta} F_k(\theta_k)$, $\rho_k > 0$, such that, for any fixed $x \in \mathcal{X}$, the loss ℓ has Lipschitz continuous Hessian, i.e., there exists a constant $P_k > 0$, such that, for any $\theta, \theta' \in U_k$, we have*

$$\|\nabla^2 \ell(\theta; x) - \nabla^2 \ell(\theta'; x)\| \leq P_k \|\theta - \theta'\|.$$

Remark 9. *Assumption 5 requires each population loss to be well-behaved in a neighborhood of the optimal model. Akin to Assumption C in [14], this assumption is required for averaging methods to work. We refer the reader to [14] and references therein, for an elaborate discussion on this requirement.*

Note that, for each $k \in [K]$, the set U_k is compact. Using the continuity of the Hessian on U_k, via Assumption 5, we can conclude that the Hessian of ℓ is bounded on all U_k, $k \in [K]$. Denote the bounding constants as H_k, $k \in [K]$, i.e.,
$H_k = \sup_{x \in \mathcal{X}, \theta \in \Theta} \|\nabla^2 \ell(\theta; x)\|$, $k \in [K]$.

Assumptions 2-5 are standard in the literature, e.g., the reader is referred to [14], [34] and the references therein. The authors in [14] require that the population loss is strongly convex only in a neighborhood of the optimal model, which is more relaxed than the requirement of Assumption 4. However, this condition is required in [34], whose results we apply to construct a high probability bound in the following sections.

Note that the formulation (8) implicitly assumes the knowledge of the true clustering structure. In reality, the distributions and their associated clustering structures are not known. Moreover, even the exact number of different distributions, K, is typically not available. Therefore, the formulation (8) is impossible to obtain and solve in practice. In what follows, we propose a method that is able to deal with these issues, by correctly identifying the true clusters and producing models that offer the same order-optimal MSE guarantees, as the models with knowledge of the true clustering structure, obtained by (8).

2.1 The method from [14]

The authors in [14] study the problem of finding the optimal model for (1), under the assumption that all the distributions are the same, i.e., $\mathcal{D}_k = \mathcal{D}$, for all $k \in [K]$. They propose the following two-step method, that requires only one round of communication:

1. Each user i obtains a local model $\widehat{\theta}_i$, by solving $\widehat{\theta}_i = \arg\min_{\theta \in \Theta} f_i(\theta)$ and sends it to the server.

2. The sever receives the local models and produces the final model by averaging, i.e., $\overline{\theta} = \frac{1}{m}\sum_{i=1}^{m} \widehat{\theta}_i$.

Assuming $n_i = n$, $i \in [m]$, for some $n \in \mathbb{N}$, the authors show that, when $n \geq m$, the method results in the order-optimal MSE, i.e., we have

$$\mathbb{E}\|\overline{\theta} - \theta^\star\|^2 = \mathcal{O}\left(\frac{1}{mn}\right).$$

Here, $\theta^\star = \arg\min_{\theta \in \Theta} F(\theta)$ is the optimal model for the entire population.

2.2 Convex clustering

Convex clustering is a well studied approach to clustering, e.g.. [35], [36] [37], wherein the clustering problem is formulated as a strongly convex optimization problem with group lasso regularization. As such, the method is guaranteed to have a unique solution and, moreover, does not require knowledge of the true number of clusters K. Formally, for a given dataset $A = \{a_1, \ldots, a_n\} \subset \mathbb{R}^d$, the problem of convex clustering is formulated as

$$\arg\min_{u_1,\ldots,u_n \in \mathbb{R}^d} \frac{1}{2}\sum_{i=1}^{n} \|a_i - u_i\|^2 + \lambda \sum_{i<j} \|u_i - u_j\|, \tag{9}$$

where $\lambda > 0$ is a tunable parameter. Let $\mathcal{V} = \{V_k\}_{k \in [K]}$ be a partition of A, such that $\cup_{k \in [K]} V_k = A$ and $V_k \cap V_l = \emptyset$, $k \neq l$. The authors in [37, Corollary 7] show that, if λ satisfies

$$\max_{k \in [K]} \frac{diam(V_k)}{|V_k|} \leq \lambda < \min_{\substack{k \neq l \\ k,l \in [K]}} \frac{\|c(V_k) - c(V_l)\|}{2n - |V_k| - |V_l|}, \tag{10}$$

the partition, i.e., the clustering, is recovered, in the sense that, for a mapping $\psi(x_i) = u_i^\star$, we have $u_i^\star = u_j^\star$, for all $i, j \in V_k$ and $u_i^\star \neq u_j^\star$, for all $i \in V_k$, $j \in V_l$, $k \neq l$. Here, $\{u_i^\star\}_{i=1}^n = \{u_i^\star(\lambda)\}_{i=1}^n$ is the (unique) optimal solution produced by (9), $diam(S) = \max\{\|x - y\| : x, y \in S\}$, is the diameter of a set $S \subset \mathbb{R}^d$, while $c(S) = \frac{1}{|S|}\sum_{x \in S} x$, is the centroid of S.

3 Algorithm design

In this section, we outline our one-shot algorithm for FL in heterogeneous environments. Subsection 3.1 describes the proposed one-shot method. Subsection 3.2 outlines some considerations when applying the method in practice.

3.1 The proposed method

In order to deal with the presence of multiple data distributions, we propose a method that works as follows:

1. Each user i obtains a local model $\widehat{\theta}_i$, by solving $\widehat{\theta}_i = \arg\min_{\theta_i \in \Theta} f_i(\theta_i)$ and sends it to the server.

2. The server receives the local models $\{\widehat{\theta}_i\}_{i=1}^m$, chooses a value $\lambda > 0$ (check Subsection 3.2 ahead) and runs the convex clustering algorithm (9), with the local models as inputs, resulting in K' clusters $\mathcal{C}' = \{C'_{k'}\}_{k' \in [K']}$.

3. The server then averages the local models according to the resulting clusters, i.e., for each obtained cluster $C'_{k'}$, $k' \in [K']$, the server performs $\overline{\theta}_{k'} = \frac{1}{|C'_{k'}|} \sum_{i \in C'_{k'}} \widehat{\theta}_i$.

4. The server then sends the models to each user, corresponding to their cluster assignment, i.e., each user $i \in C'_{k'}$ receives the model $\overline{\theta}_{k'}$.

Note that the main difference between the method in [14] and the proposed method is in step 2, where the server performs clustering of the models. This step is necessary, as we aim to identify the true clustering structure, and produce a model that maintains the guarantees of the clustered approach (8). We chose the convex clustering method, e.g., [37], [35], as it does not require knowledge of the exact number of clusters K. Note that, if knowledge of the number of clusters was available, a simpler algorithm, like K-means, e.g., [38], [39], or gradient clustering, e.g., [40], can be applied.

3.2 Practical considerations

The lower and upper bounds in the recovery condition (10) both depend on the recovered clustering, which in turns depends on the value of λ, via (9). This shows that (10) (and (12) ahead) can only be verified in "a posteriori" manner, after (9) is solved. Therefore, choosing an appropriate value of λ

can be difficult in practice. In this subsection we provide an algorithm that includes practical guidelines on choosing an appropriate value of the parameter λ, elaborating on step 2) from the previous subsection. The algorithm works as follows:

1. Each user i obtains a local model $\widehat{\theta}_i$, by solving $\widehat{\theta}_i = \arg\min_{\theta_i \in \Theta} f_i(\theta_i)$ and sends it to the server.

2. The server receives the local models $\{\widehat{\theta}_i\}_{i=1}^m$ and chooses a range of strictly increasing values of λ, $\{\lambda_1, \lambda_2, \dots, \lambda_N\}$, such that solving the convex clustering problem (9) results in the number of clusters K_{λ_i} satisfying $K_{\lambda_1} = m$ and $K_{\lambda_N} = 1$[2]. The server runs the convex clustering algorithm for each value of λ_i and verifies the condition (10).

   (a) If the condition (10) is verified for some values of λ_i, the server takes a value λ_i (and the associated clustering) such that the same number of clusters K_{λ_i} is recovered for the largest number of λ_i's verifying (10).

   (b) If the condition (10) is not verified for any value of λ_i, the server takes a value λ_i (and the associated clustering) such that the violation of the condition (10) is minimal, i.e., take a λ_i such that the difference between the associated lower and upper bounds is the smallest.

3. The server then averages the local models according to the resulting clusters $\mathcal{C}' = \{C'_{k'}\}_{k' \in [K']}$, i.e., for each obtained cluster $C'_{k'}$, $k' \in [K']$, the server performs $\overline{\theta}_{k'} = \frac{1}{|C'_{k'}|} \sum_{i \in C'_{k'}} \widehat{\theta}_i$.

4. The server then sends the models to each user, corresponding to their cluster assignment, i.e., each user $i \in C'_{k'}$ receives the model $\overline{\theta}_{k'}$.

The procedure in step 2 is known as "clusterpath", e.g., [36]. The intuition behind it is to either take a value of λ that results in a clustering that is the likeliest to be "true", or to take a value of λ for which the resulting clustering is the likeliest to be "close" to a true clustering. Note that in general, the recovery guarantees of convex clustering hold only when λ satisfies (10). However, in practice, convex clustering is known to perform well even when

[2]From the formulation of convex clustering (9), it is obvious that, for λ sufficiently small, the optimal solution is going to be $u_i^\star = a_i$, $i \in [m]$, i.e., $K_\lambda = m$. On the other hand, the authors in [41] show that, for λ sufficiently large, we have $K_\lambda = 1$. Hence, the choices of λ guaranteeing $K_\lambda = m$ and $K_\lambda = 1$ always exist.

the condition (10) is not met, e.g., [37] show that exact clustering can be recovered even for values of λ not in (10), with, e.g., [36], [42], validating the performance on real data, without the knowledge of (10).

4 Theoretical guarantees

In this section, we present the theoretical guarantees of the proposed method. In this section, for the sake of simplicity, we assume $n_i = n$, for all $i \in [m]$. Subsection 4.1 introduces some technical details and lemmas used in our work and presents the main result of the paper. Subsection 4.2 offers a detailed comparison of our method with the method from [2[illegible]]. Subsection 4.3 presents the MSE guarantees if the exact solutions of the local empirical risks are replaced by approximate ones.

4.1 Main result

Specializing (10) to our method, we can see that, for the clustering in step 3 of our approach to be correct, i.e., to have $K' = K$ and for all $k \in [K]$, a unique $k' \in [K']$ to satisfy $C'_{k'} = C_k$, we need the following condition satisfied

$$\max_{k\in[K]} \frac{diam(W_k)}{|W_k|} \leq \lambda < \min_{\substack{k\neq l\\ k,l\in[K]}} \frac{\|c(W_k) - c(W_l)\|}{2m - |W_k| - |W_l|}, \tag{11}$$

where $W_k = \{\widehat{\theta}_i : i \in C_k\}$, $k \in [K]$ is the cluster containing the ERMs of all users belonging to cluster C_k. Using the definitions of $diam(\cdot)$, $c(\cdot)$ and W_k, $k \in [K]$, we get that (11) is equivalent to

$$\max_{k\in[K],\, i,j\in C_k} \frac{\left\|\widehat{\theta}_i - \widehat{\theta}_j\right\|}{|C_k|} \leq \lambda < \min_{\substack{k\neq l\\ k,l\in[K]}} \frac{\|\overline{\theta}_k - \overline{\theta}_l\|}{2m - |C_k| - |C_l|}. \tag{12}$$

Remark 10. *Note that in general, condition* (11) *(and equivalently* (12)*) might not hold. However, in what follows, we consider all the possible outcomes and quantify the probability of* (12) *being satisfied.*

Next, we state some important results used in the rest of the section, from [14] and [34].

Lemma 1 (Theorem 3 in [34])**.** *Under Assumptions 1-4, for any $k \in [K]$, any $i \in C_k$ and any $0 < \delta < \frac{1}{2}$, $\epsilon > 0$, with probability at least $1 - 2\delta$, we* *have*

$$F_k(\widehat{\theta}_i) - F_k(\theta_k^\star) \leq \frac{16R^2LC(\epsilon,\delta)}{n} + \frac{8R\|\nabla f_i(\theta_k^\star)\| \log\frac{2}{\delta}}{n} + \frac{8LF_k(\theta_k^\star)\log\frac{2}{\delta}}{\mu_{F_k} n} + \left(8RL + G_{F_k} + \frac{4RLC(\epsilon,\delta)}{n}\right)\epsilon,$$

where $C(\epsilon,\delta) \coloneqq 2\left(\log\frac{2}{\delta} + d\log\frac{6R}{\epsilon}\right)$.

Lemma 2 (Theorem 1 in [14])**.** *Under Assumptions 1-5, for each* $k \in [K]$ *and* $\overline{\theta}_k = \frac{1}{|C_k|}\sum_{i\in C_k}\widehat{\theta}_i$*, we have*

$$\mathbb{E}\|\overline{\theta}_k - \theta_k^\star\|^2 \leq \frac{2E_k}{n|C_k|} + \frac{5}{\mu_{F_k}^2 n^2}\left(H_k^2 \log d + F_k\right)E_k + \mathcal{O}\left(|C_k|^{-1} n^{-2}\right) + \mathcal{O}(n^{-3}),$$

where $E_k \coloneqq \mathbb{E}\|\nabla^2 F_k(\theta_k^\star)^{-1}\nabla\ell(\theta_k^\star; X)\|^2$.

Note that the original results in [14] and [3[illegible]] concern the global population loss (1) and the corresponding empirical loss (3). These directly translate to each individual cluster in our framework, i.e., to each component in (8). Additionally, note that Lemma 2 assumes knowledge of the true clusters C_k, as the averaging is performed across the true clusters.

We are now ready to state the main result of the paper.

Theorem [illegible]. *Let Assumptions 1-5 hold. If the number of samples per user satisfies* $n \geq$ [illegible] *and moreover*

$$\frac{n}{\log n} > \frac{2M(2m - |C_{(K-1)}| - |C_{(K)}|)^2}{|C_{(K)}|^2(D-2\gamma)^2},$$

where $\beta \geq 1$ *and* $0 < \gamma < \frac{D}{2}$ *are tunable parameters, while* $M = M(\beta) = \max_{i,j\in C_k, k\in[K]} M_{ik} + M_{jk}$*, and for all* $i \in C_k$, $k \in [K]$

$$M_{ik} = \frac{64R^2L\left(\log 2 + d\log 6R + (d+1)\beta\right)}{\mu_{F_k}} + \frac{16LF_k(\theta_k^\star)(\log 2 + \beta)}{\mu_{F_k}^2} + \frac{16R\|\nabla f_i(\theta_k^\star)\|(\log 2 + \beta)}{\mu_{F_k}} + \frac{2G_{F_k} + 16RL\left(1 + \log 2 + d\log 6R + (d+1)\beta\right)}{\mu_{F_k}},$$

then, for any choice of $\lambda \in \left[\sqrt{\frac{2M\log n}{n}}, \frac{|C_{(K)}|(D-2\gamma)}{2m-|C_{(K-1)}|-|C_{(K)}|}\right)$*, we have that, for all* $k \in [K]$*, the models produced by the proposed method achieve the MSE*

$$\mathbb{E}\|\overline{\theta}_k - \theta_k^\star\|^2 \leq \frac{2E_k}{n|C_k|} + \frac{4K\widetilde{E}R^2}{n|C_{(K)}|\gamma^2} + \frac{4KR^2|\widetilde{C}|^2}{n^\beta}$$
$$+ \mathcal{O}\left(\frac{\log d}{n^2}\right) + \mathcal{O}\left(\frac{K\log d}{n^2\gamma^2}\right) + \mathcal{O}\left(\frac{1}{n^2|C_k|}\right)$$
$$+ \mathcal{O}\left(\frac{K}{n^2|C_{(K)}|\gamma^2}\right) + \mathcal{O}\left(\frac{1}{n^3}\right) + \mathcal{O}\left(\frac{K}{n^3\gamma^2}\right),$$

where $E_k = \mathbb{E}\|\nabla^2 F_k(\theta_k^\star)^{-1}\nabla\ell(\theta_k^\star; X)\|^2$*,* $\widetilde{E} = \frac{1}{K}\sum_{k\in[K]} E_k$ *and* $|\widetilde{C}|^2 = \frac{1}{K}\sum_{k\in[K]}|C_k|^2$*.*

Theorem 1 provides the MSE rate of the proposed method. If, in addition to the conditions of Theorem 1, we have $n \geq |C_{(1)}|$, then, for the choice of $\beta \geq 2$, we have that the MSE rate is dominated by the first two terms, i.e.,

$$\frac{2E_k}{n|C_k|} + \frac{4K\widetilde{E}R^2}{n|C_{(K)}|\gamma^2}. \tag{13}$$

Since $0 < \gamma < \frac{D}{2}$ is a tunable parameter, if $D > 2\sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}}$, we can choose $\gamma = \sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}}$, so that (13) becomes

$$\frac{2E_k}{n|C_k|} + \frac{4K\widetilde{E}R^2}{n|C_{(1)}|}.$$

This observation directly leads to the following corollary.

Corollary 1. *Let conditions of Theorem 1 hold. If additionally* $D > 2\sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}}$ *and* $n \geq |C_{(1)}|$*, then for the choices of* $\beta \geq 2$ *and* $\gamma = \sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}}$*, we have the following MSE, for all* $k \in [K]$

$$\mathbb{E}\|\overline{\theta}_k - \theta_k^\star\|^2 \leq \mathcal{O}\left(\frac{1}{n|C_k|}\right).$$

Corollary 1 shows that, if the populations are sufficiently separated, our method can achieve the order-optimal MSE rate for each cluster, provided that users have sufficient number of samples available. This rate is equivalent to the rate achieved by training a centralized learner on each cluster and as we discuss in Subsection 4.2 ahead, it is a stronger result compared to the current literature, where the convergence rate depends on the size of the smallest cluster, $|C_{(K)}|$. Remarkably, this is achieved with significant communication savings, requiring only a single communication round. Some remarks are now in order.

Remark 11. *The MSE guarantees in Lemma 2 are established without any requirements on the sample size n. This stems from the fact that the method from Lemma 2 can be seen as an oracle method that knows the true clustering structure. On the other hand, the sample size requirement in Theorem 1 stems from the fact that our method does not know the true clustering, hence a sufficiently large sample size that guarantees the true clustering can be recovered, is required.*

Remark 12. *Recall that the parameters $\beta \geq 1$ and $0 < \gamma < \frac{D}{2}$ are tunable. From Theorem 1, we can see that both parameters offer a trade-off between convergence speed and sample requirements. In particular, larger values of β and γ result in faster convergence, at the expense of higher sample requirements.*

Remark 13. *Recall the condition on the number of samples, given by*

$$\frac{n}{\log n} \geq \frac{2M(2m - |C_{(K-1)}| - |C_{(K)}|)^2}{|C_{(K)}|^2(D-2\gamma)^2},$$

where $\beta \geq 1$ and $0 < \gamma < \frac{D}{2}$ are tunable parameters, while $M = \max_{i,j\in C_k, k\in[K]} M_{ik} + M_{jk}$, and for any $i \in C_k$, $k \in [K]$

$$\begin{aligned} M_{ik} = & \frac{64R^2L\left(\log 2 + d\log 6R + (d+1)\beta\right)}{\mu_{F_k}} \\ & + \frac{16LF_k(\theta_k^\star)(\log 2 + \beta)}{\mu_{F_k}^2} + \frac{16R\|\nabla f_i(\theta_k^\star)\|(\log 2 + \beta)}{\mu_{F_k}} \\ & + \frac{2G_{F_k} + 16RL\left(1 + \log 2 + d\log 6R + (d+1)\beta\right)}{\mu_{F_k}}. \end{aligned}$$

We can identify three components of the condition that quantify the complexity of different aspects of the system:

- *M - quantifies the difficulty of the learning problems, as it depends on problem parameters, such as the dimension of the parameter space d,* *the smoothness and strong convexity parameters L, G_{F_k}, μ_{F_k}, etc. It also depends on the population minimal value $F_k(\theta_k^\star)$ and the proximity of the population and local empirical risks in the form of $\|\nabla f_i(\theta_k^\star)\|$. Hence, an easier learning problem implies smaller M.*

- $\frac{(2m-|C_{(K)}|-|C_{(K-1)}|)^2}{|C_{(K)}|^2}$ *- quantifies how well balanced the clusters are. For example, when the clusters are well balanced, so that $|C_k| = \frac{m}{K}$, for all $k \in [K]$, we have $\frac{(2m-|C_{(K)}|-|C_{(K-1)}|)^2}{|C_{(K)}|^2} = 4(K-1)^2$, while in the extreme case of $|C_{(K)}| = |C_{(K-1)}| = 1$, we have $\frac{(2m-|C_{(K)}|-|C_{(K-1)}|)^2}{|C_{(K)}|^2} = 4(m-1)^2$. As $K \leq m$, this again shows that balanced clusters are favored over unbalanced ones.*

- $(D-2\gamma)^{-2}$ *- quantifies the difficulty of the clustering problem. If D is smaller, population optima corresponding to different populations are closer to one another and it is more difficult to cluster the local ERMs correctly, hence requiring more samples. On the other hand, for larger D, the clustering problem becomes easier and we require fewer samples per user for correct clustering.*

Remark 14. *Recall the condition on D in Corollary 1, $D > 2\sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}}$. When the clusters are well balanced, so that $|C_k| = \frac{m}{K}$, $k \in [K]$, our method can achieve order-optimal rates if the minimal separation between population optima of different clusters is $D > 2$, i.e., independent of any problem parameters. On the other hand, in the worst case, we can have $D > 2\sqrt{m-1}$, if there are only two clusters C_1, C_2, such that $|C_1| = 1$, $|C_2| = m-1$.*

Proof of Theorem [illegible]. We start by noting that, for any event Ψ, we have

$$\mathbb{E}\|\widehat{\theta}_k - \theta_k^\star\|^2 = \mathbb{E}\|\widehat{\theta}_k - \theta_k^\star\|^2 \mathbb{I}_\Psi + \mathbb{E}\|\widehat{\theta}_k - \theta_k^\star\|^2 \mathbb{I}_{\Psi^c}, \tag{14}$$

where $\mathbb{I}_\Psi$ is the indicator random variable. We now proceed to define a specific event Ψ and establish the resulting bounds.

Applying Lemma 1 for the choice of $\delta = \epsilon = \frac{1}{n^\beta}$, for some $\beta > 0$, we get that, for all $k \in [K]$ and all $i \in C_k$, we have

$$\begin{aligned}\|\widehat{\theta}_i - \theta_k^\star\|^2 \le \frac{32R^2LC(\epsilon,\delta)}{n\mu_{F_k}} + \frac{16LF_k(\theta_k^\star)(\log 2 + \beta\log n)}{n\mu_{F_k}^2} \\ + \frac{16R\|\nabla f_i(\theta_k^\star)\|(\log 2 + \beta \log n)}{n\mu_{F_k}} \\ + \frac{\left(16RL + 2G_{F_k} + \frac{8RLC(\epsilon,\delta)}{n}\right)}{n^\beta \mu_{F_k}},\end{aligned} \tag{15}$$

with probability at least $1-\frac{2}{n^\beta}$, where $C(\epsilon,\delta) = 2\left(\log 2 + d\log 6R + (d+1)\beta\log n\right)$. Here, we used strong convexity of F_k, which implies

$$\|\widehat{\theta}_i - \theta_k^\star\|^2 \le \frac{2}{\mu_{F_k}}\left(F_k(\widehat{\theta}_i) - F_k(\theta_k^\star)\right).$$

Note that, for $\beta \ge 1$ and $n \ge 3$, the dominating term in (15), in terms of the number of samples n, is of the order $\mathcal{O}\left(\frac{\log n}{n}\right)$. We can therefore upper-bound the right-hand side of (15) by $\frac{M_{ik}\log n}{n}$, where M_{ik}, $i \in C_k$, $k \in [K]$ is defined as follows

$$\begin{aligned}M_{ik} = \frac{64R^2L(\log 2 + d\log 6R + (d+1)\beta)}{\mu_{F_k}} \\ + \frac{16LF_k(\theta_k^\star)(\log 2 + \beta)}{\mu_{F_k}^2} + \frac{16R\|\nabla f_i(\theta_k^\star)\|(\log 2 + \beta)}{\mu_{F_k}} \\ + \frac{2G_{F_k} + 16RL\left(1 + \log 2 + d\log 6R + (d+1)\beta\right)}{\mu_{F_k}}.\end{aligned}$$

As $\frac{M_{ik}\log n}{n}$ is an upper bound on the right-hand side of (15), we can therefore conclude that, for any $i \in C_k$, $k \in [K]$

$$\mathbb{P}\left(\|\widehat{\theta}_i - \theta_k^\star\|^2 \le \frac{M_{ik}\log n}{n}\right) \ge 1 - \frac{2}{n^\beta}. \tag{16}$$

Next, define the events

$$\begin{aligned}\Sigma_{ij} &= \left\{\omega : \|\widehat{\theta}_i - \widehat{\theta}_j\|^2 \le \frac{2(M_{ik} + M_{jk})\log n}{n}\right\}, \\ \Upsilon_i &= \left\{\omega : \|\widehat{\theta}_i - \theta_k^\star\|^2 \le \frac{M_{ik}\log n}{n}\right\}.\end{aligned}$$

for all $i, j \in C_k$, $i \neq j$, $k \in [K]$. Noting that

$$\|\widehat{\theta}_i - \widehat{\theta}_j\|^2 \leq 2\|\widehat{\theta}_i - \theta_k^\star\|^2 + 2\|\widehat{\theta}_j - \theta_k^\star\|^2,$$

we can conclude that, for all $i, j \in C_k$, $k \in [K]$

$$\mathbb{P}\left(\Upsilon_i \cap \Upsilon_j\right) \leq \mathbb{P}\left(\Sigma_{ij}\right). \tag{17}$$

For $\Sigma = \cap_{i,j\in C_k, i\neq j, k\in[K]}\Sigma_{ij}$, we then get the following bound

$$\begin{aligned}
\mathbb{P}(\Sigma) &\geq 1 - \sum_{\substack{i\neq j\\ i,j\in C_k\\ k\in[K]}} \mathbb{P}(\Sigma_{ij}^{\mathsf{c}}) \geq 1 - \sum_{\substack{i\neq j\\ i,j\in C_k\\ k\in[K]}} \mathbb{P}\left((\Upsilon_i \cap \Upsilon_j)^{\mathsf{c}}\right) \\
&\geq 1 - \sum_{\substack{i\neq j\\ i,j\in C_k\\ k\in[K]}} \frac{4}{n^\beta} \geq 1 - \frac{2}{n^\beta}\sum_{k\in[K]} |C_k|\left(|C_k| - 1\right) \\
&\geq 1 - \frac{2K|\widetilde{C}|^2}{n^\beta},
\end{aligned}$$

where $|\widetilde{C}|^2 = \frac{1}{K}\sum_{k\in[K]}|C_k|^2$, the first inequality follows from the union bound, the second inequality follows from (17), while the third inequality follows from the union bound and (16). Next, for any $k, l \in [K]$, we have that

$$\|\overline{\theta}_k - \overline{\theta}_l\| \geq \|\theta_k^\star - \theta_l^\star\| - \|\overline{\theta}_k - \theta_k^\star\| - \|\overline{\theta}_l - \theta_l^\star\|. \tag{18}$$

For any $\gamma > 0$ and any $k \in [K]$, applying Chebyshev's inequality and Lemma 2, we get the following bound

$$\begin{aligned}
\mathbb{P}\big(\|\overline{\theta}_k - \theta_k^\star\| > \gamma\big) &\leq \frac{\mathbb{E}\|\overline{\theta}_k - \theta_k^\star\|^2}{\gamma^2} \leq \frac{2E_k}{n|C_k|\gamma^2} \\
&+ \frac{5\left(H_k^2 \log d + E_k\right)E_k}{\mu_{F_k}^2 n^2\gamma^2} + \mathcal{O}\left(\frac{1}{|C_k|n^2\gamma^2}\right) + \mathcal{O}\left(\frac{1}{n^3\gamma^2}\right).
\end{aligned} \tag{19}$$

Define the event $\Lambda = \cap_{k\in[K]} \left\{\omega : \|\overline{\theta}_k - \theta_k^\star\| \leq \gamma\right\}$. We then have

$$\begin{aligned}
\mathbb{P}(\Lambda) &\geq 1 - \sum_{k\in[K]} \mathbb{P}\left(\|\overline{\theta}_k - \theta_k^\star\| > \gamma\right) \\
&\geq 1 - \sum_{k\in[K]} \left(\frac{2E_k}{n|C_k|\gamma^2} + \frac{5\left(H_k^2 \log d + E_k\right) E_k}{\mu_{F_k}^2 n^2 \gamma^2}\right. \\
&\quad \left. + \mathcal{O}\left(\frac{1}{|C_k| n^2 \gamma^2}\right) + \mathcal{O}\left(\frac{1}{n^3\gamma^2}\right)\right) \\
&\geq 1 - \frac{2K\widetilde{E}}{n|C_{(K)}|\gamma^2} - \frac{5K\left(\widetilde{H}^2 \log d + \widetilde{E}\right) E_{\max}}{\mu_{F_{\min}}^2 n^2 \gamma^2} \\
&\quad - \Omega\left(\frac{K}{|C_{(K)}| n^2 \gamma^2}\right) - \Omega\left(\frac{K}{n^3\gamma^2}\right),
\end{aligned}$$

where $\widetilde{E} = \frac{1}{K}\sum_{k\in[K]} E_k$, $\widetilde{H}^2 = \frac{1}{K}\sum_{k\in[K]} H_k^2$, $E_{\max} = \max_{k\in[K]} E_k$ and $\mu_{F_{\min}} = \min_{k\in[K]} \mu_{F_k}$. Recall that $D = \min_{k\neq l} \|\theta_k^\star - \theta_l^\star\|$. Applying (18), we then have that on Λ, for any $k, l \in [K]$

$$\|\overline{\theta}_k - \overline{\theta}_l\| \geq D - 2\gamma, \tag{20}$$

which is valid for any $\gamma < \frac{D}{2}$. Next, notice that on Σ, for all $i, j \in C_k$, $k \in [K]$, we have

$$\|\widehat{\theta}_i - \widehat{\theta}_j\| \leq \sqrt{\frac{2(M_{ik} + M_{jk})\log n}{n}}. \tag{21}$$

Plugging (20) and (21) in (12), we get that the true clustering can be recovered if

$$\sqrt{\frac{2M\log n}{n}} < \frac{|C_{(K)}|(D - 2\gamma)}{2m - |C_{(K-1)}| - |C_{(K)}|}, \tag{22}$$

where $M = \max_{i,j\in C_k,\, k\in[K]}(M_{ik} + M_{jk})$. For (22) to hold we need the number of samples per user to be such that

$$\frac{n}{\log n} > \frac{2M(2m - |C_{(K-1)}| - |C_{(K)}|)^2}{|C_{(K)}|^2 (D - 2\gamma)^2}. \tag{23}$$

We then have that on $\Psi = \Sigma \cap \Lambda$, if the number of samples per user satisfies (23), the true clustering can be recovered and Lemma 2 applies to our method. On the other hand, we have

$$\begin{aligned}
\mathbb{P}(\Psi) &\geq \mathbb{P}(\Sigma) + \mathbb{P}(\Lambda) - 1 \\
&\geq 1 - \frac{2K\widetilde{E}}{n|C_{(K)}|\gamma^2} - \frac{5K\left(\widetilde{H}^2 \log d + \widetilde{E}\right)E_{\max}}{\mu_{F_{\min}}^2 n^2\gamma^2} \\
&\quad - \Omega\left(\frac{K}{|C_{(K)}|n^2\gamma^2}\right) - \Omega\left(\frac{K}{n^3\gamma^2}\right) - \frac{2K|\widetilde{C}|^2}{n^\beta},
\end{aligned}$$

which implies

$$\begin{aligned}
\mathbb{P}(\Psi^{\mathsf{c}}) \leq \frac{2K\widetilde{E}}{n|C_{(K)}|\gamma^2} + \frac{2K|\widetilde{C}|^2}{n^\beta} + \mathcal{O}\left(\frac{K}{|C_{(K)}|n^2\gamma^2}\right) \\
+ \frac{5K\left(\widetilde{H}^2 \log d + \widetilde{E}\right)E_{\max}}{\mu_{F_{\min}}^2 n^2\gamma^2} + \mathcal{O}\left(\frac{K}{n^3\gamma^2}\right).
\end{aligned}$$

Combining everything in (14), we finally get

$$\begin{aligned}
&\mathbb{E}\|\overline{\theta}_k - \theta_k^\star\|^2 \leq \mathbb{E}\|\widehat{\theta}_k - \theta_k^\star\|^2 \mathbb{1}_\Psi + R^2\mathbb{P}(\Psi^{\mathsf{c}}) \\
&\leq \frac{2E_k}{n|C_k|} + \frac{2K\widetilde{E}R^2}{n|C_{(K)}|\gamma^2} + \frac{2KR^2|\widetilde{C}|^2}{n^\beta} + \mathcal{O}\left(\frac{1}{|C_k|n^2}\right) \\
&+ \frac{5E_k}{\mu_{F_k}^2 n^2}\left(H_k^2 \log d + E_k\right) + \frac{5R^2KE_{\max}}{\mu_{F_{\min}}^2 n^2\gamma^2}\left(\widetilde{H}^2 \log d + \widetilde{E}\right) \\
&+ \mathcal{O}\left(\frac{K}{|C_{(K)}|n^2\gamma^2}\right) + \mathcal{O}\left(\frac{1}{n^3}\right) + \mathcal{O}\left(\frac{K}{n^3\gamma^2}\right),
\end{aligned}$$

for n satisfying (23). □

4.2 Comparison with order-optimal CFL methods

In this section we compare the results from Theorem 1 with the guarantees of other CFL methods. As discussed in the Introduction, many methods for CFL have been proposed, with various requirements and guarantees. For example, we can split the methods into the ones requiring knowledge of the number of clusters K, e.g., [28], [29], [30] and the ones not requiring it, e.g., [31], [32]. On the other hand, we can split them into the ones that estimate the true clustering iteratively, e.g., [28], [29], [31], [32] and the ones that perform clustering only ones during training, [30]. While all the proposed methods offer certain advantages (as illustrated by the previous classification), we specifically compare our method with two methods, namely

Iterative Federated Clustering Algorithm (IFCA), from [29], and the method from [30]. The main reasons for comparing with these specific methods are: 1) both methods analyze their performance in terms of statistical guarantees; 2) both methods are order-optimal, up to logarithmic factors and 3) both methods derive explicit requirements for the number of communication rounds. In what follows, due to some similarities of the methods and their performances, we will provide a detailed outline of IFCA, while highlighting where [30] differs significantly.

IFCA is an iterative algorithm for CFL that alternates between the following two steps: inferring cluster membership and updating the models. To that end, IFCA is initialized by first producing K different models $\{\theta_k^0\}_{k\in[K]}$, where the superscript denotes the iteration counter. The method then proceeds as follows, for $t = 0, \dots, T-1$:

1. The server broadcasts $\{\theta_k^t\}_{k\in[K]}$ to each user.
2. Each user evaluates the models on their local data and chooses the model $\theta_{(i)}^t$, where $(i) = \arg\min_{k\in[K]} f_i(\theta_k^t)$.
3. Each user computes the local stochastic gradient $g_i^t = \widetilde{\nabla} f_i(\theta_{(i)}^t)$, evaluated at $\theta_{(i)}^t$. Users send the gradients back to the server, along with a one-hot encoding vector $s_i \in \mathbb{R}^K$, such that $s_{ij} = \begin{cases} 1, & (i) = j \\ 0, & (i) \neq j \end{cases}$, notifying the server which model was updated by user i.
4. The server forms clusters of users that updated specific models, based on the received tokens $\{s_i\}_{i\in[m]}$ and performs the model update, i.e., $\theta_k^{t+1} = \theta_k^t - \frac{\alpha}{|C_k^t|}\sum_{i\in C_k^t} g_i^t$, where $\alpha > 0$ is the step-size, while $C_k^t = \{i \in [m] : s_{ik} = 1\}$ is the cluster of users that updated model k at iteration t.

On the other hand, the method in [30] can be seen as a modular method, as it depends on the following three steps:

1. Each user trains the local ERMs and sends them to the server.
2. The server performs a clustering procedure.
3. A FL algorithm is run on the resulting clusters for T iterations to produce the final models.

From the algorithm above, we can see that both IFCA and the method [30] require knowledge of the true number of distributions K (or at least a good estimate), which is typically unavailable or would require running a separate learning algorithm in practice (e.g., community detection). Secondly, both require T rounds of communication, whereas our method requires a single round of server-user communication. Comparing to our algorithm, IFCA alleviates the computational requirements on the server side, by only requiring the server to average the received models. On the other hand, our algorithm assumes that the server has enough computation resources to run the convex clustering algorithm and perform inference on the underlying clustering structure, significantly decreasing the communication cost.

Assumptions. Similarly to our algorithm, IFCA assumes that the population risks F_k, $k \in [K]$ are L-smooth and μ_F-strongly convex. The method [30] requires a stronger assumption - namely, that the loss function ℓ is strongly convex. Additionally, IFCA assumes bounded variance of ℓ with respect to all $\mathcal{D}_k$, $k \in [K]$, i.e.,

$$\mathbb{E}_{X \sim \mathcal{D}_k}\left[(\ell(\theta; X) - F_k(\theta))^2\right] \leq \eta^2,$$

for some $\eta > 0$. Intuitively, this assumption is made to ensure that the empirical loss f_i of user $i \in C_k$ stays close to the true population loss F_k, enabling clustering inference via the local loss. An additional assumption made by IFCA, that is required for the convergence of the algorithm is *sufficiently close initialization*, i.e., for all $k \in [K]$, the authors require

$$\|\theta_k^0 - \theta_k^\star\| \leq \left(\frac{1}{2} - \alpha_0\right) D \sqrt{\frac{\mu_{F_{\min}}}{L}},$$

where $\alpha_0 \in (0, \frac{1}{2})$ is a tunable parameter that determines the proximity of the initialization to the true population optima. Note that such an assumption is quite strong, as it requires $\|\theta_k^0 - \theta_k^\star\| < \frac{1}{2}D$, for all $k \in [K]$. In order to find such an initialization, the knowledge of underlying clusters, as well as D, would have to be available. The method [30], like our method, does not require such an assumption. IFCA requires three further assumptions:

1. $|C_{(K)}| \gtrsim \log(mn)^3$, i.e., the size of the smallest cluster has to grow at least logarithmically in the number of total samples available in the system;

[3]Note that the authors in [29] use $n' = \frac{n}{2T}$ in their theoretical analysis, i.e., they require that each user contains $n = 2Tn'$ samples and all conditions in the original paper are expressed in terms of n'. However, for the sake of simplicity, we will represent the conditions in terms of n, effectively reducing the original sample size requirement by a factor of $2T$.

2. $n \gtrsim \frac{K\eta^2}{\alpha_0^2 \mu_{F_{\min}}^2 D^4}$, i.e., each user contains a sufficient number of samples;

3. $D \geq \widetilde{\mathcal{O}}\left(\max\left\{\alpha_0^{-2/5} n^{-1/5}, \alpha_0^{-1/3} m^{-1/6} n^{-1/3}\right\}\right)$, i.e., the population optimal models across different populations are sufficiently well separated.

Here, the operator $x \gtrsim y$ indicates the existence of global constant C that does not depend on the problem parameters, such that $x \geq Cy$ (the operator $x \lesssim y$ is defined similarly), while $\widetilde{\mathcal{O}}(\cdot)$ hides logarithmic factors that do not depend on m and n. On the other hand, both our method and the method [30] do not require any assumptions on the separation parameter D or on the size of the smallest cluster $|C_{(K)}|$, effectively covering the cases for which IFCA might fails, i.e., highly unbalanced clusters, e.g., $|C_{(K)}| = \mathcal{O}(1)$ and small separation between optimal models of different populations. Comparing the requirements on the number of samples of our method, IFCA and [30], we have, respectively,

$$\frac{n}{\log n} > \frac{2M(2m - |C_{(K-1)}| - |C_{(K)}|)^2}{|C_{(K)}|^2(D - 2\text{[illegible]})^2},$$

$$n \gtrsim \frac{K\eta^2}{\alpha_0^2 \mu_{F_{\min}}^2 D^4},$$

$$n \gtrsim \frac{G_{F_{\max}}^2 L \log m}{\mu_{F_{\min}}^2},$$

where $G_{F_{\max}} = \max_{k \in [K]} G_{F_k}$. We can first see that that for our method, the requirement is expressed in terms of $n/\log n$, which, for $n \geq 3$ is always more relaxed than placing requirements directly on n. Comparing the right-hand sides of the inequalities, we can see that the dependence of IFCA on K is much better, as (recall Remark 13) the term $\frac{(2m-|C_{(K-1)}|-|C_{(K)}|)^2}{|C_{(K)}|^2}$ evaluates to $4(K-1)^2$ in the best case, while being $4(m-1)^2$ in the worst case. The method [30] depends logarithmically on m. For $D > 1$, the dependence of IFCA is again better, while, for $D < 1$, our method has a much better dependence. Finally, the dependence on the problem parameters, encapsulated in M, are again better for IFCA, as typically one would expect $M > \frac{\eta^2}{\mu_{F_{\min}}^2}$. However, we stress that IFCA and [30] are iterative algorithms, allowing for multiple rounds of communication, whereas the method we propose is a one-shot method. Therefore, the higher requirements on the number of samples are to be expected, but uncover regimes in which communicating beyond one round to achieve order-optimality is redundant. Additionally, our method does not require knowledge of K, while both IFCA and [30] assume the knowledge of the true value of K. All of these facts lead to less strict requirements of IFCA on the number of samples per user, with respect to different problem parameters (in some cases).

Guarantees. The guarantees of IFCA are given it terms of high probability bounds, while our guarantees, expressed in terms of the MSE, are sharper. IFCA provides the following guarantee (Corollary 2 in [29]): after $T = \frac{8mL}{|C_{(K)}|\mu_{F_{\min}}} \log\left(\frac{2D}{\varepsilon}\right)$ communication rounds, with probability at least $1-\delta$

$$\|\theta_k^T - \theta_k^\star\| \leq \varepsilon,$$

where

$$\begin{aligned}\varepsilon \lesssim & \frac{\sigma_{\max} K L \log(mn) \left({}^{m}/{}_{|C_{(K)}|}\right)^2}{\mu_{F_{\min}} \delta \sqrt{n|C_{(K)}|}} + \mathcal{O}\left(\frac{1}{n\sqrt{m}}\right) \\ & + \frac{\eta^2 L^2 \left({}^{m}/{}_{|C_{(K)}|}\right)^2 K \log(mn)}{\mu_{F_{\min}}^4 \delta D^4 n}\end{aligned} \tag{24}$$

We can see that, assuming $n \geq |C_{(1)}|$, the dominating term in (24) becomes

$$\|\theta_k^T - \theta_k^\star\| = \mathcal{O}\left(\frac{\log(mn)}{\sqrt{n|C_{(K)}|}}\right),$$

for all $k \in [K]$, which is almost order-optimal, up to a logarithmic factor and dependence on the smallest cluster size. The guarantees of [30] are similar, i.e., via Theorem 1 in [30], we have that: after $T = \mathcal{O}\left(\frac{L+\mu_{F_{\max}}}{\mu_{F_{\min}}} \log\left(\frac{\mu_{F_{\max}}}{2\varepsilon}\right)\right)$ communication rounds, with high probability, for all $k \in [K]$

$$\|\theta_k^T - \theta_k^\star\| \leq \mathcal{O}\left(\frac{\log mn}{\sqrt{n|C_k|}}\right).$$

On the other hand, from Theorem 1, for $n \geq |C_{(1)}|$, we have

$$\mathbb{E}\|\overline{\theta}_k - \theta_k^\star\| = \mathcal{O}\left(\frac{1}{\sqrt{n|C_{(K)}|}}\right),$$

for all $k \in [K]$, which is almost order-optimal, with the dependence on the smallest cluster size. Therefore, we can see that our method removes the logarithmic dependence on the total number of samples, of both IFCA and [30], while simultaneously reducing the communication cost by a factor of $\mathcal{O}\left(\frac{\kappa}{p}\log\left(\frac{2D}{\varepsilon}\right)\right)$ with respect to IFCA (and similar with respect to [30]), where $\kappa = \frac{L}{\mu_{F_{\min}}} \geq 1$ is the condition number, while $p = \frac{|C_{(K)}|}{m} < 1$ is the fraction of users belonging to the smallest cluster, reflecting the difficulty of the clustering problem.

However, we can see that Theorem 1 provides guarantees in terms of the size of the smallest cluster, $|C_{(K)}|$, while Theorem 1 in [30] provides the guarantees in terms of the true cluster size $|C_k|$, for each $k \in [K]$. Applying Corollary 1, for $D > 2\sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}}$, our method matches the dependence on individual cluster sizes of [30], while removing the logarithmic dependence on the total number of samples, thus achieving the order-optimal rate

$$\mathbb{E}\|\overline{\theta}_k - \theta_k^\star\| = \mathcal{O}\left(\frac{1}{\sqrt{n|C_k|}}\right),$$

for all $k \in [K]$, while still providing a reduction in communication cost by a factor of $\mathcal{O}\left(\frac{\kappa}{p}\log\left(\frac{2D}{\varepsilon}\right)\right)$. Hence, we can see that our method provides order-optimal convergence guarantees, improving on the guarantees of both IFCA and [30] by a factor logarithmic in the total number of samples in the system. Remarkably, this is achieved while simultaneously reducing the communication cost by a factor of $\mathcal{O}\left(\frac{\kappa}{p}\log\left(\frac{2D}{\varepsilon}\right)\right)$ and without requiring any knowledge of the underlying structure, while both IFCA and [30] assume knowledge of K.

4.3 Inexact ERMs

In this section we consider replacing the ERM model $\widehat{\theta}_i = \arg\min_{\theta_i \in \Theta} f_i(\theta_i)$, $i \in [m]$, by an *inexact estimate*, i.e., an estimate $\widetilde{\theta}_i \in \Theta$, such that

$$\|\widetilde{\theta}_i - \widehat{\theta}_i\| \leq \varepsilon, \tag{25}$$

for some $\varepsilon > 0$. To that end, we need an additional assumption on the strong convexity of the empirical losses f_i, $i \in [m]$.

Assumption 6. *For all $i \in [m]$ the empirical loss f_i is strongly convex, i.e., there exists a constant $\mu_{f_i} > 0$, such that, for all $\theta, \theta' \in \Theta$, we have*

$$f_i(\theta') \geq f_i(\theta) + \langle \nabla f_i(\theta), \theta' - \theta \rangle + \frac{\mu_{f_i}}{2}\|\theta - \theta'\|^2.$$

Denote by $\mu_f = \min_{i\in[m]} \mu_{f_i}$.

Remark 15. *Note that in general, Assumption 6 allows for the loss function* ℓ *to be convex, as long as the average across local samples,* $f_i(\theta) = \frac{1}{n}\sum_{j=1}^{n} \ell(\theta; x_{ij})$ *is strongly convex.*

Assumption 6 allows for each user to apply iterative solvers, to obtain parameters $\widehat{\theta}_i$ that satisfy (25). A standard choice is the stochastic gradient descent (SGD) algorithm [43]. SGD follows a simple update rule, given by

$$\theta^{t+1} = \theta^t - \eta^t g^t,$$

where θ^t is the estimate of the parameter of interest at iteration t, η^t is the step-size and g^t is a stochastic gradient, evaluated at θ^t.

SGD can be implemented in both the online setting, where users only have access to a single stochastic gradient at a time and in the batch setting, where users have access to the entire local dataset. Additionally, SGD offers the most general guarantees with respect to the mini-batch size and can be implemented even with a mini-batch size of 1. We discuss at the end of the section how different assumptions can allow for the implementation of more efficient algorithms, in terms of the local iteration complexity per user. Next, we state an additional assumption on the stochastic gradients of f_i.

Assumption 7. *For each* $i \in [m]$ *and all* $\theta \in \Theta$, *stochastics gradient* g_i *of* f_i, *evaluated at* θ, *are unbiased, i.e.,* $\mathbb{E}\left[g_i\right] = \nabla f_i(\theta)$. *Additionally, the stochastic gradients have bounded variance, i.e., there exists a* $\sigma_i > 0$, *such that for all* $\theta \in \Theta$, *we have*

$$\mathbb{E}\|g_i - \nabla f_i(\theta)\|^2 \leq \sigma_i^2.$$

Remark 16. *Assumption 7 is standard in the analysis of stochastic algorithms, e.g., [44], [45], [46].*

Remark 17. *Recall the discussion in Section 2 and Remarks 5-7, that imply bounded gradients of* f_i, *with constant* G_{f_i}. *Combining with Assumption 7, it then follows that, for all* $\theta \in \Theta$

$$\mathbb{E}\|g_i\|^2 \leq 2\mathbb{E}\|g_i - \nabla f_i(\theta)\|^2 + 2\|\nabla f_i(\theta)\|^2 \leq 2\sigma_i^2 + 2G_{f_i}^2.$$

Define $\Gamma_i^2 := 2\sigma_i^2 + 2G_{f_i}^2$, $i \in [m]$ and denote by $\Gamma^2 = \max_{i\in[m]} \Gamma_i^2$. We now state two well-known result on the convergence of SGD from [44], used in the rest of the section.

Lemma 3 (Lemma 1 in [44]). *Under Assumptions 2, 3, 6 and 7, for all $i \in [m]$, if we set the step-size rule of SGD as $\eta^t = \frac{1}{\mu_{f_i} t}$, it holds for any $T \geq 1$ and any $i \in [m]$ that*

$$\mathbb{E}\|\theta_i^T - \widehat{\theta}_i\|^2 \leq \frac{4\Gamma_i^2}{\mu_{f_i}^2 T}.$$

Lemma 4 (Lemma 2 in [44]). *Let Assumptions 2, 3, 6 and 7 hold and let $\|g^t\|^2 \leq \Gamma^2$, with probability 1. Then, for all $i \in [m]$ and any $\delta \in (0, 1/e)$, $T \geq 4$, if we set the step-size rule of SGD as $\eta^t = \frac{1}{\mu_{f_i} t}$, it holds with probability $1 - \delta$, for any $t \in \{8, \ldots, T-1, T\}$ and any $i \in [m]$, that*

$$\|\theta_i^t - \widehat{\theta}_i\|^2 \leq \frac{12\Gamma^2}{\mu_{f_i}^2 t} + 8G(121G + 1)\frac{\log(\log(t)/\delta)}{t}.$$

We are now ready to state and prove counterparts of Lemmas 1 and 2, when an inexact ERM estimator is used.

Lemma 5. *Let Assumptions 1-4, 6 and 7 hold and $\|g^t\|^2 \leq \Gamma^2$ with probability 1. If each user runs SGD locally for T iterations, with the step-size rule $\eta^t = \frac{1}{\mu_f t}$, to produce θ_i^T, $i \in [m]$ and T is chosen such that $T \geq 15$ and $\frac{T}{\log\log(T)} \geq \left(\frac{12\Gamma^2}{\mu_f^2} + 8G(121G+1)(1+\log\frac{1}{\delta})\right)\frac{1}{\varepsilon^2}$, then for any $k \in [K]$, any $i \in C_k$ and any $\epsilon > 0$, $0 < \delta < \frac{1}{e}$, with probability at least $1 - 3\delta$, we have, for any $i \in C_k$, $k \in [K]$*

$$\begin{aligned} F_k(\theta_i^T) - F_k(\theta_k^\star) \leq{}& \frac{16R^2LC(\epsilon,\delta)}{n} + \frac{8R\|\nabla f_i(\theta_k^\star)\|\log\frac{2}{\delta}}{n} \\ &+ \frac{8LF_k(\theta_k^\star)\log\frac{2}{\delta}}{\mu_{F_k} n} + \left(8RL + G_{F_k} + \frac{4RLC(\epsilon,\delta)}{n}\right)\epsilon \\ &+ \varepsilon G_{F_k}, \end{aligned}$$

where $C(\epsilon,\delta) := 2\left(\log\frac{2}{\delta} + d\log\frac{6R}{\epsilon}\right)$.

Proof. For any $\theta \in \Theta$, any $k \in [K]$ and any $i \in C_k$, we have

$$F_k(\theta) - F_k(\theta_k^\star) \leq \left|F_k(\theta) - F_k(\widehat{\theta}_i)\right| + F_k(\widehat{\theta}_i) - F_k(\theta_k^\star). \tag{26}$$

We can bound the second term on the right hand side of (26) using Lemma 1. To bound the first term, we use Lipschitz continuity of F_k (recall the discussion in Section 2), to get

$$\left|F_k(\theta) - F_k(\widehat{\theta}_i)\right| \leq G_{F_k}\|\theta - \widehat{\theta}_i\|. \tag{27}$$

Next, applying Lemma 4, we have that, with probability at least $1-\delta$

$$\|\theta_i^T - \widehat{\theta}_i\|^2 \le \frac{12\Gamma^2}{\mu_f^2 T} + 8G(121G+1)\frac{\log\left(^{\log(T)}/\delta\right)}{T}.$$

As $T \ge 15$, we can use the following upper-bound, with probability at least $1-\delta$

$$\|\theta_i^T - \widehat{\theta}_i\|^2 \le \frac{12\Gamma^2}{\mu_f^2}\frac{\log\log(T)}{T} + 8G(121G+1)\left(1+\log\frac{1}{\delta}\right)\frac{\log\log(T)}{T}.$$

From the conditions of the Lemma, we can then conclude that

$$\|\theta_i^T - \widehat{\theta}_i\| \le \varepsilon. \tag{28}$$

Plugging (28) into (27) and combining it with (26), we finally get that, with probability at least $1-\delta$

$$F_k(\theta_i^T) - F_k(\theta_k^\star) \le G_{F_k}\varepsilon + F_k(\widehat{\theta}_i) - F_k(\theta_k^\star).$$

The result is completed by applying Lemma 1 to the second term on the right hand side of the final inequality. □

Lemma 6. *Let Assumptions 1-7 hold and each user runs SGD locally for T iterations, to produce* θ_i^T*. If* $T \ge \frac{4\Gamma^2}{\mu_f^2\varepsilon}$*, then for* $\widetilde{\theta}_k = \frac{1}{|C_k|}\sum_{i\in C_k}\theta_i^T$*,* $k\in[K]$*, we have*

$$\mathbb{E}\|\widetilde{\theta}_k - \theta_k^\star\|^2 \le \frac{4E_k}{n|C_k|} + \frac{10}{\mu_{F_k}^2 n^2}\left(H_k^2\log d + E_k\right)E_k + \mathcal{O}\left(|C_k|^{-1}n^{-2}\right) + \mathcal{O}(n^{-3}) + \varepsilon,$$

where $E_k \coloneqq \mathbb{E}\|\nabla^2 F_k(\theta_k^\star)^{-1}\nabla\ell(\theta_k^\star;X)\|^2$*.*

Proof. From Lemma 3, we know that, for each $i\in[m]$, running SGD locally for $T \ge \frac{4\Gamma^2}{\mu_f^2\varepsilon}$ iterations results in

$$\mathbb{E}\|\theta_i^T - \widehat{\theta}_i\|^2 \le \varepsilon. \tag{29}$$

Define the across-cluster average of ε-inexact approximations as $\widetilde{\theta}_k = \frac{1}{|C_k|}\sum_{i\in C_k}\theta_i^T$. We then have

$$\mathbb{E}\|\widetilde{\theta}_k - \theta_k^\star\|^2 \le 2\mathbb{E}\|\overline{\theta}_k - \theta_k^\star\|^2 + 2\mathbb{E}\|\widetilde{\theta}_k - \overline{\theta}_k\|^2, \tag{30}$$

where $\overline{\theta}_k = \frac{1}{|C_k|}\sum_{i \in C_k} \widehat{\theta}_i$. We can bound the first term on the right-hand side of (30) using Lemma 2. For the second term, we use (29), to obtain

$$\mathbb{E}\|\widetilde{\theta}_k - \overline{\theta}_k\|^2 \leq \frac{1}{|C_k|}\sum_{i \in C_k} \mathbb{E}\|\theta_i^T - \widehat{\theta}_i\|^2 = \varepsilon.$$

Combining the results and plugging in (30), we get

$$\begin{aligned}\mathbb{E}\|\widetilde{\theta}_k - \theta_k^\star\|^2 \leq \frac{4E_k}{n|C_k|} + \frac{10}{\mu_{F_k}^2 n^2}\left(H_k^2 \log d + E_k\right) E_k \\ + \mathcal{O}\left(|C_k|^{-1} n^{-2}\right) + \mathcal{O}(n^{-3}) + \varepsilon,\end{aligned}$$

which completes the proof. □

Lemmas 5 and 6 give us the counterparts of Lemmas 1 and 2 in the case where an approximate solution to the ERM is used instead of the exact one. We can apply them to prove the following.

Theorem 2. *Let Assumptions 1-7 hold and $\|g\|^2 \leq \Gamma^2$ with probability 1. If each user runs SGD locally for T iterations to produce θ_i^T, $i \in [m]$ and the number of samples per user n and the number of local iterations T are such that $n > 3$, $T \geq \max\left\{15, \frac{4\Gamma^2}{\mu^2}\right\}$ and moreover*

$$\begin{aligned}\frac{n}{\log n} &> 2M\left(\frac{(D-2\gamma)^2|C_{(K)}|^2}{(2m - |C_{(K)}| - |C_{(K-1)}|)^2} - 4\varepsilon S_F\right)^{-1}, \\ \frac{T}{\log\log(T)} &\geq \left(\frac{12\Gamma^2}{\mu_f^2} + 8G(121G+1)(1+\beta\log n)\right)\frac{1}{\varepsilon^2}\end{aligned}$$

where $\beta \geq 1$ and $0 < \gamma < \frac{D}{2}$ are tunable parameters, $S_F = \max_{k \in [K]} \frac{G_{F_k}}{\mu_{F_k}}$, while $M = M(\beta) = \max_{i,j \in C_k, k \in [K]} M_{ik} + M_{jk}$, and for all $i \in C_k$, $k \in [K]$

$$\begin{aligned}M_{ik} = &\frac{64R^2L\left(\log 2 + d\log 6R + (d+1)\beta\right)}{\mu_{F_k}} \\ &+ \frac{16LF_k(\theta_k^\star)(\log 2 + \beta)}{\mu_{F_k}^2} + \frac{16R\|\nabla f_i(\theta_k^\star)\|(\log 2 + \beta)}{\mu_{F_k}} \\ &+ \frac{2G_{F_k} + 16RL\left(1 + \log 2 + d\log 6R + (d+1)\beta\right)}{\mu_{F_k}},\end{aligned}$$

then, for any choice of $\lambda \in \left[\sqrt{\frac{2M\log n}{n} + 4\varepsilon S_F}, \frac{|C_{(K)}|(D-2\gamma)}{2m-|C_{(K-1)}|-|C_{(K)}|}\right)$*, we have that, for all* $k \in [K]$*, the models produced by the inexact method achieve the MSE*

$$\begin{aligned}
\mathbb{E}\|\widetilde{\theta}_k - \theta_k^\star\|^2 &\leq \frac{4E_k}{n|C_k|} + \frac{4K\widetilde{E}R^2}{n|C_{(K)}|\gamma^2} + \frac{3KR^2|\widetilde{C}|^2}{n^\beta} \\
&+ \frac{10E_k}{\mu_{F_k}^2 n^2}\left(H_k^2\log d + E_k\right) + \frac{10R^2KE_{\max}}{\mu_{F_{\min}}^2 n^2\gamma^2}\left(\widetilde{H}^2\log d + \widetilde{E}\right) \\
&+ \mathcal{O}\left(\frac{1}{|C_k|n^2}\right) + \mathcal{O}\left(\frac{K}{|C_{(K)}|n^2\gamma^2}\right) + \mathcal{O}\left(\frac{1}{n^3}\right) \\
&+ \mathcal{O}\left(\frac{K}{n^3\gamma^2}\right) + \varepsilon\left(1 + \frac{2R^2K}{\gamma^2}\right),
\end{aligned}$$

where $E_k = \mathbb{E}\|\nabla^2 F_k(\theta_k^\star)^{-1}\nabla\ell(\theta_k^\star; X)\|^2$*,* $E_{\max} = \max_{k\in[K]} E_k$*,* $\widetilde{E} = \frac{1}{K}\sum_{k\in[K]} E_k$*,* $\widetilde{H} = \frac{1}{K}\sum_{k\in[K]} H_k$ *and* $|\widetilde{C}|^2 = \frac{1}{K}\sum_{k\in[K]} |C_k|^2$*.*

We can provide an analogue to Corollary 1 in the inexact ERM scenario.

Corollary 2. *Let conditions of Theorem 2 hold. If additionally* $D > 2\sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}}$ *and* $n \geq |C_{(1)}|$*, then for the choice of* $\beta \geq 2$ *and* $\gamma = \sqrt{\frac{|C_{(1)}|}{|C_{(K)}|}}$*, we have the following MSE, for all* $k \in [K]$

$$\mathbb{E}\|\widetilde{\theta}_k - \theta_k^\star\|^2 \leq \mathcal{O}\left(\frac{1}{n|C_k|} + \varepsilon\right).$$

The proof of Theorem 2 follows the same idea as the proof of Theorem 1, replacing the results of Lemmas 1 and 2 with results from Lemmas 5 and 6. For the sake of brevity, we omit the proof. Some comments are now in order.

Remark 18. *Comparing the MSE rates of Theorem 1 and Theorem 2, we can see that the main difference is the presence of an additional term in Theorem 2, that being*

$$\varepsilon\left(1 + \frac{2R^2K}{\gamma^2}\right),$$

with $\varepsilon > 0$ *representing the accuracy up to which we solve the local ERM. We can therefore see that, as long as the local ERMs are solved up to precision* $\varepsilon = \mathcal{O}\left(\frac{1}{n|C_{(1)}|}\right)$*, the rates of Theorem 1 are recovered, i.e., the final MSE* *is of the order* $\mathcal{O}\left(\frac{1}{n|C_{(k)}|}\right)$, *for all* $k \in [K]$. *This in turns leads to a local iteration requirement of* $T \geq \max\left\{15, \frac{4n|C_{(1)}|\Gamma^2}{\mu_\ell^2}\right\}$ *and*

$$\frac{T}{\log\log(T)} \geq \left(\frac{6L\Gamma^2}{\mu_\ell^2} + 4LG(121G+1)(1+\beta\log n)\right)n^2|C_{(1)}|^2.$$

Remark 19. *We can see from Corollary 2 that, if we solve the local problems up to precision* $\varepsilon = \frac{1}{n|C_{(1)}|}$, *we again obtain the order-optimal MSE rates*

$$\mathbb{E}\|\widetilde{\theta}_k - \theta_k^\star\|^2 = \mathcal{O}\left(\frac{1}{n|C_k|}\right),$$

for all $k \in [K]$.

Remark 20. *Note that the sample size requirement implicitly places a requirement on the precision up to which we solve the local ERMs, i.e., we have*

$$\varepsilon < \frac{(D-2\gamma)^2|C_{(K)}|^2}{4S_F(2m-|C_{(K)}|-|C_{(K-1)}|)^2}.$$

This requirement can again be seen in terms of the "problem difficulty", with respect to different system aspects. For example, if the clusters are well separated, so that $D-2\gamma$ *is large, we can solve the local ERMs up to moderate, or even low precision, while for clusters that are not well separated, we need to solve the local ERMs to high precision in order to achieve the optimal rates. Similarly, if the clusters are well balanced, i.e.,* $|C_k| = \frac{m}{K}$, *for all* $k \in [K]$, *the term* $\frac{|C_{(K)}|^2}{(2m-|C_{(K)}|-|C_{(K-1)}|)^2}$ *evaluates to* $\frac{1}{4(K-1)^2}$, *while in the extreme case of* $|C_{(K)}| = |C_{(K-1)}| = 1$, *the term evaluates to* $\frac{1}{4(m-1)^2}$. *For* $K \ll m$, *we see that balanced clusters (easier clustering problem) again lead to a lower precision requirement than the imbalanced clusters case. Finally, recall that* $S_F = \max_{k\in[K]}\frac{G_{F_k}}{\mu_{F_k}}$, *where* G_{F_k} *is the Lipschitz constant of* F_k *(not the gradient!), while* μ_{F_k} *is the strong convexity constant of* F_k, *hence showing that, if* F_k*'s are strongly convex (high* μ_{F_k}*) and don't have big jumps (low Lipschitz consant* G_{F_k}*), the overall precision to which we have to solve the local ERMs is relaxed.*

Remark 21. *The choice of SGD as the local solver is based on the flexibility offered by the algorithm. The results from Lemmas 3 and 4 do not depend on either the setting being online or locally stored data, nor do they place any requirement on the mini-batch size used. This however leads to sub-optimal* *dependence on ε in the requirements on the number of local iterations each user has to run.*

Remark 22. *If all the n local data samples were available to each user, variance reduction methods such as SAGA [45] and SVRG [46] could be applied, making the number of iterations T dependence on ε only logarithmical, i.e., $T = \mathcal{O}\left(\log \frac{1}{\varepsilon}\right)$.*

Remark 23. *Finally, we remark that Assumption 6 is the most general form assumption on the loss function and as such, leads to the requirement of solving the ERM to precision ε^2. As shown in [33], Theorem 2, if the loss is a generalized linear loss, then it suffices to solve the ERM up to precision ε. While such an assumption is satisfied by a certain class of strongly convex loss functions, such as support vector machines, linear and logistic regression, it is less general than Assumption 6.*

5 Numerical experiments

In this section we present numerical experiments on linear regression problem. All of the experiments are implemented in python. To solve the local empirical risk problems, we use CVXPY [47]. The results presented in subsections below are averaged across 20 runs.

We consider a linear regression problem, where the data generating process for each cluster is given by

$$y = \langle x, u_k^\star \rangle + \epsilon,$$

where $\epsilon \sim \mathcal{N}(0,1)$, i.e., ϵ follows a standard Gaussian distribution. The number of clusters is set to $K = 10$. The vectors $u_k^\star$ are d-dimensional, with $d = 20$, and each component is drawn from a uniform distribution, independent of one another. Specifically, we drew $u_k^\star$'s as: $u_{1i}^\star \sim \mathrm{U}([1,2])$, $u_{2i}^\star \sim \mathrm{U}([4,5])$, $u_{3i}^\star \sim \mathrm{U}([7,8])$, $u_{4i}^\star \sim \mathrm{U}([10,11])$ and $u_{5i}^\star \sim \mathrm{U}([13,14])$, with $u_6^\star$ through $u_{10}^\star$ begin generated from the corresponding negative intervals, i.e., $u_6^\star \sim \mathrm{U}([-2,-1])$, through to $u_{10i}^\star \sim \mathrm{U}([-14,-13])$, respectively, for all $i \in [d]$. Such a choice of $u_k^\star$'s ensures that $D > 0$. Each cluster is assigned a total of $N_k = 100000$ points, where the datapoints x are generated as follows: for each $x \in \mathbb{R}^d$, we choose 5 random components in $[d]$ that are distributed according to $\mathcal{N}(0,1)$, while the other components are set to zero. A similar setup was considered in [14], with $K = 1$.

To measure the error, we use the quadratic loss, i.e.,

$$\ell\left((x,y);u\right) = (y - \langle x, u \rangle)^2.$$

Under the proposed loss, we have that $u_k^\star$'s are the population optimal models, i.e., $u_k^\star = \arg\min_u F_k(u)$, $k \in [K]$.

We consider a FL system with $m = 100$ users and a balanced clustering, i.e., $|C_k| = \frac{m}{K} = 10$, for all $k \in [K]$. Each user $i \in C_k$ is assigned n points uniformly at random, from the corresponding sample N_k, such that no data point is assigned to two different users, effectively simulating an IID distribution of data within clusters. We benchmark the proposed method with the following methods:

- *Oracle Averaging* - an oracle method that knows the true clusters beforehand and applies the averaging method from [14] on each individual cluster, i.e.,
$$\overline{u}_k = \frac{1}{|C_k|} \sum_{i \in C_k} \hat{u}_i, \tag{31}$$
with $\hat{u}_i$ the local ERM of user i and C_k, $k \in [K]$ being the true underlying clustering;
- *Cluster Oracle* - an oracle method that contains all of the data points assigned to the users from the same cluster, i.e., a total of $\frac{mn}{K}$ data points per cluster and trains the models on all of the data, i.e.,
$$u_k = \arg\min_u \frac{K}{m} \sum_{i \in C_k} f_i(u),$$
with f_i's given by (4);
- *Local ERMs* - ERMs trained on each user's local data;
- *Naive averaging* - the method from [14], that averages the local ERMs across all users, oblivious to system heterogeneity.

Cluster Oracle is the equivalent of centralized learning, i.e., is the method that trains on all the data available in the cluster, achieving the best order-optimal MSE rate $\mathcal{O}\left(\frac{1}{n|C_k|}\right)$ (e.g., [33]). On the other hand, [14] show that Oracle Averaging matches the performance of Cluster Oracle if the sample size is above a threshold. Therefore, using Cluster Oracle and Oracle Averaging as benchmarks illustrates: 1) how fast our method attains the order-optimal MSE rate and 2) the additional requirements on the sample size to reach the order-optimal rate, compared to Oracle Averaging, that stem from not knowing the true clustering.

To measure the quality of performance, we present the average normalized MSE, i.e., for each of the above estimators, we compute

$$\frac{1}{m}\sum_{i=1}^{m}\frac{\|\widetilde{u}_i - u^\star_{(i)}\|^2}{\|u^\star_{(i)}\|^2}, \tag{32}$$

where $u^\star_{(i)}$ denotes the population optima associated with user i, while $\widetilde{u}_i$ is the estimator associated with user i. For example, if we measure the performance of Oracle Averaging estimator from (31), (32) evaluates to

$$\frac{1}{K}\sum_{k\in[K]}\frac{\|\overline{u}_k - u^\star_k\|^2}{\|u^\star_k\|^2}.$$

To select the parameter λ, we first compute the lower and upper bounds in (12). If the condition is satisfied, so that the lower bound is strictly smaller than the upper bound, we choose λ uniformly at random from the interval defined by the lower and upper bounds in (12). Otherwise, for simplicity, we take lambda to be equal to the upper bound.

Figure 1 presents the performance using the linear regression models. On y-axis we plot the averaged normalized MSE (32), while on the x-axis, we present the number of samples n available to each user. We can see that, for a small number of samples (less than [illegible]00), our method clusters each user to an individual cluster, effectively performing like the local ERMs. This can be explained by the fact that in the small sample size regime, the condition (10) is not satisfied (with high probability) and typically the upper bound will be small, hence resulting in a large number of clusters. The results can potentially be improved by running clusterpath, but for illustrative purposes, we went with the simple choice of setting lambda to be equal to the upper bound. On the other hand, as n grows, we see a sharp phase transition in the quality of our estimator, in the interval between 300 and 400 samples, after which the performance of our method matches the order-optimal performance of both the oracle methods, as predicted by the theory. Oracle Averaging performs slightly worse than Cluster Oracle in the small sample regime, but quickly matches the performance of Cluster Oracle, as expected. The difference in the number of samples required for reaching order-optimal rates of our proposed method and the Oracle Averaging (450 and 350 samples required, respectively), as outlined above, stems from the additional requirements of our method to produce an exact clustering. Finally, we see that the naive averaging method consistently performs badly, as it is completely oblivious to the clustering structure, hence illustrating that a global model approach can be bad in the presence of system heterogeneity.

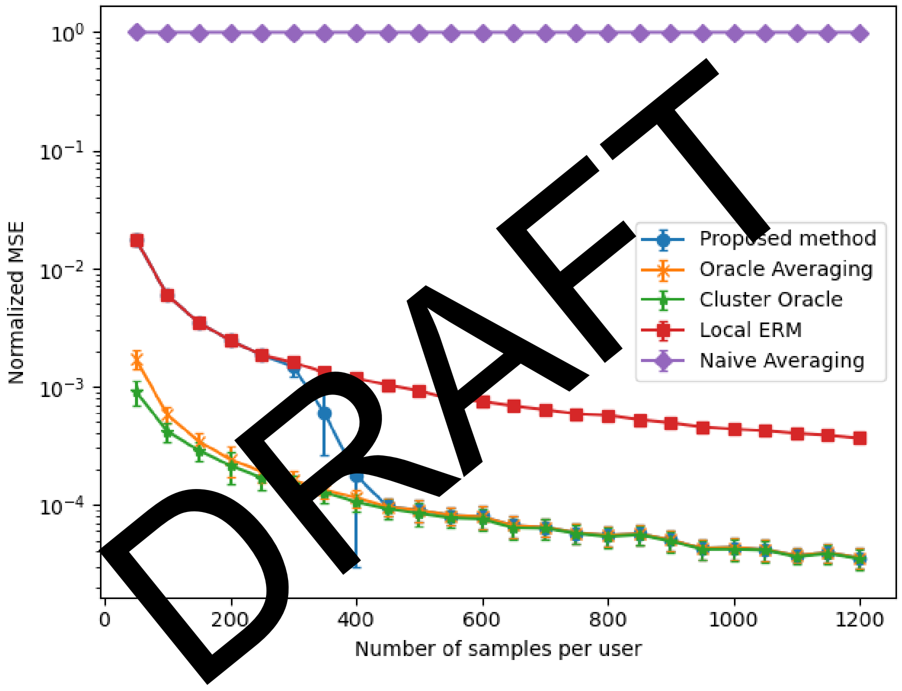


Figure 1: Performance of different methods for linear regression, versus the number of samples available per user. We can see that our method matches the order-optimal MSE rates for a sufficiently large sample size.

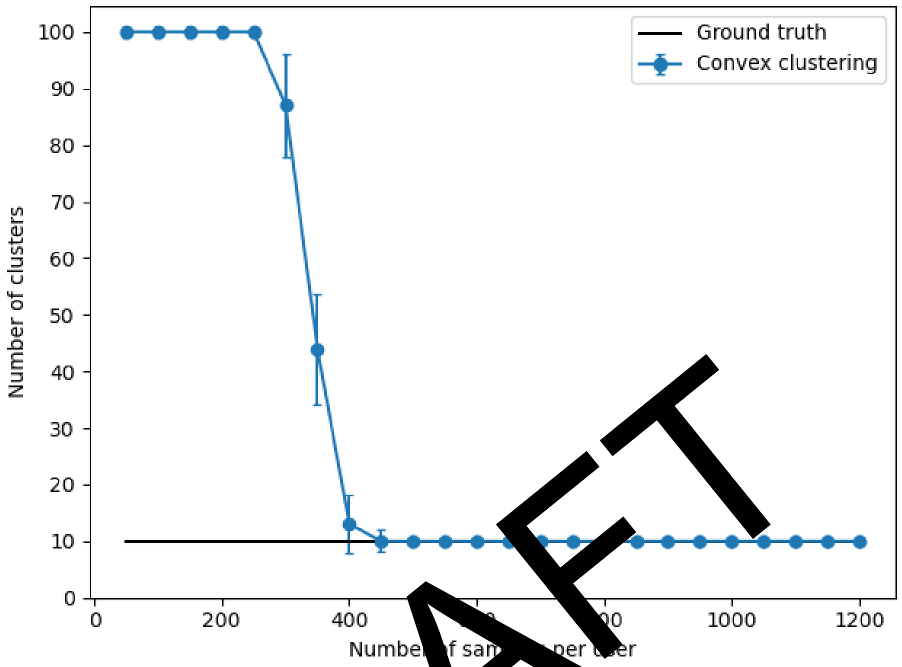


Figure 2: Number of clusters produced by convex clustering for linear regression, versus the number of samples available per user. We can see that convex clustering is able to recover the exact clustering for sufficiently large sample size available to each user.

Figure 2 presents the performance of convex clustering. On y-axis, we plot the number of clusters produced by the convex clustering algorithm. On x-axis, we again plot the number of samples n. Figure 2 is consistent with the results from Figure 1, as it shows that, for small n (less than 300), convex clustering clusters each user separately, which, due to the low sample regime and our sub-optimal choice of λ, is to be expected. On the other hand, there is a sharp phase transition in the number of clusters for n between 300 and 400, after which convex clustering consistently produces $K' = 10$ clusters. Moreover, we can see that the clustering produced by the convex clustering method is correct, as our method matches the performance of both oracle methods that know the true clustering.

6 Conclusion

We proposed a one-shot approach for CFL, based on a simple inference and averaging scheme. The proposed approach is communication efficient, as it requires a single round of communication. Moreover, our theoretical analysis showed that the method provides order-optimal MSE rates, in terms of the sample size. Compared to the state-of-the-art algorithms that require multiple rounds of communication, our method improve the existing results by a factor that is logarithmic in the total number of samples in the system, our metod provides significant communication reduction. Remarkably, unlike other methods that require knowledge of K, e.g., [29], [28],[30], our method does not require any knowledge of the underlying number of clusters K. Numerical experiments corroborate our findings.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[2] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[3] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/6c340f25839e6acdc73414517203f5f0-Paper.pdf

[4] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, S. Chiappa

and R. Calandra, Eds., vol. 108. PMLR, 26–28 Aug 2020, pp. 2021–2031. [Online]. Available: https://proceedings.mlr.press/v108/reisizadeh20a.html

[5] A. Koloskova, S. Stich, and M. Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 3478–3487. [Online]. Available: https://proceedings.mlr.press/v97/koloskova19a.html

[6] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified sgd with memory," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/b440509a0106086a67bc2ea9df0a1dab-Paper.pdf

[7] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/3328bdf9a4b9504b9398284244fe97c2-Paper.pdf

[8] Y. J. Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," *arXiv preprint arXiv:2010.01243*, 2020.

[9] W. Chen, S. Horvath, and P. Richtarik, "Optimal client sampling for federated learning," *arXiv preprint arXiv:2010.13723*, 2020.

[10] M. Ribero and H. Vikalo, "Communication-efficient federated learning via optimal client sampling," *arXiv preprint arXiv:2007.15197*, 2020.

[11] S. U. Stich, "Local sgd converges fast and communicates little," in *ICLR 2019-International Conference on Learning Representations*, no. CONF, 2019.

[12] A. Khaled, K. Mishchenko, and P. Richtarik, "Tighter theory for local sgd on identical and heterogeneous data," in *Proceedings of the*

Twenty Third International Conference on Artificial Intelligence and Statistics, ser. Proceedings of Machine Learning Research, S. Chiappa and R. Calandra, Eds., vol. 108. PMLR, 26–28 Aug 2020, pp. 4519–4529. [Online]. Available: https://proceedings.mlr.press/v108/bayoumi20a.html

[13] K. Mishchenko, G. Malinovsky, S. Stich, and P. Richtarik, “ProxSkip: Yes! Local gradient steps provably lead to communication acceleration! Finally!” in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 15 750–15 769. [Online]. Available: https://proceedings.mlr.press/v162/mishchenko22b.html

[14] Y. Zhang, M. J. Wainwright, and J. C. Duchi, “Communication-efficient algorithms for statistical optimization,” in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/e7f8a7fb0b77bcb3b283af5be021448f-Paper.pdf

[15] N. Guha, A. Talwalkar, and V. Smith, “One-shot federated learning,” *arXiv preprint arXiv:1902.11175*, 2019.

[16] Y. Zhou, G. Pu, X. Ma, X. Li, and D. Wu, “Distilled one-shot federated learning,” *arXiv preprint arXiv:2009.07999*, 2020.

[17] S. Salehkaleybar, A. Sharifnassab, and S. J. Golestani, “One-shot federated learning: Theoretical limits and algorithms to achieve them,” *Journal of Machine Learning Research*, vol. 22, no. 189, pp. 1–47, 2021. [Online]. Available: http://jmlr.org/papers/v22/19-1048.html

[18] D. K. Dennis, T. Li, and V. Smith, “Heterogeneity for the win: One-shot federated clustering,” in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 2611–2620. [Online]. Available: https://proceedings.mlr.press/v139/dennis21a.html

[19] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, “Tackling the objective inconsistency problem in heterogeneous federated optimization,” *Advances in neural information processing systems*, vol. 33, pp. 7611–7623, 2020.

[20] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," *arXiv preprint arXiv:2002.04758*, 2020.

[21] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/6211080fa89981f66b1a0c9d55c61d0f-Paper.pdf

[22] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," *arXiv preprint arXiv:2002.05516*, 2020.

[23] F. Hanzely, S. Hanzely, S. Horváth, and P. Richtárik, "Lower bounds and optimal algorithms for personalized federated learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 2304–2315, 2020.

[24] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 3557–3568. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/24389bfe4fe2eba8bf9aa9203a44cdad-Paper.pdf

[25] M. Khodak, M.-F. F. Balcan, and A. S. Talwalkar, "Adaptive gradient-based meta-learning methods," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/f4aa0dd960521e045ae2f20621fb4ee9-Paper.pdf

[26] C. T Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with moreau envelopes," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 394–21 405, 2020.

[27] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6357–6368.

[28] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *arXiv preprint arXiv:2002.10619*, 2020.

[29] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," *IEEE Transactions on Information Theory*, pp. 1–1, 2022.

[30] A. Ghosh, J. Hong, D. Yin, and K. Ramchandran, "Robust federated learning in a heterogeneous environment," *arXiv preprint arXiv:1906.06629*, 2019.

[31] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3710–3722, 2021.

[32] A. Armacki, D. Bajovic, D. Jakovetic, and S. Kar, "Personalized federated learning via convex clustering," *arXiv preprint arXiv:2202.00718*, 2022.

[33] K. Sridharan, S. Shalev-Shwartz, and N. Srebro, "Fast rates for regularized objectives," in *Advances in Neural Information Processing Systems*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., vol. 21. Curran Associates, Inc., 2008. [Online]. Available: https://proceedings.neurips.cc/paper/2008/file/abd815286ba1007abfbb8415b83ae2cf-Paper.pdf

[34] L. Zhang, T. Yang, and R. Jin, "Empirical risk minimization for stochastic convex optimization: $O(1/n)$- and $O(1/n^2)$-type of risk bounds," in *Proceedings of the 2017 Conference on Learning Theory*, ser. Proceedings of Machine Learning Research, S. Kale and O. Shamir, Eds., vol. 65. PMLR, 07–10 Jul 2017, pp. 1954–1979. [Online]. Available: https://proceedings.mlr.press/v65/zhang17a.html

[35] A. Panahi, D. Dubhashi, F. D. Johansson, and C. Bhattacharyya, "Clustering by sum of norms: Stochastic incremental algorithm, convergence and cluster recovery," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 2769–2777. [Online]. Available: https://proceedings.mlr.press/v70/panahi17a.html

[36] T. D. Hocking, A. Joulin, F. Bach, and J.-P. Vert, "Clusterpath: An algorithm for clustering using convex fusion penalties," in *Proceedings of the 28th International Conference on International Conference on*

Machine Learning, ser. ICML'11. Madison, WI, USA: Omnipress, 2011, p. 745–752.

[37] D. Sun, K.-C. Toh, and Y. Yuan, "Convex clustering: Model, theoretical guarantee and efficient algorithm." *J. Mach. Learn. Res.*, vol. 22, pp. 9–1, 2021.

[38] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[39] A. Banerjee, S. Merugu, I. S. Dhillon, J. Ghosh, and J. Lafferty, "Clustering with bregman divergences." *Journal of machine learning research*, vol. 6, no. 10, 2005.

[40] A. Armacki, D. Bajovic, D. Jakovetic, and S. Kar, "Gradient based clustering," in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 929–947. [Online]. Available: https://proceedings.mlr.press/v162/armacki22a.html

[41] W. Ben-Ameur, P. Bianchi, and J. Jakubowicz, "Robust distributed consensus using total variation," *IEEE Transactions on Automatic Control*, vol. 61, no. 6, pp. 1550–1564, 2016.

[42] E. C. Chi and K. Lange, "Splitting methods for convex clustering," *Journal of Computational and Graphical Statistics*, vol. 24, no. 4, pp. 994–1013, 2015, pMID: 27087770. [Online]. Available: https://doi.org/10.1080/10618600.2014.948181

[43] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400 – 407, 1951. [Online]. Available: https://doi.org/10.1214/aoms/1177729586

[44] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, 2012, pp. 1571–1578.

[45] A. Defazio, F. Bach, and S. Lacoste-Julien, "Saga: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence,

and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/ede7e2b6d13a41ddf9f4bdef84fdc737-Paper.pdf

[46] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in Neural Information Processing Systems*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013. [Online]. Available: https://proceedings.neurips.cc/paper/2013/file/ac1dd209cbcc5e5d1c6e28598e8cbbe8-Paper.pdf

[47] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

A Appendix

In this Appendix we show conditions under which it is suitable for two users with different distributions to form a cluster by merging their respective data. As in the main body, we use $\theta_k^\star$ and $\widehat{\theta}_i$ to denote the population optimal of cluster $k \in [K]$ and local ERM of user $i \in [m]$, respectively, i.e., $\theta_k^\star = \arg\min_{\theta\in\Theta} F_k(\theta)$ and $\widehat{\theta}_i = \arg\min_{\theta\in\Theta} f_i(\theta)$. We then have the following result.

Lemma 7. *Let Assumption 3 hold and assume ℓ is strongly convex. Let all users sample data from a unique distribution, i.e., each user i samples data following $\mathcal{D}_i$, $i \in [m]$. Denote by $\mathcal{D}_k$ the mixture of distributions $\mathcal{D}_i$ and $\mathcal{D}_j$, i.e., the distribution such that $F_k(\theta) = p_iF_i(\theta) + p_jF_j(\theta)$, where $0 < p_i, p_j < 1$, such that $p_i + p_j = 1$. If the distributions $\mathcal{D}_i$ and $\mathcal{D}_i$ are such that*

$$\|\theta_i^\star - \theta_j^\star\|^2 < \epsilon,$$

then, with high probability

$$\|\widehat{\theta}_k - \theta_m^\star\|^2 = \mathcal{O}\left(\frac{1}{n_i + n_j} + \epsilon\right),$$

with $m = i, j$, where $\widehat{\theta}_k = \arg\min_{\theta\in\Theta} p_if_i(\theta) + p_jf_j(\theta)$.

Some remarks are now in order.

Remark 24. *Lemma 7 tells us that, as long as* $\frac{1}{n_i+n_j}+\epsilon < \min\left\{\frac{1}{n_i}, \frac{1}{n_j}\right\}$*, i.e.,* $\epsilon < \frac{\min\{n_i,n_j\}}{\max\{n_i,n_j\}(n_i+n_j)}$*, we have that the model trained on the joint datasets of users* i *and* j *is beneficial to both users. For example, when* $n_i = n$*,* $\forall i \in [m]$*, the condition on* ϵ *evaluates to* $\epsilon < \frac{1}{2n}$*.*

Remark 25. *If* $\frac{1}{n_i+n_j} + \epsilon < \min\left\{\frac{1}{n_i}, \frac{1}{n_j}\right\}$*, Lemma 7 tells us that it is beneficial to treat the users* i *and* j *as belonging to the same cluster. Therefore, averaging the local ERMs trained by users* i *and* j *leads to mutual benefits, even though the two users come from different, but mutually close distributions (as measured by the distance of the population optima). Therefore, it is beneficial to treat users* i*,* j *as belonging to the same cluster, justifying the assumption that* $1 < K < m$*.*

Proof of Lemma 7. Applying the results of [33], we have that, with high probability

$$\|\widehat{\theta}_i - \theta_i^\star\|^2 = \mathcal{O}\left(\frac{1}{n_i}\right).$$

Denote by $\theta_k^\star$ the population optima of the mixture distribution $\mathcal{D}_k$. We then have

$$\begin{aligned}\|\widehat{\theta}_k - \theta_i^\star\|^2 &\leq 2\|\widehat{\theta}_k - \theta_k^\star\|^2 + 2\|\theta_k^\star - \theta_i^\star\|^2 \\ &\leq \mathcal{O}\left(\frac{1}{n_i + n_j}\right) + 2\|\theta_k^\star - \theta_i^\star\|^2,\end{aligned} \tag{33}$$

where the second inequality again follows from [33]. Using strong convexity of F's, we have that

$$\begin{aligned}\frac{\mu}{2}\|\theta_k^\star - \theta_i^\star\|^2 &\leq F_k(\theta_i^\star) - F_k(\theta_k^\star) \\ &= p_i F_i(\theta_i^\star) + p_j F_j(\theta_i^\star) - p_i F_i(\theta_k^\star) - p_j F_j(\theta_k^\star) \\ &\leq p_j \left(F_j(\theta_i^\star) - F_j(\theta_k^\star)\right) \\ &= p_j \left(F_j(\theta_i^\star) - F_j(\theta_j^\star) + F_j(\theta_j^\star) - F_j(\theta_k^\star)\right) \\ &\leq p_j \left(F_j(\theta_i^\star) - F_j(\theta_j^\star)\right),\end{aligned}$$

where we used the fact that $\theta_m^\star = \arg\min_{\theta\in\Theta} F_m(\theta)$, $m \in \{i, j\}$ in the second and third inequalities, respectively, with μ the strong convexity parameter of ℓ. Finally, using L-Lipschitz continuous gradients of F's, we get that

$$F_j(\theta_i^\star) - F_j(\theta_j^\star) \leq \frac{L}{2}\|\theta_i^\star - \theta_j^\star\|^2 = \mathcal{O}(\epsilon). \tag{34}$$

Combining (33) and (34), we get, with high probability

$$\|\widehat{\theta}_k - \theta_i^\star\|^2 = \mathcal{O}\left(\frac{1}{n_i + n_j} + \epsilon\right).$$

Analogous results can be obtained for user j, hence the claim follows. $\square$

C Anomaly Detection through Unsupervised Federated Learning

The appended paper follows.

Anomaly Detection through Unsupervised Federated Learning

Mirko Nardi
Scuola Normale Superiore
Pisa, Italy
mirko.nardi@sns.it

Lorenzo Valerio
IIT-CNR
Pisa, Italy
lorenzo.valerio@iit.cnr.it

Andrea Passarella
IIT-CNR
Pisa, Italy
andrea.passarella@iit.cnr.it

***Abstract*—Federated learning (FL) is proving to be one of the most promising paradigms for leveraging distributed resources, enabling a set of clients to collaboratively train a machine learning model while keeping the data decentralized. The explosive growth of interest in the topic has led to rapid advancements in several core aspects like communication efficiency, handling non-IID data, privacy, and security capabilities. However, the majority of FL works only deal with supervised tasks, assuming that clients' training sets are labeled. To leverage the enormous unlabeled data on distributed edge devices, in this paper, we aim to extend the FL paradigm to unsupervised tasks by addressing the problem of anomaly detection (AD) in decentralized settings. In particular, we propose a novel method in which, through a preprocessing phase, clients are grouped into communities, each having similar majority (i.e., inlier) patterns. Subsequently, each community of clients trains the same anomaly detection model (i.e., autoencoders) in a federated fashion. The resulting model is then shared and used to detect anomalies within the clients of the same community that joined the corresponding federated process. Experiments show that our method is robust, and it can detect communities consistent with the ideal partitioning in which groups of clients having the same inlier patterns are known. Furthermore, the performance is significantly better than those in which clients train models exclusively on local data and comparable with federated models of ideal communities' partition.**

***Index Terms*—federated learning, unsupervised, anomaly detection**

I. Introduction

Distributed/decentralized ML executed at the edge represents one of the most promising approaches capable of addressing the issues that afflict centralized solutions.

In this regard, the Federated Learning (FL) [1] paradigm has proved to be an effective and promising approach to face the hard challenges triggered by these distributed settings. It essentially aims to collaboratively train an ML model while keeping the data decentralized through the exchange of models' parameters updates (instead of raw data) that, in its vanilla version, are iteratively aggregated and shared by a central coordinating node.

Given its effectiveness, in the last years plenty of subsequent research works have been released focusing on different core aspects: improving communication efficiency [2], increasing model performance in combination with non-IID data [3], extending privacy and security capabilities [4] and addressing client hardware variability [5].

Nevertheless, FL applications and implementations for mobile edge devices are still largely designed for supervised learning tasks as a spontaneous consequence of its original development purpose [6]. Thus, one of the least treated aspects is the extension of FL to other ML paradigms like unsupervised learning, reinforcement learning, active learning, and online learning [illegible].

This paper specifically aims to apply FL on unsupervised tasks for mobile edge devices. Unsupervised learning (as well as semi-supervised and self-supervised learning) has recently been considered one of the next great frontiers for AI [8]. Unlabeled data far surpasses labeled data in real-world applications. Hence its integration with federated contexts is mandatory to fully unleash the potential of this approach.

In this paper, we consider nodes that have to learn a common ML model (e.g., a classifier). We assume that sets of these nodes "see" similar data patterns. However, as we assume that data are not labeled, nodes need to automatically group themselves into those sets, to perform FL across members of the same set. As a specific application case, we consider anomaly detection (AD) [9], namely, the problem of identifying instances of rare events (i.e., anomalies or outliers) that are inconsistent for the majority of data considered as normal (i.e., inliers). Specifically, our methodology consists of a preprocessing phase in which each node of the system detects a membership group (cluster or community) such that each member shares similar majority/inlier patterns. In fact, to ensure the effectiveness of an anomaly detection task, a federated model must be trained on data coming from the same distribution. Once the nodes are grouped in communities, a federated learning process is spawned for each of them: nodes of the same group use their local data to collaboratively train an autoencoder to recognize their majority pattern (i.e., the inlier class). Autoencoders are particularly suitable for this purpose since typical FL protocols involve using a neural network-based model. However, the methodology is orthogonal to the specific model trained via FL. Once the federated process is finished, each client gets a much more accurate global model than it would have obtained using only its local data, as long as it has joined the proper community.

This work has been partly funded under the H2020 MARVEL (grant 957337), HumaneAI-Net (grant 952026), SoBigData++ (grant 871042) and CHIST-ERA SAI (grant CHIST-ERA-19-XAI-010, by MUR, FWF, EPSRC, NCN, ETAg, BNSF).

The proposed methodology is particularly suited for mobile environments for several reasons. First, it allows nodes not to exchange local data, thus addressing privacy and network resource limitations. Second, it supports heterogeneous settings when the federation is not under the control of a single entity (like in a datacenter), but where nodes join "freely" the federation. Third, it is tailored to using tiny ML models on individual nodes, which is mandatory for realistically implementing decentralized model training on mobile devices.

This work can subsequently be framed in a more general context of anomaly detection in which normal data belong to multiple classes (in contrast to the typical AD task involving only a single inlier class). For instance, the methodology proposed, whose output is a set of models each specialized in identifying a single normal pattern, can be further extended with ensemble-based methods to efficiently tackle the multi-class anomaly detection problem, as shown in [10].

The remainder of the paper is organised as follows. In Section II an overview of the problem and the related works are discussed. In Section IV we list the preliminaries and describe our method in detail. In Section V we discuss the results of the experiments, and in Section VI we draw the conclusions.

II. Related Works

Federated Learning is a distributed learning framework particularly amenable to optimize the computing power and the data management on edge devices. It is now widely considered modern and more effective evolution of the more traditional distributed paradigms [11]–[15], in which models are trained on large but 'flat' datasets within a fully controlled environment in terms of resource availability and data management.

FL enables to relax many of the traditional constraints and, since its introduction [1], several lines of research contribute to fast advances [7]; additionally, from the application perspective, many specific use-case solutions have already been deployed by major service providers [6], [16], [17].

Due to space reasons, in the rest of the section, we provide an overview of unsupervised approaches to FL, which are the closest area with respect to the focus of this paper.

Very few works combining federated learning and unsupervised approaches have been released, each of them dealing with limited scenarios and settings. Reference [18] is the first to introduce unsupervised representation learning in a federated setting, but it simply combines the two concepts without assuming the typical issues of distributed settings, particularly for mobile environments (e.g., dealing with non-IID data, scaling the number of devices, different application domains).

Reference [19] make progress on the same problem by adding and facing two relevant challenges: (i) inconsistency of representation spaces, due to non-IID data assumption, i.e., clients generate local models focused on different categories; (ii) misalignment of representations, given by the absence of unified information among clients.

Reference [20] introduced an unsupervised federated learning (FL) approach for speech enhancement and separation with non-IID data across multiple clients. An interesting aspect of this work is that a small portion of supervised data is exploited to boost the main unsupervised task through a combination of updates from clients, with supervised and unsupervised data.

In [21] authors present a first effort for introducing a collaborative system of autoencoders for distributed anomaly detection. However, the data collected by the edge devices are used to train the models in the cloud, which violates an essential FL feature. Locally, the models are used for inference only.

A more recent work [22] in a similar direction proposes a federated learning (FL)-based anomaly detection approach for identification and classification intrusion in IoT networks using decentralized on-device data. Here the authors use federated training rounds on Gated Recurrent Units (GRUs) models and keep the data intact on local IoT devices by sharing only the learned weights with the central server of the FL. However, dealing with a classification task still assumes the availability of labeled data.

III. Problem Formulation and Preliminaries

We consider a distributed learning system with a set of clients M and a set of data distributions C, such that $|C| \leq |M|$. With data distribution, we refer to a set of identically distributed data representing a specific pattern (e.g., observations of phenomena belonging to the same class of events, in case of a classification task). We assume that every client receives a portion $d \in$ (0%, [illegible]) of its samples from a single distribution $C_{out} \in C$, and the remaining $(100 - d)\%$ from $C_{in} \in C$, such that $C_{in} \neq C_{out}$. Thereby, the two samples partitions within each client form the outlier and inlier classes, respectively. This split represents a basic assumption when dealing with AD tasks [9]. $d \in [5\%, 15\%]$ is generally a realistic value [23], thus adopted in the majority of related works. Note that this scenario corresponds to assuming local skewed data, i.e., that each node "sees" a prevalence of data of a single class (its inlier class) and a minority of data from (one of the) other classes. This is also quite realistic in practice in AD tasks.

The challenge addressed in the paper is the following. In case of supervised learning, data belonging to each class are labelled, so each node knows which other nodes "see" the same majority class, and therefore forming FL groups is straightforward. In unsupervised cases, each node can detect its majority class from local data, but has no direct information to know which other nodes see the same majority class. Therefore, the main objective of our methodology is to identify an effective algorithm for nodes to form consistent groups (i.e., groups that see the same majority class), to then run a standard FL process across nodes of the same group.

Note that, as will be clear from the detailed description in Section IV, at the end of the first step of our methodology clients become partitioned into k disjoint groups $S_1, \ldots, S_k$. In the ideal case, each group corresponds to the (unknown to the clients) set of nodes seeing the same inlier class C_{in}, and therefore in the ideal case $k = |C|$.

IV. Proposed Methodology

As anticipated in Section I our methodology consists in two logical steps. In the first step we group clients that "see" the same inlier class, via a fully autonomous and unsupervised process. In the second step, we run a standard FL process among clients belonging the same group. We present the two steps in the following sections.

A. Step I: group identification

The aim of this phase is to make the clients join a group (i.e. cluster) having the same (or similar) majority class C_{in}.

To achieve this, we firstly train a "classical" AD model (e.g., OCSVM) on every client, using only its local data, such that each of them is able to compute a preliminary split of its data into inliers and outliers. Thereafter, every couple of clients perform the following steps: (i) they exchange their respective models, and (ii) they use the partner's model to split its local data into "normal" and "anomalous" data through an inference step. In other words, for every pair of nodes (m_i, m_j), node m_i uses node's m_j local model to classify its own local data, and vice versa. If the classification accuracy is high enough, it means that node's m_j model has been trained on the same inlier class of node a, and therefore m_i and m_j should be in the same group.

Note that, it is not necessary to use a very complex local model at this step. Although the local model of a client only enables an approximate preliminary inliers/outliers split, it suffices to detect clients sharing the same majority class of data, as long as those patterns of data in those classes are sufficiently different (as it is the case in typical AD tasks).

Given a client m_i, from its perspective this phase is detailed in Algorithm 1. Specifically, on the local data of the i-th client, i.e., x_i, an inference step is computed using its own locally trained model (line 4) and all the models of other clients (line 9). $y_{j,i}$ is the output binary vector given by the AD model of the j-th client on the data of the i-th client. Thus, $in_{j,i}$ is the portion of inliers in the vector $y_{j,i}$. The boolean $b_{j,i}$ indicates whether the i-th client flags the j-th client as a candidate for the association. The output the process corresponds to the group of candidate clients G_i with inlier classes similar to m_i.

At the end of algorithm 1, each client has a local view of which other clients should belong to its group. However, different clients in the same group may have different local views (i.e., even if m_j is in G_i, G_j may not be identical to G_i). In order to obtain an overall view of the groups, shared by all nodes, we adopt the following method.

Since the association of two clients is reciprocal (line 14), a undirected graph can be built from all the resulting groups of candidates of each client. A link between two nodes means that those two nodes mutually "think" to be in the same group. Finally, a community detection algorithm is run on this graph to detect which groups of nodes should be considered part of the same set and thus undergo a standard FL step. In other words, we assume that communities found at the end of this step are the groups of clients with the same inlier class.

Algorithm 1 Client m_i local training and association

Input: AD Model *Mod*$_i$, contamination d, association threshold q, set of other clients M
Output: Group G_i of candidate clients similar to m_i

```
 1: procedure LocalAD($Mod_i, d, q, M$)
 2:     $G_i \leftarrow \emptyset$
 3:     $Mod_i = Mod_i.fit(x_i, d)$
 4:     $y_{i,i} = Mod_i.predict(x_i)$
 5:     $in_{i,i} = inlierPercCount(y_{i,i})$
 6:     $send(Mod_i, M)$
 7:     for all $m_j$ in $M$ do
 8:         $Mod_j = receive(m_j)$
 9:         $y_{j,i} = Mod_j.predict(x_i)$
10:         $in_{j,i} = inlierPercCount(y_{j,i})$
11:         $b_{j,i} = in_{i,i} - q \leq in_{j,i} \leq in_{i,i} + q$
12:         $send(b_{j,i}, m_j)$
13:         $b_{i,j} = receive(b_{i,j}, m_j)$
14:         if $b_{j,i}$ $AND$ $b_{i,j}$ then
15:             $G_i \leftarrow m_i$
16:         end if
17:     end for
18:     return $G_i$
19: end procedure
```

B. Step II: federated outlier detection

The result of the first phase is a set of k groups (or communities) $G_0, \ldots, G_k$; for each of them a FL instance is started using autoencoders as models. Autoencoders are suitable for the purpose for two main reasons: (i) they naturally fit into the FL framework, being NN-based; (ii) they can be effectively used in AD task. In fact, they essentially learn a compressed representation of the unlabeled data used for the training, performing a nonlinear dimensionality reduction. Once trained, the reconstruction error of a given sample can be used to classify it using a threshold.

We use the vanilla version of the Federated Averaging (FedAvg) [1], a FL protocol based on averaging the local stochastic gradient descent updates to compute the global model. At the end of each federation process, the trained autoencoder is shared among the clients of the same group.

Note that, the community detection step requires either a central entity that runs the algorithm once and for all nodes, or that the graph is shared among all nodes and each runs the same community detection algorithm individually. Even in the former case, our methodology does not require that nodes share local data with any central controller, and thus can address situations where centralized learning is unfeasible or impractical (e.g., due to data ownership reasons).

V. Experiments

In this section, we describe the numerical simulations to assess the performance of the proposed methodology. The baseline is given by the **local** model scheme, in which every client trains its model using only local data. We show a further comparison with an **ideal** partitioning scheme in which the groups of clients having the same inlier patterns are known. This corresponds to a supervised FL algorithm, where all data are labeled by a central entity. Our code is based on well-accessed and standard frameworks: Tensorflow, Scikit-Learn, PyOD and Flower. For the sake of reproducibility, the code is available at https://github.com/mirqr/FedAD

A. Datasets and setup

We test our methodology on the MNIST [24] and the fashion-MNIST [25] datasets, using the original 60000-10000 train-test splits. Since both have ten classes, we have $|C| = 10$ data distributions.

Locally, given a portion of outlier d, the train set of every client has d percent of its samples from a single distribution $C_{out} \in C$, and the remaining $(100 - d)$ percent from $C_{in} \in C$, such that $C_{in} \neq C_{out}$.

With a view to a collaborative anomaly detection task, we ensure that all the datasets owned by the clients are numerically balanced and disjoint. The set of clients M that compose an experimental setup is configured as follows: we define a parameter p as the number of clients within the same data distribution (i.e., class), meaning that the train samples of a class C_{in} of the original dataset (e.g., MNIST) are evenly and randomly spread to form the inliers of p clients. Accordingly, the portion of outliers for each client within the same group and characterized by the same C_{in}, is given by the samples of a class different from C_{in}. We ensure that the outlier classes $C \setminus C_{in}$ are equally represented within the group, meaning that for each client of the group the minority class is circular through the set $C \setminus C_{in}$.

As an example, using all the available data distributions of the dataset (i.e., 10 classes), and by setting $p = 9$, then the training data distribution among the clients of the group $C_{in} = 0$ is shown in Fig. 1. The same applies to every group, i.e., an experimental system configuration ends up with $|M| = |C|p$ clients. Consequently, the ideal partitioning we aim to find through the community detection phase is composed by $k = |C| = 10$ groups with p clients each.

Note that, without loss of generality, to obtain an balanced distribution of the outliers classes among the clients of a group, it is convenient to set $p = (|C| - 1)n, n \in \mathbb{N}$. Additionally, since for each configuration run, we exploit all the samples of the dataset involved, as a higher value of p leads to smaller local datasets for the clients.

B. Models

In the first phase, every client detects the partners having the same inlier class. As explained in Section IV, a client tests the others' trained models on its local data and selects as partners those whose model produces an inliers/outliers ratio similar to its own. We select the "association" threshold q in the interval $[0.01, 0.10]$, i.e., q represents the maximum of the percentage difference between the data classified as normal by the local model, and those considered normal by using the partner's model. In other words, the local client considers another client as partner if the model of the latter produces a fraction of normal data on the local dataset equal to the percentage produced by the local node, $\pm q$. In particular, we found that the value $q = 0.08$ turns out to work well on every experiment.

We choose the model of the first phase with the following requirements: (i) it must be easy to set up and fast to train; (ii) it must be light to store and to be transmitted; (iii) it must provide a preliminary sufficiently good outlier detection to allow the clients to correctly group for the next phase.

There is not a model generally suitable for this purpose; it strongly depends on the type of data used, especially for AD tasks [26]. Moreover, the abovementioned requirements force us to discard any NN-based AD model. Thus, we have identified OC-SVM [27] to be a good choice for our cases. It requires essentially two parameters to be set: the kernel and the parameter $\nu \in (0, 1]$, which is an upper bound on the fraction of training errors and a lower bound on the fraction of support vectors. The fine-tuning of ν in contaminated data can be challenging without any assumptions on the distribution of the outliers. However, since in our tests we assume to know (only) the contamination value $d = 10\%$ for every dataset, we can set $\nu = 0.1$. Moreover, we use the RBF kernel.

For the second phase, we use a fully connected autoencoder, a NN-based model that naturally fits into a federated learning framework, with a three-layers topology (64-32-64), ReLU activation on the hidden layers, and Sigmoid activation on the output layer. Thirty-two neurons for the middle layer is a reasonable value to avoid an information bottleneck. We empirically observed that using more layers/neurons does not significantly improve the effectiveness due to the tendency of the neural network to overfit on this specific dataset.

C. Group detection and anomaly detection performance

For both the MNIST and the fashion-MNIST datasets, we run four tests varying the value of $p : \{9, 18, 27, 36\}$. In all the tests we use the contamination parameter $d = 10\%$ and we take into account all the available classes, i.e., $|C| = 10$. Let $m_{C_i,j}$ be the j-th client with majority class C_i; we define I_{C_i} as the ideal set of clients having the same majority class C_i, e.g., $I_0 = \{m_{0,0}, \dots m_{0,p-1}\}$.

In Table I, we show the results of the community detection phase for the MNIST dataset: we find nine communities, and in most cases, they match with the ideal group of clients. The major exception is given by G_4, that in all the four cases is given by the union of I_4 and I_9, meaning that the clients having 4 and 9 as inlier class join the same community. This is a consequence of the OC-SVM model's inability to distinguish the two digits, and it represents a typical behaviour when dealing with image classification using MNIST. A similar result occurs for G_5 when $p = 36$ (Table Id), in which the union of I_5 and I_8 is detected as single community. In this case, recalling that a higher value of p leads to smaller local datasets for the clients, it is reasonable that for $p = 36$ the local models do not have enough samples and are no longer able to distinguish the two digits. We can observe the anticipation

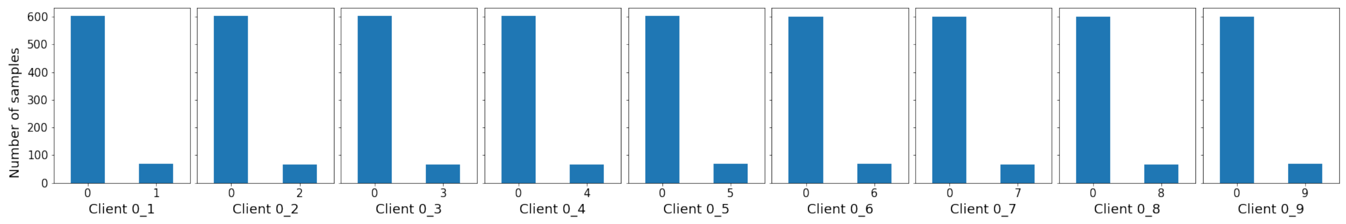


Fig. 1: Histograms of training data distribution for the group $C_{in} = 0$ (i.e., 0 is the common inlier class) with $p = 9$.

TABLE I: Community detection for MNIST

(a) $p = 9$

Community ID	Members
G_0	I_0
G_1	I_1
G_2	I_2
G_3	I_3
G_4	$I_4 \cup I_9$
G_5	I_5
G_6	I_6
G_7	I_7
G_8	I_8

(b) $p = 18$

Community ID	Members
G_0	I_0
G_1	I_1
G_2	I_2
G_3	I_3
G_4	$I_4 \cup I_9$
G_5	I_5
G_6	I_6
G_7	I_7
G_8	I_8

(c) $p = 27$

Community ID	Members
G_0	I_0
G_1	I_1
G_2	I_2
G_3	I_3
G_4	$I_4 \cup I_9$
G_5	$I_5 \cup m_{8,18}$
G_6	I_6
G_7	I_7
G_8	$I_8 \setminus m_{8,18}$

(d) $p = 36$

Community ID	Members
G_0	I_0
G_1	I_1
G_2	I_2
G_3	I_3
G_4	$I_4 \cup I_9$
G_5	$I_5 \cup I_8$
G_6	I_6
G_7	I_7
G_8	[illegible]

TABLE II: Community detection for fashion-MNIST

(a) $p = 9$

Community ID	Members
G_0	I_0
G_1	$I_1 \cup I_3$
G_2	$I_2 \cup I_4 \cup I_6$
G_3	I_5
G_4	I_6
G_5	I_7
G_6	I_8

(b) $p = 18$

Community ID	Members
G_0	$I_0 \cup I_2 \cup I_4 \cup I_6$
G_1	$I_1 \cup I_3$
G_2	$I_5 \setminus m_{5,6}$
G_3	I_6
G_4	I_7
G_5	I_8
G_6	$m_{5,6}$

(c) $p = 27$

Community ID	Members
G_0	$I_0 \cup I_2 \cup I_4 \cup I_6$
G_1	$I_1 \cup I_3$
G_2	I_5
G_3	I_6
G_4	I_7
G_5	I_8

(d) $p = 36$

Community ID	Members
G_0	$I_0 \cup I_2 \cup I_4 \cup I_6$
G_1	$I_1 \cup I_3$
G_2	I_5
G_3	I_6
G_4	I_7
G_5	I_8

of this behaviour when $p = 27$ in Table Ic in which the client $m_{8,18}$ mistakenly joins $I5$.

Similar considerations can be done for the fashion-MNIST case (Table II). Here the ideal groups of clients I_1 and I_3 are detected as a single community in the four testes. The same applies to the groups I_0, I_2, I_4, I_6, excluding the case $p = 9$ (Table IIa), in which I_0 is correctly isolated. This result is expected as fashion-MNIST is notably harder than MNIST.

D. Experimental result: federated outlier detection

We compare our methodology with two baselines: (i) **local**, where clients only train on local data; (ii) **ideal**, in which a client $m_{C_i,j}$ uses the model trained through federated learning on the set of clients I_{C_i}, i.e., the set of the clients sharing the same majority class. The test samples for each client are randomly sampled from the MNIST/fashion-MNIST test set, following the same inlier/outlier classes and the ratio of the corresponding client.

In Tables III and IV we show the test AUC score on MNIST and fashion-MNIST by varying the value of p, meaning that for each row we compute the average AUC score of $p|C|$ clients. Our methodology performs almost as the upper bound baseline, represented by the ideal federations of clients. Nevertheless, the results are consistent with the partitioning we obtain in the first step with the community detection that, especially for MNIST, identifies the right groups of clients in most of the cases. In the fashion-MNIST case, there are more exceptions to this behaviour. For instance, clients with different inlier classes all join a common group, as shown in Tabel IV (e.g., G_1). This affects the average AUC scores, which appear slightly less than the ideal upper bound (as opposed to nearly identical MNIST scores), but are still satisfactory.

More detailed results are shown in Tables V and VI, in which we only consider the detected communities that do not match the ideal cases. In these tables, each row corresponds to the average test AUC score for a fixed p and all the clients having

TABLE III: Test AUC on MNIST. For each p, mean $\pm$ std are computed on $p|C|$ clients

p	Local	Community (ours)	Ideal
9	0.773 ± 0.205	0.836 ± 0.18	0.839 ± 0.185
18	0.769 ± 0.207	0.835 ± 0.18	0.836 ± 0.181
27	0.77 ± 0.208	0.836 ± 0.18	0.84 ± 0.181
36	0.766 ± 0.207	0.819 ± 0.191	0.838 ± 0.182

TABLE IV: Test AUC on fashion-MNIST. For each p, mean $\pm$ std are computed on $p|C|$ clients

p	Local	Community (ours)	Ideal
9	0.714 ± 0.166	0.761 ± 0.161	0.772 ± 0.155
18	0.71 ± 0.173	0.747 ± 0.166	0.769 ± 0.155
27	0.706 ± 0.165	0.75 ± 0.162	0.765 ± 0.154
36	0.707 ± 0.166	0.749 ± 0.161	0.765 ± 0.151

TABLE V: Test AUC $\pm$ std on MNIST

p	C_{IN}	Local	Community (ours)	Ideal
9	4	0.749 ± 0.245	0.833 ± 0.197	0.833 ± 0.232
	9	0.823 ± 0.184	0.86 ± 0.159	0.881 ± 0.138
18	4	0.774 ± 0.2	0.819 ± 0.204	0.855 ± 0.19
	9	0.828 ± 0.176	0.872 ± 0.149	0.881 ± 0.139
27	4	0.762 ± 0.214	0.823 ± 0.208	0.84 ± 0.205
	9	0.836 ± 0.158	0.862 ± 0.161	0.882 ± 0.132
36	4	0.76 ± 0.215	0.799 ± 0.213	0.84 ± 0.201
	9	0.838 ± 0.156	0.862 ± 0.157	0.881 ± 0.13
	5	0.708 ± 0.194	0.718 ± 0.188	0.807 ± 0.177
	8	0.677 ± 0.196	0.696 ± 0.195	0.719 ± 0.219

TABLE VI: Test AUC $\pm$ std on MNIST

p	C_{IN}	Local	Community (ours)	Ideal
9	1	0.911 ± 0.051	0.94 ± 0.028	0.946 ± 0.025
	3	0.741 ± 0.127	0.788 ± 0.139	0.83 ± 0.094
	2	0.663 ± 0.146	0.686 ± 0.154	0.719 ± 0.125
	4	[illegible] ± 0.128	0.762 ± 0.13	0.782 ± 0.117
	6	0.642 ± 0.142	0.675 ± 0.144	0.698 ± 0.137
18	[illegible]	$0.913 \pm$ [illegible]	0.935 ± 0.036	0.944 ± 0.026
	3	$0.751 \pm$ [illegible]	0.792 ± 0.14	0.831 ± 0.082
	0	0.683 ± 0.125	0.742 ± 0.124	0.775 ± 0.089
	2	0.665 ± 0.153	0.667 ± 0.16	0.711 ± 0.126
	[illegible]	0.713 ± 0.142	0.724 ± 0.134	0.775 ± 0.115
	[illegible]	0.626 ± 0.142	0.68 ± 0.137	0.704 ± 0.133
27	1	0.907 ± 0.04	0.937 ± 0.033	0.944 ± 0.024
	[illegible]	0.74 ± 0.099	0.77 ± 0.164	0.813 ± 0.088
	0	0.688 ± 0.109	0.743 ± 0.101	0.773 ± 0.08
	2	0.674 ± 0.136	0.692 ± 0.145	0.763 ± 0.109
	4	0.71 ± 0.13	0.725 ± 0.117	0.777 ± 0.107
	6	0.63 ± 0.125	0.705 ± 0.126	0.714 ± 0.133
36	1	0.907 ± 0.041	0.936 ± 0.035	0.943 ± 0.024
	3	0.73 ± 0.118	0.762 ± 0.16	0.803 ± 0.093
	0	0.68 ± 0.113	0.754 ± 0.095	0.772 ± 0.078
	2	0.675 ± 0.127	0.694 ± 0.144	0.733 ± 0.123
	4	0.714 ± 0.132	0.743 ± 0.119	0.783 ± 0.107
	6	0.639 ± 0.131	0.698 ± 0.125	0.717 ± 0.127

majority class C_{IN}. The difference between the community (ours) and the ideal case is that in the former the clients of C_{IN} are trained through the corresponding federation G such that $C_{IN} \in G$ (Tables I and II), while in the latter they are trained through the perfect federation $C_{IN} = I_{IN}$.

As regards the MNIST case, we always obtain a community $G_4 = I_4 \cup I_9$ and, for $p = 36$, we have an additional community $G_5 = I_5 \cup I_8$. We ignore the one client mismatch in the $p = 27$ (Table Ic) as we verified that its influence is negligible. In Table V we observe that the clients with majority class $C_{IN} = 4$ still perform well with our methodology, with an average increase of 6% in the AUC score from the local case and an average decrease of 2% from the ideal case. $C_{IN} = 9$ scores end up approximately in the middle of the two bounds, highlighting, however, that the local case already reaches a good score of 0.83 for any p. $C_{IN} = 5$ is the only case that performs noticeably worse than the ideal case, with a decrease of 9% in the AUC score. However, also in this case there is a noticeable improvement over using the local models only.

For the fashion-MNIST case (Table VI), the scores are predictably lower than in the previous case; the gaps between the two bounds are generally tighter, but in any test, the scores of our methodology still fall in the middle. Clients of $C_{IN} = 1$ almost reach the ideal result, although the difference with the local one is minimal, while clients with $C_{IN} = 3$ have on average a $\sim 4\%$ increase/decrease on both the lower/upper baseline. Clients of $C_{IN} = 2$, $C_{IN} = 4$ have an average AUC score very close ($+1\%$) to the lower baseline for $p > 8$; this is precisely the value beyond which their federation is the union of four sets, i.e., $I_0 \cup I_2 \cup I_4 \cup I_6$, thus totalling four different majority classes. On the other hand, the remaining clients of this big federation, $C_{IN} = 0$ and $C_{IN} = 6$, are still able to reach a $\sim 7\%$ increase on the local case and be very close to the ideal case.

VI. Conclusions and Future Work

In this paper we propose a new methodology for federated learning in unsupervised settings, particularly amenable for dynamic mobile environments without central coordination. We specifically focus on Anomaly Detection tasks to define the details and test the methodology. The methodology is composed by two sequential steps: in the first step we detect the communities of clients having similar majority patterns (i.e., inlier class); this is achieved by having the clients perform a preliminary inlier/outlier split of their local data through the training of an AD model. Two clients join the same community when both agree in the inliers/outliers proportion after exchanging their respective models and computing an inference step on their local data. Then, each of the resulting community collaboratively trains a NN-based anomaly detection model through the federated learning framework.

We tested our methodology on the MNIST and fashion-MNIST datasets; in most cases, the communities found match with the ideal groups of clients, which are used as an upper bound baseline in experimental part. When the ideal groups are not found, our methodology merges 2-4 ideal groups into one community; it occurs in two MNIST classes, obtaining 9 groups, and in 6 fashion-MINIST classes, obtaining 6 groups in the worst case. The aggregation usually occurs for clients having similar majority classes (e.g., 4 and 9 in the case of MNIST).

We finally test the resulting AD federated models trained by the detected communities in term of AUC score, with local test sets on each client. In both cases, the results show clear advantage over the models locally trained (i.e., the lower baseline), while the performance is comparable with the federated models of ideal communities' partition, even for detected communities in which different majority classes are merged. This indicates that, even though we may not always be able to group clients as in the ideal (supervised) case, still the accuracy of the resulting model is close to optimal, and significantly better than using local models trained only on local data.

Future directions can involve several aspects of the proposed solution. Firstly, the optimization of the community detection phase, i.e., the all-to-all exchange of the local models may be suboptimal for high numbers of clients. Moreover, another possible improvement is the selection of the specific algorithms used to train local and federated models. For example, the "flat" fully connected autoencoder we use for the federated training may be too simple; as an example, when dealing with images, convolutional autoencoders may be introduced.

Finally, we aim to frame this solution in a more general context of anomaly detection in which normal data belong to multiple classes, in contrast to the typical AD task that only involves a single inlier class.

Acknowledgment

This work has been partly funded under the H2020 MARVEL (grant 957337), HumaneAI-Net (grant 952026), SoBigData++ (grant 871042) and CHIST-ERA SAI (grant CHIST-ERA-19-XAI-010, by MUR, FWF, EPSRC, NCN, ETAg, BNSF).

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[2] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[3] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[4] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[5] L. U. Khan, S. R. Pandey, N. H. Tran, W. Saad, Z. Han, M. N. Nguyen, and C. S. Hong, "Federated learning for edge networks: Resource optimization and incentive mechanism," *IEEE Communications Magazine*, vol. 58, no. 10, pp. 88–93, 2020.

[6] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving google keyboard query suggestions," *arXiv preprint arXiv:1812.02903*, 2018.

[7] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021. [Online]. Available: http://dx.doi.org/10.1561/2200000083

[8] Y. LeCun, "The next ai revolution will not be supervised," 2018.

[9] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection," *ACM Computing Surveys*, vol. 41, pp. 1–58, 7 2009. [Online]. Available: https://dl.acm.org/doi/10.1145/1541880.1541882

[10] M. Nardi, L. Valerio, and A. Passarella, "Centralised vs decentralised anomaly detection: when local and imbalanced data are beneficial," in *Proceedings of the Third International Workshop on Learning with Imbalanced Domains: Theory and Applications*, ser. Proceedings of Machine Learning Research, N. Moniz, P. Branco, L. Torgo, N. Japkowicz, M. Woźniak, and S. Wang, Eds., vol. 154. PMLR, 17 Sep 2021, pp. 7–20. [Online]. Available: https://proceedings.mlr.press/v154/nardi21a.html

[11] K. Ota, M. S. Dao, V. Mezaris, and F. G. B. D. Natale, "Deep learning for mobile multimedia: A survey," vol. 13, no. 3, pp. 1–22. [Online]. Available: https://dl.acm.org/doi/10.1145/3092831

[12] K. Chahal, M. S. Grover, and K. Dey, "A hitchhiker's guide on distributed training of deep neural networks." [Online]. Available: http://arxiv.org/abs/1810.11787

[13] T. Ben-Nun and T. Hoefler, "Demystifying parallel and distributed deep learning: An in-depth concurrency analysis." [Online]. Available: http://arxiv.org/abs/1802.09941

[14] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," vol. 53, no. 2, pp. 30:1–30:33. [Online]. Available: https://doi.org/10.1145/3377454

[15] T. Tuor, S. Wang, K. K. Leung, and K. Chan, "Distributed machine learning in coalition environments: Overview of techniques," in *2018 21st International Conference on Information Fusion (FUSION)*, pp. 814–821.

[16] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.

[17] W. A. Group, "Federated learning white paper v1." [Online]. Available: https://aisp-1251170195.cos.ap-hongkong.myqcloud.com/fedweb/1552917186945.pdf

[18] B. van Berlo, A. Saeed, and T. Ozcelebi, "Towards federated unsupervised representation learning," in *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, 2020, pp. 31–36.

[19] F. Zhang, K. Kuang, Z. You, T. Shen, J. Xiao, Y. Zhang, C. Wu, Y. Zhuang, and X. Li, "Federated unsupervised representation learning," *arXiv preprint arXiv:2010.08982*, 2020.

[20] E. Tzinis, J. Casebeer, Z. Wang, and P. Smaragdis, "Separate but together: Unsupervised federated learning for speech enhancement from non-iid data," *arXiv preprint arXiv:2105.04727*, 2021.

[21] T. Luo and S. G. Nagarajany, "Distributed anomaly detection using autoencoder neural networks in WSN for IoT," Tech. Rep., 2018.

[22] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, "Federated-learning-based anomaly detection for iot security attacks," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2545–2554, 2021.

[23] C. C. Aggarwal, *Outlier Analysis*. Springer International Publishing, 2017.

[24] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," 2010.

[25] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

[26] S. Han, X. Hu, H. Huang, M. Jiang, and Y. Zhao, "Adbench: Anomaly detection benchmark," *arXiv preprint arXiv:2206.09426*, 2022.

[27] B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," *Advances in neural information processing systems*, vol. 12, 1999.

D Federated Feature Selection for Cyber-Physical Systems of Systems

The appended paper follows.

Federated Feature Selection for Cyber-Physical Systems of Systems

Pietro Cassará, Alberto Gotta, *Member, IEEE*, and Lorenzo Valerio

***Abstract*—Autonomous vehicles (AVs) generate a massive amount of multi-modal data that once collected and processed through Machine Learning algorithms, enable AI-based services at the Edge. In fact, only a subset of the collected data present informative attributes to be exploited at the Edge. Therefore, extracting such a subset is of utmost importance to limit computation and communication workloads. Doing that in a distributed manner imposes the AVs to cooperate in finding an agreement on which attributes should be sent to the Edge. In this work, we address such a problem by proposing a federated feature selection (FFS) algorithm where the AVs collaborate to filter out, iteratively, the less relevant attributes in a distributed manner, without any exchange of raw data, thought two different components: a Mutual-Information-based feature selection algorithm run by the AVs and a novel aggregation function based on the Bayes theorem executed on the Edge. The FFS algorithm has been tested on two reference datasets: MAV with images and inertial measurements of a monitored vehicle, WESAD with a collection of samples from biophysical sensors to monitor a relative passenger. The numerical results show that the AVs converge to a minimum achievable subset of features with both the datasets, i.e., 24 out of 2166 (99%) in MAV and 4 out of [illegible] (50%) in WESAD, respectively, preserving the informative content of data.**

***Index Terms*—Artificial intelligence, autonomous system, feature selection, federated learning, human state monitoring, Internet of things, machine learning.**

I. Introduction

AUTOMATION enables a Cyber Physical System of Systems (CPSoS) to run with a minimum human assistance and evolves into autonomy when the human is taken out of the sensing, decision, and actuation loop. Automation can be used to operate a CPSoS comprising complex, dynamic, virtual and physical resources, such as telecommunication networks, computing units, software, sensors, and machines [1]. Humans can interact with an autonomous system either as passive end-users (such as passengers in autonomous transportation system) or rather as active co-operators in a mutual empowerment relationship towards a shared goal. Such cooperative, connected, and autonomous systems have the potential to be a game-changer in multiple domains if they will be capable of positively exploiting such an inescapable human factor. The increasing development of semi-Autonomous Driving Systems (ADSs) poses the challenge of taking the end-user, in the middle of the evolution process toward fully ADSs. Aside from vehicle control, a CPSoS needs to monitor the comfort/discomfort of the passenger, as well, to improve its well-being and to acknowledges the degree of safety and satisfaction perceived about the ADS. Artificial Intelligence (AI) is a fundamental technology for deploying the future CPSoS for ADSs [2]. The stringent computational and memory requirements for Machine Learning (ML) algorithms will impose a significant rethinking of the underlying computing and communication system and will have to fit the constraints of the onboard units. Information extraction should follow as much as possible optimal criteria, cooperating with the inherently distributed nature of the automotive scenario.

Moreover, local processing of information can also be an advantage in specific scenarios with intermittent connectivity or when data privacy is a key issue [3]. Hence, reducing the transfer time needed of either raw data or the relative features is of the utmost importance in determining the performance of computation offloading. Intuitively, traditional data compression techniques [4] could reduce such a delay component, but will also degrade the relative classification performance [5], prolonging the training phases as well as degrading the inference performance.

Conversely, when information extraction algorithms produce massive streams of features, selecting the most relevant ones to feed a ML model becomes very convenient, both in terms of compression and accuracy preservation. Such an operation is known as Feature Selection (FS) [6] and allows for achieving simpler and, therefore, more efficient ML-based models [7].

This work focuses on feature selection efficiency within a fleet of Autonomous Vehicles (AVs), which collect, through their sensors, multi-modal raw measurements. Collected data need to be pre-processed and delivered to feed a remote edge server for inference tasks. Such a procedure can introduce information redundancy, which leads to a waste of computing and communication resources. The AV ensemble aims at limiting the transmission to the top relevant features only. However, just a subset of the top-features can be extracted from each local data collection w.r.t. the whole top-set extracted from the union of all the local datasets but in a centralized manner. In fact, the former case may lead to an inconsistent model *w.r.t.* to the

Manuscript received 8 September 2021; revised 15 February 2022; accepted 2 May 2022. Date of publication 27 May 2022; date of current version 19 September 2022. This work was supported in part by MIUR PON Project OK-INSAID under Grant GA #ARS01_00917, and in part by H2020 Projects TEACHING under Grant GA #871385, in part by HumanAI-Net under Grant GA #952026, in part by MARVEL under Grant GA #957337, and in part by SoBigData++ under Grant GA #871042. The review of this article was coordinated by Dr. Tomaso De Cola. *(Corresponding author: Pietro Cassarà.)*

Pietro Cassará and Alberto Gotta are with the National Research Council, Institute of Information Science and Technologies, 56124 Pisa, Italy (e-mail: pietro.cassara@isti.cnr.it; alberto.gotta@isti.cnr.it).

Lorenzo Valerio is with the National Research Council, Institute of Informatics and Telematics, 56124 Pisa, Italy (e-mail: lorenzo.valerio@iit.cnr.it).

Digital Object Identifier 10.1109/TVT.2022.3178612

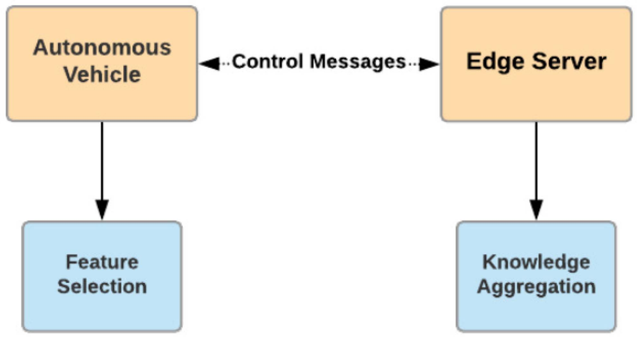


Fig. 1. Feature selection and aggregation components of the proposed FFS system.

latter. Therefore, the AVs, shall participate to a collaborative FS process, in order to exploit the whole information in a federated manner. We tackle this problem, by proposing, for the first time, a Federated-Feature Selection (FFS) algorithm, exploiting a distributed computing paradigm applied to AVs. In FFS all AVs collaborate to come up with the minimal set of features selected from their local datasets.

The proposed FFS system is made up of two components provided in Fig. 1:

- a local FS process runs on each AV and aims at generating a local distribution probability that ranks the information associated to a given feature, according to the Mutual Information (MI) metric [8], [9], which is solved by using the Cross-Entropy (CE) [10].
- An aggregation algorithm executed on the Edge Server (ES) that combines the local estimates received from the AVs. The aggregation algorithm is based on a Bayesian approach to merge the local information into a global one.

The messages delivered by AVs contain probability vectors where each element is the probability to select that feature. The ES returns the "federated" probability vector which is derived by the aggregation of the vectors received by the AVs, as detailed in the following, to replace each of the local ones. Note that, the proposed approach does not need to share any local raw data but only the estimates of the local most informative features. Moreover, it guarantees that all the AVs reach a consensus on the subset of the most informative features, after a finite number of communication rounds, i.e., the messages exchanged between the AVs and the ES.

As we show in the paper, the proposed algorithm *(i)* significantly limits the control messages exchanged during the FFS process and *(ii)* provably let the AVs converge to a subset of top features, which effectively reduce the information stored and transmitted by the AVs. Specifically, numerical results show that, on reference benchmarks, our solution limits data processing and transmission, by removing up the to 99% of redundant features from the selected datasets, without loss of accuracy on the learning model.

Summarising, the novel contributions of this paper are:

- A novel FFS algorithm based on the MI CE (client-side) on the AV and a Bayesian aggregation approach on the ES.
- The theoretical proof that such an algorithm converges to a stable solution in a fixed number of iterations.
- An extensive numerical evaluation tested on two real-world datasets that shows the efficiency of our solution.

The paper is organized as follows: related works are presented in Section II; the reference scenario and the system assumptions are presented in Section III; the theoretical background underlying the proposed feature selection approach is presented in Section IV; the federated version of the feature selection algorithm in presented in Section V; Section VI presents the experimental results of a study case with two real world datasets, belonging to different application domains; conclusions in Section VII.

II. Related Works

A. Feature Selection

Many FS procedures have been proposed in the literature. In [6], [11], [12] authors provide a comprehensive overview of the existing methods. Additionally, they consider the most important application domains and review comparative studies on feature selection therein, in order to investigate, which methods outperform for specific tasks. Authors highlight that FS is based on the identification of the relevance and redundancy provided by the features with respect to a class attribute function. The main approaches of FS fall into three categories: filtering, wrapping, and embedded methods. This categorisation is based on the interaction between the selected features and the learning model adopted to take a decision. The output of the wrapping and embedded methods is tightly connected to the learning model that uses the selection. Therefore, with these methods FS and model training cannot be uncoupled. Conversely, filtering methods are suitable for being used regardless the presence of a learning model to train.

As shown in [6], [11], [12], most of the well-known filtering algorithms use information-based metrics for FS, and can deal with samples of variable lengths, as presented in [13], [14]. A suitable information-based metric for the FS is the MI. MI has gained increasing popularity in data mining, for its ease to use, effectiveness, and strong theoretical foundation. mRMR [15] and HJMI [16] are some of the most used methods that exploit MI. These approaches rank the features according to the maximization of the MI and let the user to select a desired subset k. Differently, the proposed algorithm automatically select a minimal subset of relevant features, also capturing the mutual dependencies. Note that the formulation of the underlying optimization problem is NP-Hard [8], [9], i.e., MI-based feature selection problem involves the integer programming or, in some cases, the quadratic integer programming. In [17]–[19] authors show how to adopt the CE approach to address such native computational complex problems, for different application scenarios. Beyond MI, other filtering methods can use different metrics, such as in [20] where the authors evaluate the variance of all the features to measure the impact that each of them has on the learning process. This method relies on the concept that the features with zero variance add no information, by considering the relation between the target variable and feature vectors.

To the best of our knowledge, all these algorithms are designed for being executed in a centralised setting, i.e., under the assumption that the whole dataset is available to the learning agent.

B. *Distributed Learning*

Distributed learning is considered from several perspectives in the literature. A very consistent body of work deals with distributed learning based on the Federated Learning (FL) framework. FL is a distributed learning framework initially proposed by Google, where a large number of mobile or edge devices participate in a collective and distributed training of a shared model. [21], [22]. FL is an iterative procedure spanning over several communication rounds until the convergence is reached. Based on this paradigm, several modifications have been proposed concerning (i) new distributed optimisation algorithms [23]–[26], and (ii) privacy-preserving methods for FL [27], [28]. Alternatively, other approaches do not rely on a centralised coordinating server. In [29], [30], authors propose a distributed and decentralised learning approach based on Hypothesis Transfer Learning. Similarly to the FL framework, authors assume that several devices hold a portion of a dataset to be analysed by some distributed machine learning algorithms. The aim of [29], [30] is to provide a learning procedure able to train, in a decentralised way, an accurate model while limiting the network traffic generated by the learning process. The vast majority of the distributed learning solutions, presented in the literature, focus on the model's training, giving the feature engineering phase for granted. Until now, the idea of performing FS, directly, on edge devices remains unexplored.

In the literature only few approaches cope with FS in distributed settings. In [31], authors present a distributed algorithm for FS based on the Intermediate Representation, which aims at preserving the privacy of data, allowing the node to exchange each other the data they hold. Therefore, in this method FS is performed under the assumption that all data are available to the FS algorithm. Moreover, the method presented by the author depends from the specific learning model that uses the selected features.

In [32], the authors propose an information-theoretic FFS approach called Fed-FiS. Fed-FiS estimates feature-feature mutual information and feature-class mutual information to generate a local feature subset in each user device. Then a central server ranks each feature and generates a global dominant feature subset using a classification approach. This approach has some commonalities with ours, such as the adopted metric (MI) and the federated settings. However, differently from [32] *(i)* we provide directly the minimum set of relevant features instead of a ranking, *(ii)* we propose an aggregation based on Bayes' theorem that does not rely on any Machine Learning scheme to finalise the selection (i.e., no regression or classification methods are adopted in our solution), resulting in a computationally more suitable approach for vehicular scenarios.

In light of this and to the best of our knowledge, this is the first paper that proposes a federated mechanism of feature selection explicitly designed to meet the requirements of the CPSoS context.

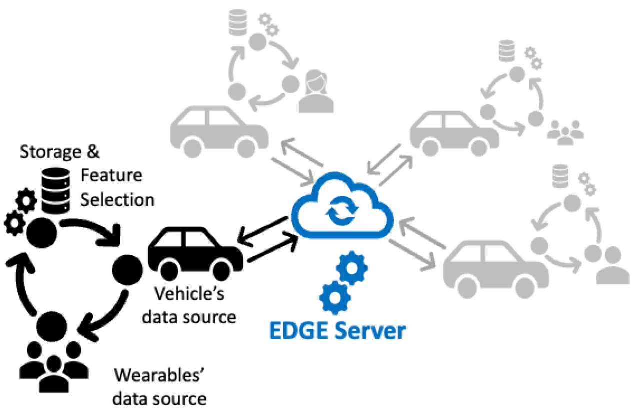


Fig. 2. System architecture. Data sources characterize two different Cyber Physical Systems (CPSs): the former that monitors the user through wearable sensors, the latter relative to the ADS.

III. SYSTEM ASSUMPTIONS

In this section we describe the reference scenario and the system assumptions considered in this paper. As shown in Fig. 2, we consider a set of AVs implementing an ADS each, collecting data generated by the sensors integrated in a CPSoS and that collaborates with the others ADSs to learn a minimal, and most informative set of features from their local datasets. To this end, the AVs execute an in-network data filtering process through our FFS approach to reach a consensus in identifying the most informative feature subset. Finally, the globally shared feature set is used like a compression scheme before transmitting it to an ES. Note that, in this system the AVs are only responsible for finding the best compression scheme applicable to the their local data in a collaborative way, based only on the control information they exchange with the ES. Moreover, the ES has a three-fold role: i) it acts as central coordinating entity in the FFS process whose purpose is to aggregate the partial control information sent by the AVs; ii) it acts as final collector for the compressed data, once the FFS is completed and, iii) runs the AI services to extract knowledge from data but that is used only for performance evaluation in this paper. We target two different user cases to validate the performance of the proposed FFS method. The former refers to the localization of an AV in the environment based on images and inertial measurements, and the latter regards the physiological-state monitoring of a passenger in the automotive domain. We define two different sub-systems part of the same CPSoS: the ADS of above, and an Human State Monitoring System (HSMS) to learn the feeling perceived from a passenger relatively to the ADS driving style. Therefore, we assume each AV to be equipped with a camera to capture images from the surrounding environment aside some inertial sensors for the former learning task, and a set of body sensors, such as, Electrocardiography (ECG), Electrodermal Activity (EDA), Electromyography (EMG), and Respiration (RSP) for the latter.

Each AV is able to locally synchronize the multi-sensory data such that, for each image, it is possible to associate the corresponding inertial measurements leading to an *enhanced Raw Input Datum* (eRID). Note that for the scope of this paper it is not important the specific semantic of the labelling, but it is enough to assume a labelling process on the collected data. The AVs are also equipped with a relatively small edge computing unit (e.g., a RaspeberryPi or, at most, an Nvidia Jetson Nano) able to cache data and execute the FS task, before transmitting the features. Additionally, the AVs are endowed with a radio communication interface to communicate toward the ES. It must be noted that the task is not collecting images of the environment, or physiological parameters of the user but, conversely, retrieving the information associated to those images or to those physiological sensors, e,g., the position of the AV with respect to the surrounding or the user mood. In particular, the latter is labelled according to the classification scale provided by questionnaires like PANAS, SSSQ or SAM [33], which associates numerical labels to the physiological states.

IV. Feature Selection

In this section, we provide the theoretical background of the MI-based FS algorithm and the relative implementation based on the CE method.

A. Background Feature Selection Based on Mutual Information

To make the paper self-contained, we report in this Section the necessary theoretical background needed to get an intuition about the internal details of the CE-based FS method presented in Section IV-B.

First, let us define the FS problem as follows:

Definition (FS Problem): Given the input data matrix $\mathbf{X}$ composed by n samples of m features ($\mathbf{X} \in \mathbb{R}^{n\times m}$), and the target attributes' (or labels) vector $\mathbf{y} \in \mathbb{R}^n$, the FS problem is to find a k-dimensional subset $\mathbf{U} \subseteq \mathbf{X}$ with $k \leq m$, by which we can characterize $\mathbf{y}$.

The method we adopt in the paper performs the FS measuring, through the Mutual Information metric, the amount of information that a subset of features (or attributes) $\mathbf{U}$ expresses with respect to a specific target label $\mathbf{y}$.

Formally, the MI between random variables can be defined as [34], [35]:

$$\mathbf{I}(\mathbf{U};\mathbf{y}) = \mathbf{H}(\mathbf{y}) - \mathbf{H}(\mathbf{y}|\mathbf{U}), \tag{1}$$

where $\mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_k \mid k \leq m\} \subseteq \mathbf{X}$, and $\mathbf{H}(\mathbf{y}|\mathbf{U})$ is the conditional entropy which measures the amount of information needed to describe $\mathbf{y}$, conditioned by the information carried by $\mathbf{U}$. Hence, $\mathbf{I}(\mathbf{U};\mathbf{y})$ represents the dependence between $\mathbf{U}$ and $\mathbf{y}$, i.e., the greater the value of $\mathbf{I}$, the greater the information carried by $\mathbf{U}$ on $\mathbf{y}$. We recall that the MI between two random variables $\mathbf{A}$ and $\mathbf{B}$ is strictly related to the entropy $\mathbf{H}(\cdot)$, which defines the amount of information held by the variables, i.e., the entropy of a random variable $\mathbf{A}$ (i.e., $\mathbf{H}(\mathbf{A})$) and its probability are inversely proportional: the greater the entropy of a random variable $\mathbf{A}$, the greater its unpredictability and vice-versa. Hence, we can assert that the entropy measures the diversity of $\mathbf{A}$ in terms of the uncertainty of its outcomes.

In MI-based FS the features to be selected are those that maximise (1). These features are typically referred as Essential Attributes (EAs). By solving the following optimization problem

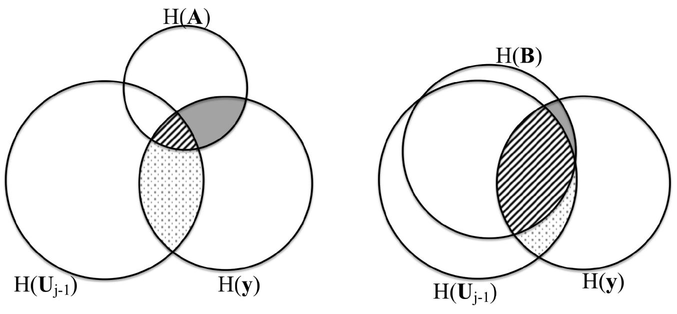


Fig. 3. Example of the relationship between Mutual Information and Entropy.

we would obtain the optimal global solution to the FS problem defined in IV-A:

$$\arg\max_{\mathbf{U}} \mathbf{I}(\mathbf{U};\mathbf{y})$$
$$\mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_k \mid k \leq m\} \subseteq \mathbf{X} \tag{2}$$

Note that the problem (2) belongs to the class of Integer Programming (IP) optimization problems and finding its optimal solution is NP-hard [36], i.e., the optimal solution $\mathbf{U}$ would be found among all combinations of feature indices of the native set $\mathbf{X}$.

The problem (2) becomes computationally tractable if approached through an iterative algorithm which selects and adds to the subset $\mathbf{U}$ one feature at a time. Therefore, instead of solving 2, we address the problem defined in (3):

$$\arg\max_{\mathbf{x}_j \in \mathbf{X}\setminus\mathbf{U}} \mathbf{I}(\mathbf{x}_j;\mathbf{y}|\mathbf{U}),$$
$$\mathbf{U} = \{\mathbf{x}_1 \cdots \mathbf{x}_{k-1} \mid k \leq m\} \subseteq \mathbf{X}. \tag{3}$$

For the sake of clarity, we provide an intuitive example based on the relation between MI and the entropy. Considering Fig. 3, the circles are the entropy of the random variables $\mathbf{A}, \mathbf{B}, \mathbf{U}, \mathbf{y}$, and the grey regions are the information carried by the variable $\mathbf{A}$ (or $\mathbf{B}$) on $\mathbf{y}$. The dashed area shows the information redundancy of the variable $\mathbf{A}$ (or $\mathbf{B}$) given the already selected variables in $\mathbf{U}_{j-1}$. In this example, the variable $\mathbf{A}$ should be added to the set $\mathbf{U}$ since it is more informative than $\mathbf{B}$ on $\mathbf{y}$, i.e., its grey area is larger than $\mathbf{B}$'s, and it is less redundant than $\mathbf{B}$ w.r.t. to $\mathbf{U}_{j-1}$.

The main drawback of this approach is that it might end up with a sub-optimal solution because, by selecting the features one by one, the algorithm makes the implicit assumption that they are independent, which might not hold true. Theoretical foundations for the incremental version of the FS algorithms has been proven by the authors in [34], [35]. It is worth mentioning that a connected issue with problem (3) regards the efficient evaluation of the MI, which might become prohibitive even for datasets with a small number of samples. We overcome this problem by adopting the MIToolbox [37], a state-of-the-art tool for numerical optimization.

B. CE-Based Feature Selection Algorithm

In this section, we describe the CE-based algorithm that finds, in a finite number of steps, a solution that well approximates the one found by solving problem (2), while making negligible the assumption of independence among features introduced in problem (3). In other words, with CE-based FS, instead of selecting one EAs at a time, we select a set of EAss *jointly*.

The CE-based algorithm is based on the following intuition: if the set $\mathbf{U}$ contains only EAss, then $\mathbf{I}(\mathbf{U};\mathbf{y}) \to \mathbf{H}(\mathbf{y})$, which implies that $\mathbf{H}(\mathbf{y}|\mathbf{U}) \to 0$ [34], [35]. Note that with our approach, we avoid the greedy research of the set $\mathbf{U}$ among all the possible $\binom{m}{k}$ solutions which realizes $\mathbf{H}(\mathbf{y}|\mathbf{U}) \to 0$. Instead, we adopt the stochastic approach. Precisely, we associate each i-th feature with a random variable $z_i \sim \text{Bernoulli}(p_i)$. The CE-based algorithm identifies which variables $z_i,\ i = 1, \ldots, m$ must have $p_i \to 1$, so that the objective function $\mathcal{O}(\mathbf{U}(\mathbf{z})) = \mathbf{H}(\mathbf{y}|\mathbf{U})$ gets close to 0. This is called Associated Stochastic Problem (ASP) [10]. In this way, we get the optimal distribution of the binary vector $\mathbf{z}$ through which we identify the features to be selected, i.e. the i-th feature is selected if $p_i \to 1$. It is worth noting that searching for the solution of the optimization problem through the definition of the ASP has the advantage of addressing the native problem in (2) as a convex problem.[1]

We formulate the ASP as a minimization problem, as shown in (5). In the following we present the essential steps that brings to its formulation. Briefly, we need to find the probability distribution $g(\mathbf{z},\mathbf{p})$ of the values in $\mathbf{z}$ equal to 1 that solves the equation:

$$\Pr(\mathcal{O}(\mathbf{U}(\mathbf{z})) \le \gamma) = \sum_{\{\mathbf{z}\}} \mathcal{I}(\mathcal{O}(\mathbf{U}(\mathbf{z})) \le \gamma)\, g(\mathbf{z},\mathbf{p}) \quad (4)$$

where $\mathcal{I}(\cdot)$ is the indicator function of the event $\mathcal{O}(\mathbf{U}(\mathbf{z})) \le \gamma$, and γ is the minimum value for our objective function. Precisely, γ at step t is calculated as the percentile $1-\beta$ of the objective function calculated by using the samples drawn from the distribution $g(\mathbf{z},\mathbf{p})$ at step t. Note that, the authors in [10] recommend to set β in the range $0.9 - 0.95$. The indicator function is equal to 1 for all the possible configurations in $\mathbf{z}$ that verify the event $\mathcal{O}(\mathbf{U}(\mathbf{z})) \le \gamma$, and 0 otherwise.

We estimate $g(\mathbf{z},\mathbf{p})$ through the Likelihood Ratio (LR) estimator with reference parameter $\mathbf{p}$. Precisely, we apply the LR theory of estimation [10] to define the following optimization problem and to obtain the optimal value $\mathbf{p}^*$ for the distribution.

$$\mathbf{p}^* = arg\min_{\mathbf{p}} \frac{1}{S}\sum_{j=1}^{S} \mathcal{I}(\mathcal{O}(\mathbf{U}(\mathbf{z}_j)) \le \gamma)\ \ln(g(\mathbf{z}_j,\mathbf{p})) \quad (5)$$

where $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_S\}$ is a set of possible samples drawn from the distribution $g(\mathbf{z},\mathbf{p})$.

As stated above $\mathbf{z}_j = [z_{1j} \cdots z_{mj}]$ is a vector of independent Bernoulli random variables where z_{ij} takes value equal to 1 with probability p_i and 0 with probability $1-p_i$. Hence, $g(\mathbf{z}_j,\mathbf{p})$ can be written as:

$$g(\mathbf{z}_j,\mathbf{p}) = \prod_{i=1}^{m} p_i^{z_{ij}}(1-p_i)^{(1-z_{ij})}\ ;\ z_{ij} \in \{0,1\} \quad (6)$$

Given that the objective function of problem (5) is concave,[2] we can solve it in closed form by imposing:

$$\frac{\partial}{\partial p_i}\frac{1}{S}\sum_{j=1}^{S} \mathcal{I}(\mathcal{O}(\mathbf{U}(\mathbf{z}_j)) \le \gamma)\ \ln(g(\mathbf{z}_j,\mathbf{p})) = 0,$$

leading to:

$$p_i = \frac{\sum_{j=1}^{S} \mathcal{I}(\mathcal{O}(\mathbf{U}(\mathbf{z}_j)) \le \gamma) z_{ij}}{\sum_{j=1}^{S} \mathcal{I}(\mathcal{O}(\mathbf{U}(\mathbf{z}_j)) \le \gamma)} \quad i = 1 \cdots m; \quad (7)$$

In the CE-base algorithm the result in the (7) is used for updating the distribution $\mathbf{p}$ as follows:

$$p_i = (1-\alpha)p_i + \alpha\frac{\sum_{j=1}^{S} \mathcal{I}(\mathcal{O}(\mathbf{U}(\mathbf{z}_j)) \le \gamma) z_{ij}}{\sum_{j=1}^{S} \mathcal{I}(\mathcal{O}(\mathbf{U}(\mathbf{z}_j)) \le \gamma)}. \quad (8)$$

The mathematical analysis about the choice of the parameter α is provided in the Appendix VII-A of this work. Further indications on the choice of α can be found in [10], [38], [39]. The derivation of equations (5-7), as well as, the optimality of $g(\mathbf{z}_j,\mathbf{p})$ are proven in [10].

The solution of the problem defined in (5) is achieved through Algorithm 1: it starts with an initial guess of $\mathbf{p}_G$; S Bernoulli random samples of size m each (line 4) are drawn at each step t. For each sample $\mathbf{z}_s$, the values of the conditional entropy (line 7) are computed on the dataset where the only active features are those corresponding to the elements equal to one (lines [illegible]) in $\mathbf{z}_s$. The subset selection is shown in the procedure GETSUBSET$(\mathbf{X},\mathbf{z})$ (lines 15-26). Then we compute $\mathbf{p}(\mathbf{Z}_t)$ (lines 9-10) as in (7) and finally we update the current estimate of the probability vector $\mathbf{p}$ (line 11) as in (8).

V. FEDERATED FEATURE SELECTION

In this section we present how we exploit the CE-based FS algorithm presented in Section IV and summarised in Algorithm 1 to design our FFS algorithm FFS, described in Algorithms 2 and 3. They cover, respectively, the two functional blocks of FFS, i.e., Algorithm 2 is executed by the ES to coordinate the distributed FS and Algorithm 3 runs on the clients. The FFS is an iterative procedure. At the beginning, the ES sends to the clients involved in the process a vector $\mathbf{p}_G \in \mathbb{R}^m$ where each element represents the probability that each feature has to be selected according to its importance (lines 8-10 of Alg.2). Each element of $\mathbf{p}_G$ is initialized to 0.5, i.e., this is a common choice when using the CE algorithm. The vector $\mathbf{p}_G$ represents a piece of global information that the ES shares with the client nodes. Each client l uses $\mathbf{p}_G$ to initialize its local copy of the probability vector, i.e., $\mathbf{p}_l \leftarrow \mathbf{p}_G$ and runs the local FS procedure based on its local data (lines 2-3 of Algorithm 3). At the end of the local FS, the l-th client sends to the ES the locally updated probability vector $\mathbf{p}_{l_{new}}$ and a control information regarding the cardinality of its local data n_l whose purpose will become clear in the following. The ES computes the new global probability vector (line 13 of Algorithm 2) by aggregating the ones received

[1]More details are in Section 4 of [10].

[2]The logarithm is a concave function, the indicator function is 0 or 1 so the weighted sum of concave functions gives still a concave function.

Algorithm 1: CE-based Algorithm for FS.

```
 1: procedureCE$\mathbf{X}, \mathbf{y}, \mathbf{p}$, T,S
 2:   for all $t = 1, \ldots, T$ do
 3:     $\mathbf{Z}_t \leftarrow$ GENRNDSAMPLE $(S, \mathbf{p})$          $\triangleright$ $\mathbf{Z} \in \{0,1\}^{S \times m}$
 4:     $\mathbf{u} \leftarrow \{\}$
 5:     for all $\mathbf{z}_s \in \mathbf{Z}_t$ do                  $\triangleright \mathbf{z}_s \in \{0,1\}^{1 \times m}$
 6:       $\mathbf{U} \leftarrow$ GETSUBSET$(\mathbf{X}, \mathbf{z}_s)$
 7:       $\mathbf{u} \leftarrow \mathbf{u} \cup \mathbf{H}(\mathbf{y}|\mathbf{U})$
 8:     end for
 9:     $\gamma \leftarrow$ COMPUTEPERCENTILE$(\mathbf{u}, 1 - \beta)$
10:     $\mathbf{p}(\mathbf{Z}_t) \leftarrow$ COMPUTENEWPROB$(\mathbf{u}, \gamma, \alpha)$          $\triangleright$(7)
11:     $\mathbf{p} \leftarrow (1 - \alpha)\mathbf{p} + \alpha \mathbf{p}(\mathbf{Z}_t)$              $\triangleright$(8)
12:   end for
13:   return $\mathbf{p}$
14: end procedure
15: proceduregetSubset$\mathbf{X}, \mathbf{z}$
16:   $\mathbf{U} \leftarrow \{\}$
17:   for all $\mathbf{x} \in \mathbf{X}$ do
18:     $\mathbf{u} \leftarrow \{\}$
19:     for all $j = 1, \ldots, m$ do
20:       if $z_j == 1$ then
21:         $\mathbf{u} \leftarrow \mathbf{u} \cup x_j$
22:       end if
23:     end for
24:     $\mathbf{U} \leftarrow \mathbf{U} \cup \mathbf{u}$
25:   end for
26: end procedure
```

from the clients as expressed in (9) and discussed later on. The updated vector $\mathbf{p}_G$ is transmitted to the nodes that run Algorithm 3 by updating the local probability vector with the new global one. This procedure iterates until the distribution global probability vector converges to a stable one. In FFS we check convergence by comparing the distribution of the current global probability vector $\mathbf{p}_G$ to the previous one $\mathbf{p}_{G_{old}}$ using the Kolmogov-Smirnov statistical test for two one-dimensional samples (KS-test). The procedure stops when *(i)* the p-value of the KS-test is greater than a fixed threshold[3] $\tau_1 = 0.995$ and, *(ii)* its variation from the previous one is less than $\tau_2 = 10^{-6}$ (line 7 of Algorithm 2).

The core point of Algorithm 2 regards the aggregation step (line 13 of Algorithm 2) where the ES merges the local probability vectors into the global one which, in our solution, is defined as a weighted average. The main idea is to merge the local probability vectors by a weighted average where the weights (computed as in (10)) serve the twofold purpose of *(i)* considering more (or less) those vectors that are computed from larger local datasets and *(ii)* defining a common support among all the probability vectors. This second aspect is quite crucial for the consistency of the computation in (9).

Formally, we assume that each node acquires a number of *i.i.d.* records n_l to perform the FS, and that the nodes share the same set of features $\mathbf{X}$. The global probability $\mathbf{p}_G$ used for the FS can be written as follows:

$$\mathbf{p}_G = \sum_l \mathbf{p}_l \omega_l, \tag{9}$$

where $\mathbf{p}_l$ is the solution of problem (5) at node l obtained by using Algorithm 1, and ω_l weights $\mathbf{p}_l$ *w.r.t.* the other nodes, whose formal definition is:

$$\omega_l = \frac{n_l}{\sum_l n_l}. \tag{10}$$

As anticipated, according to (10), we weight the probability vector $\mathbf{p}_l$ of node l proportionally to the size of its local dataset compared to the whole amount of data present in the system. In this way, we can contrast situations where local datasets are heterogeneous w.r.t. the size.

In FFS, the updating scheme can be, at least in principle, both synchronous and asynchronous, provided that the set of nodes involved in the process does not change over time.[4] Precisely, we assume a system where the ES after having sent the updated global probability vector, expects the nodes to receive their local updates within a fixed time slot, after which, it begins the aggregation step using only the information received. Therefore, the number of updates used to compute the new global probability vector might change because a subset of nodes could not communicate their updates within the deadline set by the ES. Regardless of the number of nodes that contributed to the aggregation step during one round of communication, the ES broadcasts the new global probability vector $\mathbf{p}_G$ to *all* nodes in the system. In this way, all nodes start the new round of local computation from the same starting point, and, consequently, we dramatically limit the potentially detrimental effects deriving from the aggregation of outdated local probability vectors. Moreover, as proved by the convergence analysis provided in AppendixVII-A and AppendixVII-B, independently from the updating scheme, FFS converges in a finite number of steps to the very same solution as running the CE in centralised settings i.e., with complete access to the entire dataset.

It's worth noting that our solution is able to cope with feature redundancy in federated settings. Precisely, this represents an issue that might prevent the possibility of performing the FS in federated settings. In fact, running a standalone FS algorithm on different local datasets where there is redundancy between features, different FSs might occur but with an equivalent information content across all the AVs. This aspect makes all the local selections completely useless regarding the communication efficiency, due to the consequent lack of agreement on the FS between the AVs. Conversely, since in FFS the AVs share at each communication round their local information, they may come up with a final agreement on the FS. Summarising, even if there is redundancy between features, the final selection is consistent among all the AVs and, according to results presented in Section VI, it is also accurate if compared to the centralized FS (i.e., when all the local raw data are transferred onto the ES).

[3]We empirically observed that the closer τ_1 to one, the more accurate the solution.

[4]Note that this condition does not imply that all nodes must be active during the entire process. In fact, as we will show in Section VI our system is robust to the presence of churning nodes.

Algorithm 2: Server Side FFS Algorithm.

```
 1: procedure Server-Node
 2:   $v \leftarrow 0$          ▷p-value of Kolmogorov-Smirnov test
 3:   $\tau_1 \leftarrow .995$
 4:   $\tau_2 \leftarrow 10^{-6}$          ▷Thresholds to check convergence
 5:   $\mathbf{p}_G \leftarrow \{1/2 \mid \forall\, p_i\ i = 1, \ldots, m\}$
 6:   do
 7:     for all $l \in L$ do
 8:       SENDTOCLIENT($l, \mathbf{p}_G$)
 9:     end for
10:     RECEIVEFROMCLIENTS($\mathbf{p}_{l_{new}}, n_l$)
11:     $\mathbf{p}_{G_{old}} \leftarrow \mathbf{p}_G$
12:     $\mathbf{p}_G \leftarrow$
        UPDATEGLOBALPROBABILITY()          ▷(9)
13:     $v_{old} \leftarrow v$
14:     $v \leftarrow$ KOLMOGOROVSMIRNOVTEST ($\mathbf{p}_G, \mathbf{p}_{G_{old}}$)
15:   while $v \geq \tau_1 \wedge |v - v_{old}| \leq \tau_2$          ▷repeat until
      convergence is met
16: end procedure
```

Algorithm 3: Client side Federated Feature Selection algorithm.

```
1: procedure Client-Node
2:   $\mathbf{p}_l \leftarrow$ RECEIVEFROMSERVER($\mathbf{p}_G$)
3:   $\mathbf{p}_{l_{new}} \leftarrow$ CE($\mathbf{X}_l, \mathbf{y}_l, \mathbf{p}_l$,T,S)          ▷Algorithm 1
4:   SENDTOSERVER($\mathbf{p}_{l_{new}}, n_l$)
5: end procedure
```

VI. Numerical Evaluation

In this section, we present the numerical results of our compression method based on the FFS algorithm presented in Section V. Before going through the results, we introduce the datasets, the simulation settings, the methodology, and the metrics used to evaluate our solution's performance.

A. Dataset Description and Simulation Settings

We based the performance evaluation of FFS on two datasets, each one mapping one of the two use cases described in Section III. The first one called MAV[5] is a publicly available dataset containing both 64×64 images and 6 Inertial Measurement Units (IMUs) collected by a AV during a mission in a controlled environment. The second dataset called WEarable Stress and Affect Detection (WESAD) is a collection of data sampled from heterogeneous biophysical sensors: ECG, EDA, EMG, Temperature, Respiration and Inertial Measurements on the three axes.

a) MAV dataset: both images and inertial measurements are synchronised to obtain a set of eRIDs. We pre-process the raw images to extract more informative features as it is customary in the computer vision domain. Feature extraction eases the training of a machine learning model and, performs a preliminary step of data compression. In fact, a raw image is made of 4102 floats (64×64 pixels + 6 IMU readings) while, after the feature extraction, it shrinks down to a vector of size 2166 floats. In our settings, we extract the Histogram of Oriented Gradient (HOG) features,[6] and we assume that the feature extraction is accomplished directly on the AV, which might be possible if equipped with a board of the kind discussed in [40]. Note that the original dataset is unlabeled. Therefore we labelled it in a way compatible with the original context of positioning. To this end, we associated with each eRID a label corresponding to the corresponding voxel.[7] Table I shows the structure of an eRID for the MAV; the first 2160 feature are HOG while the last 6 are IMUs, i.e., acceleration (ACC) and angular velocity (AV). The whole dataset contains 2911 labelled records. To simulate the federated data collection, we split it into 10 disjoint partitions of 291 records such that each partition is i.i.d. w.r.t. the entire dataset. Each subset represents a AV. The data collection is slotted; hence, the AVs draw with replacement a random sample from their local dataset for each time slot. This sample is used to perform the local computation of the distributed algorithm followed by a communication round for synchronising the AVs on the local FS. Each random draw's size is accumulated to trace the cache necessary for storing data until the completion of the distributed FS.

TABLE I
Structure of a MAV eRID

0	1	2	...	2158	...
HOG #					
2160	2161	2162	2163	2164	2165
ACC$_x$	ACC$_y$	ACC$_z$	AV$_x$	AV$_y$	AV$_z$

TABLE II
Structure of a WESAD eRID

0	1	2	3	4	5	6	7
ACC$_x$	ACC$_y$	ACC$_z$	ECG	EMG	EDA	TEMP.	RSP

b) WESAD dataset it provides data in terms of features and labels already useful to perform the detection of stress and affection state of human subjects. The dataset contains readings from two devices, i.e., Respiban and Empatica E4, positioned i) on the chest and ii) on the wrist of human subjects. Each device is equipped with multiple sensors monitoring several physiological parameters. Since the two devices have different operating settings, we focused on the Respiban, whose collection rate is homogeneous for all its sensors. The dataset contains readings collected from 17 human subjects, which perform a predetermined protocol to induce the body in one of the following states: 0-baseline, 1-amusement, 2-stress, 3-meditation, 4-recovery. The data collected for each subject amounts to ∼3.6 M records, equivalent to ∼220 MB. A complete description of the dataset is provided in [33]. Table II shows the structure of an eRID for the WESAD.Due to the huge size of the dataset we used the data from 5 out of 17 subjects, corresponding to ∼1.1 GB.

[5]dataset. [Online]. Available: https://tinyurl.com/mavmr01

[6]HOG is a standard feature extraction methodology used in computer vision and image processing to create an image descriptor that captures the spatial relations between different portions of it [40].

[7]A voxel represents a value on a regular grid in three-dimensional space.

The data is already partitioned according the subject ID, thus we keep the original partitions. In our simulated scenario, each partition corresponds to an edge device holding the data of only one subject, i.e., no artificial data re-distribution is performed. As for the previous scenario, each device executes FFS using only its own data.

We evaluate the performance of our methodology according to two metrics:

- *accuracy:* to assess the quality of the distributed FS
- *network overhead (N_{OH})*: to evaluate the impact in terms of network traffic generated by our methodology

Our target is to compress the data to be transmitted, without significantly degrading its informative content.

Accuracy metric: The quality assessment is a two-stage procedure. First, we set the baseline validating the quality of the features selected by CE executed in a centralised setting, i.e., we train a classifier using the set of selected features (CE-CFS) on the entire dataset, and we compare its prediction performance with that of a second classifier trained on the whole set of features (NO-FS). If the CE-CFS performance on a smaller group of features is comparable or equivalent with the one identified by NO-FS, we consider the FS valid. To strengthen this initial evaluation, we compare the centralised results of CE-CFS with other three reference FS algorithms: mRMR [15], HJMI [16] and ANOVA [20]. As we will show in the following, for all these benchmarks we have to specify the size k of the feature selection. Since we are interested in assessing the quality of the FS and for the sake of fairness, we set k equal to the size of the FS obtained by CE-CFS (which finds such a number in a completely autonomous way).

Then, we repeat the same procedure training both classifier on the subset of features obtained from the FFS and we compare its performance with all the centralised methods. We split the dataset in train (80%) and test set (20%). The train set is used for both FS and model training, while the test is used for performance evaluation only. The accuracy is defined as the average of correctly classified records

$$A = \frac{1}{N}\sum_{i=1}^{N} I(\hat{y}_i = y_i), \tag{11}$$

where N is the size of the test set, I is the indicator function, $\hat{y}_i$ and y_i are the i-th predicted and true label, respectively. For the sake of statistical significance, the training is repeated ten times, changing the initialisation of the classifier and the composition of training and test set. The reported results are average values accompanied by confidence intervals at 95%.

Network Overhead: we measure the network traffic generated by our solution as follows. On the one hand, we compute the network overhead generated by the FFS network defined as:

$$N_{OH} = R * L * 2 * (z + 1 + b) \tag{12}$$

where R is the number of communication rounds before all the L AVs involved in the distributed FS converge to a solution, $z + 1$ is the number of nonzero floating point numbers belonging to the probability vector $\mathbf{p}_l$ in (9) exchanged between the AVs during each round plus the weight ω_l in (10). The symbol b is the size of the bit map used to reconstruct the position of the non-zero elements exchanged between the AVs and the edge server. On the other hand, we compute the compression obtained through the FS as:

$$C = |F|/|D| \tag{13}$$

where $F \subseteq D$ is the selected set, and D is the entire set of features.

TABLE III
COMPARISON BETWEEN NO-FS AND CE-CFS ON MAV AND WESAD DATASET

Dataset	Method	Size (# record)	FS (#)	C (%)	Accuracy (%)
	NO-FS	2911	2166 (All)	-	97.5±0.4
	CE	2911	18	99	96.7±0.5
MAV	MRMR	2911	k=18	99	95.0±0.5
	ANOVA	2911	k=18	99	95.0±0.4
	HJMI	2911	k=18	99	96.3±0.7
	NO-FS	$15{*}10^6$	8 (All)	-	94.3±0.7
	CE	$15{*}10^6$	4	50	94.6±0.8
WESAD	MRMR	$15{*}10^6$	k=4	50	94.3±0.8
	ANOVA	$15{*}10^6$	k=4	50	94.5±0.5
	HJMI	$15{*}10^6$	k=4	50	90.2±1.6

[illegible] *Settings the Baseline: FS in Centralised Settings*

The following results regard the first stage of the validation, i.e., the accuracy of a classier trained using only the subset of features identified by the CE algorithm w.r.t the performance obtained by a classifier trained on the entire dataset. For this stage of validation, we train a Neural Network (NN). For MAV the NN is a multi-layer perceptron with two hidden layers of 300 and 100 neurons each. For WESAD, we used a deep NN with four hidden layers of 300,100,64,32 neurons each. The input layer's size depends on the number of features selected, while the size output layer is 37 and 5 for MAV and WESAD, respectively. The activation function is "ReLU"[8] and the optimizer is "Adam"[9] for both the models. These are very common settings which typically provides good performance [41].

Results in Table III show that CE algorithm executed on both datasets in centralised settings can autonomously identify a minimal set of features (i.e., 18 for MAV and 4 for WESAD) with the very same informative content of the whole feature set. The accuracy obtained by both the NN models trained on the CE's FS is statistically equivalent to the one obtained on the whole set of features, inducing a quite impressive compression rate (C): up to 99% and 50% of network traffic for MAV and WESAD, respectively. As a further confirmation of the CE results, we perform the FS using other three reference benchmarks, i.e., MRMR, ANOVA, HJMI. Note that all these approaches select a subset of features with the very same informative content of

[8]REctified Linear Unit
[9]Stochastic Gradient Descent with ADAptive Momentum

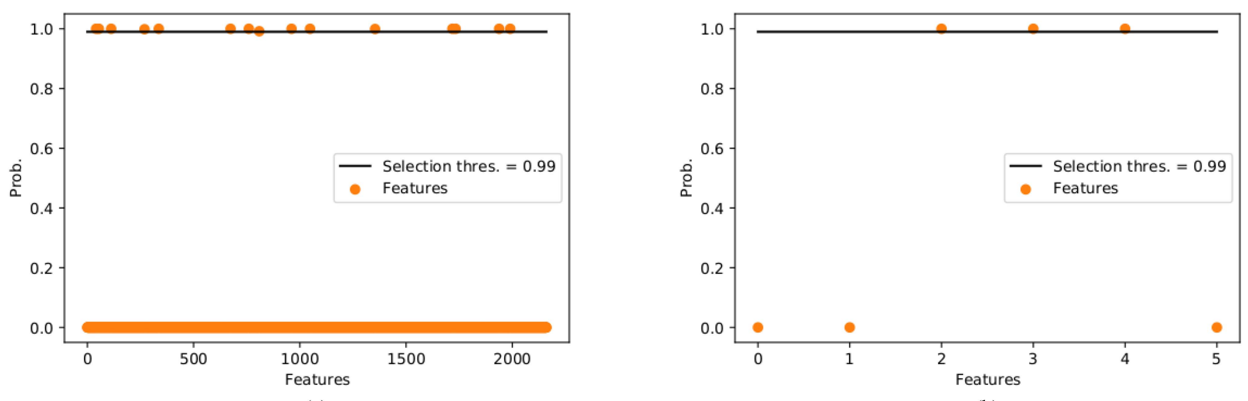


Fig. 4. Centralised FS probability for HOG and IMU. The selected features are those with probability greater than 0.99 (above threshold). (a) C-HOG. (b) C-IMU.

TABLE IV
COMPARISON BETWEEN CE-CFS AND FFS ON MAV AND WESAD

Dataset	Method	Size (# obs.)	FS (#)	C (%)	Accuracy (%±C.I.)
MAV	CE-CFS	2911	18	99	96.7±0.5
	FFS	291	24	99	96.7±0.4
WESAD	CE-CFS	15M	4	50	94.6±0.8
	FFS	3M	4	50	94.6±0.8

CE. However, we point out that for all of them we have to decide beforehand the number of features to be selected. This represent a major shortcoming that, instead, CE-based methods overcome by design, since the number of features to be selected is a byproduct of the CE algorithm. Finally, these results assess the suitability of the CE algorithm on both datasets, thus we can use them as a benchmark for the evaluation of our distributed FFS method.

C. Evaluation of Federated Feature Selection

We focus now on the analysis of our FFS method. We compare its performance to those obtained by CE executed in centralised settings (CE-CFS). We recall that, in federated (distributed) settings, each AV can process only the data it locally collects.

First we assess the performance of FFS in a static distributed scenario where the AVs have collected all the data and, before sending them to the ES, they perform the distributed FS in order to transmit only the very necessary information.

Table IV reveals that for MAV dataset, FFS finds a set of features that, although slightly larger than that found by CE-CFS (24 instead of 18), it has the very same informative content, i.e., the accuracy of the NN model trained on both subsets of features are statistically equivalent. As we can see, the results also hold for the WESAD dataset. Precisely, FFS selects the same number of features identified by CE-CFS. Specifically, FFS and CE-CFS select the same set, i.e., the features with indexes [1,2,5,6], explaining why the NN achieves the same prediction accuracy. We motivate such an exact correspondence between FFS and CE-CFS selection considering that the small size of the complete feature set of WESAD might prevent a high number of feature subset with equivalent informative content. Such an assumption also holds for the MAV dataset. In fact, as we can see in Figs. 4(b), Fig. 5(a) FFS and CE-CFS select the same subset of IMU features. Conversely, when the set of features is more redundant, as it is for the HOGs, there might exist several subsets holding the same informative content. The comparison in Figs. 5(a) and 4(a) confirms such a claim because the two feature sets are overlapping but not equal, yet the overall accuracy is comparable.

This result provides a preliminary insight regarding the effectiveness of the Bayesian aggregation used to merge the information extracted by the AVs from their local datasets. Precisely, Fig. 6 shows the number of selected features at each communication round for the MAV case. As we can see, in the beginning, the cardinality of FS remains almost constant. In this phase, due to the partitioning of data in separated datasets, the CE algorithm has not yet enough knowledge to identify the most informative features. However, the number of features added to the selection starts increasing following an almost-linear trend in a few communication rounds (16). The process ends after 44 rounds, i.e. when the distribution of probabilities indicating the most informative features becomes stable.

Our method's capability to converge quickly to the final and most informative set of features directly affects the amount of network traffic generated upon the completion of the FFS. To confirm such a claim, we performed a set of simulation in which we run FFS varying the size of the local dataset available at the edge device. In this way, we want to analyse our method's robustness when each edge device can access only a limited amount of data. In Table V we report the size of data used for each update (Size), the number of selected features (FS), the accuracy, the number of communication rounds upon convergence (R_c), the compression obtainable with FFS (C), the network overhead generated by FFS (N_{OH}), and the size of the cache needed to collect the data before starting the data transmission. Overall, we observe that, for both datasets, decreasing the size of data processed at each round does not affect significantly the number of communication rounds needed by FFS to converge to a solution, which results in limiting the network overhead generated during the process. Specifically, considering a

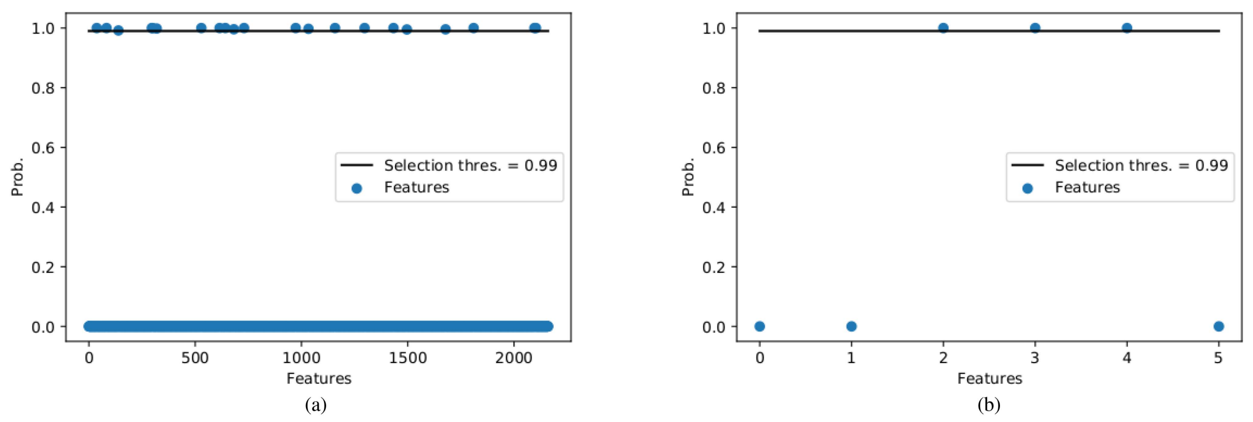


Fig. 5. FFS probability for HOG and IMU. The selected features are those with probability greater than 0.99 (above threshold). (a) F-HOG. (b) F-IMU.

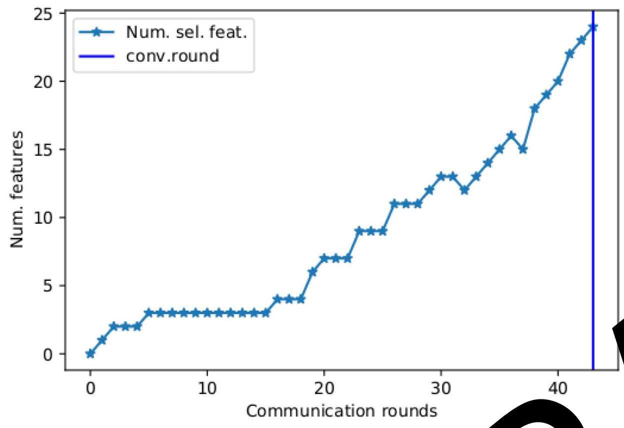


Fig. 6. FFS process on MAV dataset.

TABLE V
PERFORMANCE OF FFS VARYING THE DATA PROCESSED DURING A COMMUNICATION ROUND

Dataset	Size (# obs)	FS (#)	Accuracy (%±C.I.)	R_c (#)	C (%)	N_{OH} (MB)	Cache (MB)
	291	24	96.7±0.4	44	99	16	217
	203	26	96.5±0.3	37	99	13	128
MAV	145	34	97.0±0.3	55	98	20	134
	87	59	97.2±0.3	43	97	15	63
	29	41	97.2±0.4	53	98	19	26
WESAD	$3{\cdot}10^6$	4	93.6±0.8	11	50	0.009	$2{\cdot}10^3$
	$1{\cdot}10^6$	5	93.8±0.5	10	38	0.008	$1{\cdot}10^3$

TABLE VI
PERFORMANCE OF FFS VARYING THE PERCENTAGE NON-FAULTY AVS PER COMMUNICATION ROUND

Dataset	ρ	FS (#)	Accuracy (%±C.I.)	R_c (#)	C (%)
MAV	0.2	48	97.0±0.4	37	98
	0.3	80	97.2±0.3	35	96
WESAD	0.2	4	94.5±0.4	9	50
	0.3	3	67.4±0.4	7	63

TABLE VII
LOCAL FS FROM COMPETITORS APPROACHES

Dataset	Method	Local FS intersection (%)
MAV (k=18)	MRMR	0
	ANOVA	0
	HJMI	0
WESAD (k=4)	MRMR	100
	ANOVA	100
	HJMI	100

dynamic data collection process as in the MAV-related use case, we see that the network overhead is always i) less than the storage needed to cache the data before starting the transmission and ii) negligible considering the compression achieved (i.e., up to 99%). Interestingly, the same holds also for the WESAD scenario. In this case, the network overhead can be considered negligible w.r.t. the size of the data processed ($<$ 1MB) if compared with the compression rate achieved by FFS (up to 50%).

In Table VII we show how the benchmark methods MRMR, ANOVA and HJMI behave when run in isolation on local dataset. Each method has been configured to select the optimal number of features found in a centralised setting. This is clearly an unrealistic situation that we use to demonstrate the limitations coming from running a non-FFS algorithm in federated settings (i.e., on partial datasets). Precisely, taking into account the MAV dataset, all the algorithms run in isolation on each AV, find a different subset of features (i.e., null pairwise intersection). No agreement between AVs on the subset of features means that all the local data must be transmitted to the ES, causing a non negligible waste of network resources. We motivate this behaviour with the fact that the original subset of features is redundant, as in MAV, running the FS in isolation on portions of data is not a winning strategy. Conversely, when the original subset of features is less noisy, as in WESAD, it is more likely that all the AVs find, completely by chance, the same subset of features, i.e., without a way to coordinate the features selection in a consistent and provable way, there are no guarantees for the AVs to identify a consistent and shared subset of features.

Finally we analyse the FFS performance in presence of *faulty nodes*, i.e., a node experiencing issues in transmitting successfully its updates to the ES. Note that, the causes preventing the updates' transmission might relate to either communication-related (i.e, a noisy channel) or the presence of a power-saving policy regulating the duty cycle of AVs switching off the network interface for a time corresponding to a communication round. The aim is assessing the robustness of FFS when few nodes cannot contribute to the distributed learning at each communication round. To this end, we simulate a scenario where, at each communication round, a random number of AVs fail to communicate their updates to the ES. We model the fault of a AV performing a random draw from a Bernoulli distributed random variable, with parameter ρ. At the beginning of the simulation we set ρ and, for each communication round and for each node, we perform a random draw, where 0 means faulty and 1 means non-faulty. This means that the updates of a faulty AV are not considered for the execution of Algorithm 2. We consider a fault rate ρ equal to 0.2 and 0.3, meaning that at each round there are, on average, 2 and 3 faulty AVs out of 10, respectively. Such values can be reasonably assumed as upper bounds to evaluate the performance of the system. Higher rates would reveal that the scenario is not reasonably set up to run with any sort of reliability.

In Table VI we report the performance of FFS, for both datasets. For the MAV dataset, although FFS selects $2\times$ and $3.3\times$ more features than the case when all the AVs contribute to the process (see Table IV), the compression rate deteriorates by 1% and 3%, respectively. The quality of the selection is confirmed by the accuracy that is statistically equivalent to the case without faulty AVs. Regarding WESAD, we note that for $\rho = 0.2$ FFS performance is equivalent to the case with all non-faulty AVs. Conversely, for $\rho = 0.3$ FFS selects a smaller (i.e., 3 instead of 4) and less informative subset of features, as confirmed by the accuracy degradation. The reason is that the information collected by ES at each round is not enough to select, globally, the most informative features. A final comment is about the network overhead, which can be further reduced, limiting the number of *contributing* AVs during a communication round. In fact, Table VI suggests that there is a trade-off between accuracy, compression rate, and number of *contributing* AVs through which we might optimise both the compression and the resources spent to find it. Moreover, there is a limit below which saving resources becomes detrimental to the learning process. However, understanding the nature of such a trade-off is left to future works.

VII. Conclusion and Future Directions

The increasing development of ADSs can leverage AI to abstract both services and applications from the details of fast-flowing low-level data, such as sensor feeds. According to the Edge computing paradigm, a cyber physical system, namely AV, is deputed in collecting data from sensors and perform a lightweight round of computation, by extracting features from raw data and selecting those that maximise the knowledge on the learning task. Since the data gathering process is performed locally by each AV, the selected features might represent a partial subset of those that characterize the phenomenon and might be inconsistent to learn the model of the underlying process. We tackle this problem, by proposing a novel Federated Feature Selection (FFS) algorithm, exploiting a distributed computing paradigm applied to AVs. In FFS, AVs collaborate to iteratively come up with the minimal set of features selected from their local datasets, to be used as a compression schema for transmitting their data to the Edge Server. Feature selection is done by leveraging on the Mutual Information metric and the solution of the optimization problem is achieved through Cross-entropy method. The aggregation algorithm of the FFS solution is based on a Bayesian approach through which we merge the control information sent by the AVs to the ES. To test the proposed FFS algorithm we presented two different learning tasks, by using real-world datasets: MAV and WESAD. The former was suitable to test FFS with images and inertial measurements, which characterize the position of an AV in the environment. The latter was suitable to characterize time series produced by human state monitoring systems, like ECG, EDA, EMG, etc. The results show that our FFS algorithm identifies a minimal subset of informative features without sharing any raw data between AVs in the process. FFS is robust to feature redundancy, i.e., in presence of high rates of redundant features, all the AVs can reach a consensus on the FS achieving a compression rate up to 90x on the selected datasets. Finally, the quality of the feature selection is maintained, i.e., a learning model trained on the selected features is as accurate as a model trained on the whole feature set. Concluding, the proposed framework is general and modular, i.e., it can be applied to every incremental FS algorithm that associates a probability to each feature. We plan to investigate how to turn it into a framework to include more FS algorithms. Moreover, our solution is built on few simplifying assumptions: local datasets are *iid* and data are labelled. Therefore, for the future we plan to extend it to include non-*iid* data in possibly unsupervised or semi-supervised scenarios.

Appendix

A. Proof of Convergence of the Federated Method

In this section, we analyze the probability that the distribution $\mathbf{p}$ converges toward the optimal solution $\mathbf{p}^*$, when the Algorithm 1 is applied in a centralized way. Then, we extend this result for the proposed federated algorithm.

The convergence analysis is based on the results in [38], [39]: following that notation, we introduce some preliminary definitions. In the CE, the candidate solutions $\mathbf{Z}_t = \{\mathbf{z}_1 \cdots \mathbf{z}_S\}$ generated at iteration t are *iid* with distribution $g(\mathbf{z}, \mathbf{p}_{t-1})$.

We define $\mathcal{Z}_t := \{\mathbf{z}_{j,\tau} \neq \mathbf{z}^*\ j = 1 \cdots S,\ \tau = 1 \cdots t\} \subseteq \mathbf{Z}_t$ as the subset of $\mathbf{Z}_t$ of the samples generated up to t that do not provide the optimal solution $\mathbf{z}^*$. The probability $\Pr(\mathcal{Z}_t)$ that the optimal solution is not available until t can be found as in the following:

$$\begin{aligned}\Pr(\mathcal{Z}_t) &= \Pr(\mathcal{Z}_1)\prod_{\tau=2}^{t}\Pr(\mathcal{Z}_\tau|\mathcal{Z}_{\tau-1})\\ &= \Pr(\mathcal{Z}_1)\prod_{\tau=2}^{t}\left(\Pr(\mathbf{z}_\tau \neq \mathbf{z}^*|\mathcal{Z}_{\tau-1})\right)^S \end{aligned}\tag{14}$$

The (14) comes from the statistical independence of S identically distributed samples generated by the algorithm at iteration t. The upper bound for the probability $\Pr(\mathbf{z}_\tau \neq \mathbf{z}^*|\mathcal{Z}_{\tau-1})$ that the optimal solution was unavailable until τ is derived in [38], [39] as:

$$\Pr(\mathbf{z}_\tau \neq \mathbf{z}^*|\mathcal{Z}_{\tau-1}) \leq 1 - \Pr(\mathbf{z}_1 = \mathbf{z}^*)\prod_{i=1}^{\tau-1}(1-\alpha_i)^m \tag{15}$$

where

$$\begin{aligned}\Pr(\mathbf{z}_1 = \mathbf{z}^*) = \prod_{i=1}^{m}\big(&p_i(z_i=0)\mathcal{I}(z_i^*=1)\\ &+ (1-p_i(z_i=0))\,\mathcal{I}(z_i^*=0)\big)\end{aligned}\tag{16}$$

Note that due to the definition of $\mathcal{Z}_1$ its probability is $\Pr(\mathcal{Z}_1) = 1 - \Pr(\mathbf{z}_1 = \mathbf{z}^*)$.

Combining equations (15) and (16), (14) becomes:

$$\begin{aligned}\Pr(\mathbf{z}_t \neq \mathbf{z}^*) \leq &\left(1 - \prod_{i=1}^{m}\big(p_i(z_i=0)\mathcal{I}(z_i^*=1)\right.\\ &\left. + (1-p_i(z_i=0))\,\mathcal{I}(z_i^*=0)\big)\right)\prod_{\tau=2}^{t}\\ &\times\left(1 - \prod_{i=1}^{m}\big(p_i(z_i=0)\mathcal{I}(z_i^*=1)(1-p_i\right.\\ &\left. + (z_i=0))\mathcal{I}(z_i^*=0)\big)\prod_{j=1}^{\tau-1}(1-\alpha_j)^m\right)^S\end{aligned}\tag{17}$$

The right side of the (17) is close to 0 for $t \to \infty$, if $\sum_{\tau=1}^{\infty}\prod_{j=1}^{\tau-1}(1-\alpha_j)^m \to \infty$, i.e., the sequence of the parameters α_j are generated by the function $\frac{1}{j\cdot m}$, as proven by authors in [42] (section 3.7). Note that, (17) can be used to determine numerically a combination of parameter values that yields a desired minimum probability of generating the optimal solution within a time t.

Therefore, Algorithm (1) definitely provides the optimal solution when applied in a centralized way. We extend this result for the federated approach as follows. The AVs draw distinct samples $\mathbf{z}_1 \cdots \mathbf{z}_S$ independently from an identical distribution, as stated in the section V. This means that the node l finds an optimal solution for its $\mathbf{z}_1 \cdots \mathbf{z}_S$ that differs for that obtained by the centralized algorithm. Hence, combining the local distributions into the global one as in (9), we need to prove that the local node can receive from the server a federated solution that is close to the solution provided by the centralized scenario, for $t \to \infty$.

Defining the Hamming's distance $\mathcal{L}(\mathbf{z}^*, \mathbf{z}_\tau)$ between the sample $\mathbf{z}_\tau$ at the time τ and the optimal solution $\mathbf{z}^*$, the set $\tilde{\mathcal{Z}}_t := \{\mathbf{z}_{i,\tau} \mid \mathcal{L}(\mathbf{z}^*, \mathbf{z}_{i,\tau}) = m_l\}$ contains the samples generated up to time t that differs for m_l entries from the optimal solution $\mathbf{z}^*$.

As in (14), we can calculate the probability $\Pr(\tilde{\mathcal{Z}}_t)$ as follows:

$$\Pr(\tilde{\mathcal{Z}}_t) = \Pr(\tilde{\mathcal{Z}}_1)\prod_{\tau=2}^{t}\Pr(\tilde{\mathcal{Z}}_\tau|\tilde{\mathcal{Z}}_{\tau-1}) \tag{18}$$

Exploiting again the results in [38], [39], and the statistical independence of the S identically distributed samples generated by the algorithm at a given iteration, the following equation holds for the conditional probability for the given node l:

$$\begin{aligned}\Pr(\tilde{\mathcal{Z}}_\tau|\tilde{\mathcal{Z}}_{\tau-1}) = \Bigg[&\binom{m}{m_l}\Pr(\mathbf{z}_1 = \mathbf{z}_1^*)\cdot\prod_{i=1}^{\tau-1}(1-\alpha_{i,l})^{m-m_l}\cdot\\ &\times\left(1 - \Pr(\mathbf{z}_1 = \mathbf{z}_1^*)\prod_{i=1}^{\tau-1}(1-\alpha_{i,l})^{m_l}\right)\Bigg]^S\end{aligned}\tag{19}$$

Note that, the result provided in (18) refers to the l-th node. Hence, the global solution is obtained as the weighted average over all the local probabilities $\Pr(\tilde{\mathcal{Z}}_{l,t})$ as:

$$\Pr_{\mathbf{G}}(\tilde{\mathcal{Z}}_t) = \sum_{l=1}^{L}\Pr(\tilde{\mathcal{Z}}_{l,t})\omega_l \tag{20}$$

where ω_l are computed as in (10).

The probability in (20) is close to 0, for $t \to \infty$, if $\sum_{\tau=1}^{\infty}\prod_{i=1}^{\tau-1}(1-\alpha_{i,l})^{m_l} \to \infty \;\forall\; l = 1,\ldots,L$. Note that, if the sum of products of $(1-\alpha_{i,l})^{m_l}$ is close to ∞ also the sum of products of $(1-\alpha_{i,l})^{m-m_l}$ is close to ∞. The sequences of the $\alpha_{i,l}$ parameters guarantee the convergence also in this case. Indeed, the parameters are generated locally by the node, using the function $\frac{1}{m\cdot t}$.

B. Analysis of the Global Probability Computational Effort

In this section, we analyze the probability distribution of the number of iterations t needed to evaluate the global probability in (9). We address this issue by exploiting the result in (20), which describes the probability that the global solution obtained at the iteration t differs by m_l entries from the optimal one. Hence, the probability that the global solution is reached within t can be written as follows:

$$\Pr_{\mathbf{G}}(\mathbf{z}_t = \mathbf{z}^*) = 1 - \sum_{m_l=1}^{m}\Pr_{\mathbf{G}}(\tilde{\mathcal{Z}}_t) \tag{21}$$

We can exploit the following inequality $(1-\alpha)^m \leq e^{-\alpha m} \mid 0 \leq \alpha \leq 1,\; m \geq 0$ to find the upper bound shown in the following:

$$\Pr_{\mathbf{G}}(\mathbf{z}_t = \mathbf{z}^*) \leq 1 - \sum_{m_l=1}^{m} \sum_{l=1}^{L} \Pr(\tilde{\mathcal{Z}}_1) \prod_{\tau=2}^{t} \left[\binom{m}{m_l} \Pr(z_1 = z_1^*) \times \left(exp\left(-\sum_{i=1}^{\tau-1} \alpha_{i,l}(m - m_l) \right) - \Pr \times \ (z_1 = z_1^*)\, exp\left(-\sum_{i=1}^{\tau-1} \alpha_{i,l}\, m \right) \right) \right]^{S} \omega(l) \tag{22}$$

The difference between exponentials in (22) goes to zero faster than the binomial coefficient goes to infinity, as m increases, if the coefficient α satisfies the conditions verified in the previous appendix. Thus (22) can be used to evaluate the probability distribution of the number of iterations $t = 1, 2 \ldots, \infty$ required to converge to the optimal global solution. The numerical analysis shows an average value of 13 iterations to converge by using the parameters presented in the section VI, which is affordable for many edge devices like Nvidia Jetson Nano or RaspberryPi.

REFERENCES

[1] A. Bondavalli, S. Bouchenak, and H. Kopetz, *Cyber-Physical Systems of Systems: Foundations—A Conceptual Model and Some Derivations: The AMADEOS Legacy*, vol. 10099. Berlin, Germany: Springer, 2016.

[2] D. Bacciu *et al.*, "Teaching-trustworthy autonomous cyber-physical applications through human-centred intelligence," in *Proc. IEEE Int. Conf. Omni-Layer Intell. Syst.*, 2021, pp. 1–6.

[3] J. E. Siegel, D. C. Erb, and S. E. Sarma, "A survey of the connected vehicle landscape—architectures, enabling technologies, applications, and development areas," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2391–2406, Aug. 2018.

[4] L. Qingqing, J. P. Queralta, T. N. Gia, H. Tenhunen, Z. Zou, and T. Westerlund, "Visual odometry offloading in internet of vehicles with compression at the edge of the network," in *Proc. 12th Int. Conf. Mobile Comput. Ubiquitous Netw.*, 2019, pp. 1–2.

[5] X. Xie and K. H. Kim, "Source compression with bounded DNN perception loss for IoT edge computer vision," in *Proc. Int. Conf. ACM MobiCom*, 2019, pp. 1–16.

[6] Y. Saeys, I. Inza, and I. Larranaga, "A review of feature selection techniques in bioinformatics," *J. Bioinf. Rev.*, vol. 23, no. 19, pp. 2507–2517, Aug. 2007.

[7] S. Egea, A. R. Manez, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Intelligent IoT traffic classification using novel search strategy for fast-based-correlation feature selection in industrial environments," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1616–1624, Jun. 2018.

[8] G. Brown, "A new perspective for information theoretic feature selection," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2009, pp. 49–56.

[9] X. Nguyen, J. Chan, S. Romano, and J. Bailey, "Effective global approaches for mutual information based feature selection," in *Proc. Int. Conf. ACM Knowl. Discov. Data Mining*, 2014, pp. 1–10.

[10] D. Rubinstein and R. Y. Kroese, *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Berlin, Germany: Springer, 2004.

[11] A. Jović, K. Brkić, and N. Bogunović, "A review of feature selection methods with applications," in *Proc. Int. Conf. IEEE MIPRO*, 2015, pp. 1200–1205.

[12] B. Venkatesh and J. Anuradha, "A review of feature selection and its methods," *J. Cybern. Inf. Technol.*, vol. 19, no. 1, pp. 3–26, Feb. 2019.

[13] F. Chandrashekar and G. Sahin, "A survey on feature selection methods," *Comput. Elect. Eng.*, vol. 40, no. 1, pp. 16–28, 2014.

[14] L. Miao and J. Niu, "A survey on feature selection," in *Proc. Int. Conf. Elsevier Inf. Technol., Quantitative Manage.*, 2016, pp. 919–926.

[15] C. Ding and H. Peng, "Minimum redundancy feature selection from microarray gene expression data," in *Proc. IEEE Bioinf. Conf. Comput. Syst. Bioinf.*, 2003, pp. 523–528.

[16] A. Gocht, C. Lehmann, and R. Schöne, "A new approach for automated feature selection," in *Proc. IEEE Int. Conf. Big Data*, 2018, pp. 4915–4920.

[17] R. Petroccia, P. Cassará, and K. Pelekanakis, "Optimizing adaptive communications in underwater acoustic networks," in *Proc. Int. Conf. IEEE/MTS OCEANS*, 2019, pp. 1–7.

[18] G. F. Anastasi, P. Cassará, P. Dazzi, A. Gotta, M. Mordacchini, and A. Passarella, "A hybrid cross-entropy cognitive-based algorithm for resource allocation in cloud environments," in *Proc. Int. Conf. IEEE Self-Adaptive Self-Organizing Syst.*, 2014, pp. 11–20.

[19] F. Guarino, P. Cassará, S. Longo, M. Cellura, and E. Ferro, "Load match optimisation of a residential building case study: A cross-entropy based electricity storage sizing algorithm," *Elsevier J. Appl. Energy*, vol. 154, pp. 380–391, 2015.

[20] R. M. Heiberger and E. Neuwirth, "One-way anova," in *R Through Excel*. Berlin, Germany: Springer, 2009, pp. 165–191.

[21] J. Konečý, B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," 2015, pp. 1–5, *arXiv1511.03575*.

[22] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. AISTATS*, 2017, pp. 248–257.

[23] S. Wang *et al.*, "When edge meets learning: Adaptive control for resource-constrained distributed machine learning," in *Proc. Int. Conf. IEEE INFOCOM*, 2018, pp. 63–71.

[24] M. M. Amiri and D. Gunduz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," in *Proc. Int. Conf. IEEE ISIT*, 2019, pp. 1–5.

[25] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for on-device federated learning," 2019, pp. 1–30, *arXiv:1910.06378*.

[26] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1–11.

[27] Y. Mao, "Learning from differentially private neural activations with edge computing," in *Proc. Int. Conf. IEEE/ACM SEC*, 2018, pp. 90–102.

[28] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G Srivastava, "A survey on security and privacy of federated learning," *Elsevier J. Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, 2021.

[29] L. Valerio, A. Passarella, and M. Conti, "Hypothesis transfer learning for efficient data computing in smart cities environments," in *Proc. Int. Conf. SMARTCOMP*, 2016, pp. 1–8.

[30] L. Valerio, A. Passarella, and M. Conti, "A communication efficient distributed learning framework for smart environments," *Elsevier J. Pervasive Mobile Comput.*, vol. 41, pp. 46–68, 2017.

[31] X. Ye, H. Li, A. Imakura, and T. Sakurai, "Distributed collaborative feature selection based on intermediate representation," in *Proc. Int. Conf. IJCAI*, 2019, pp. 4242–4149.

[32] S. Banerjee, E. Elmroth, and M. Bhuyan, "Fed-FiS: A novel information-theoretic federated feature selection for learning stability," in *Neural Information Processing*, T. Mantoro, M. Lee, M. A. Ayu, K. W. Wong, and A. N. Hidayanto, Eds., Berlin, Germany: Springer, 2021, pp. 480–487.

[33] P. Schmidt, A. Reiss, R. Duerichen, C. Marberger, and K. Van Laerhoven, "Introducing WESAD, a multimodal dataset for wearable stress and affect detection," in *Proc. Int. Conf. ACM ICMI*, 2019, pp. 1–9.

[34] R. McEliece, *The Theory of Information and Coding: A. Mathematical Framework for Communication* (Encyclopedia of Mathematics and Its Applications 3). Reading, MA, USA: Addison-Wesley, 1977.

[35] J. Cover and T. M. Thomas, *Elements of Information Theory*. New York, NY, USA: Wiley, 1991.

[36] A. Chaovalitwongse, I. Androulakis, and P. Pardalos, "Quadratic integer programming: Complexity and equivalent form," in *Encyclopedia of Optimization*. Berlin, Germany: Springer, 2009, pp. 3153–3159.

[37] G. Brown, A. Pocock, M. Zhao, and M. Zheng, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *Springer J. Mach. Learn. Res.*, vol. 13, pp. 27–66, 2012.

[38] A. Costa, O. D. Jones, and D. Kroese, "Convergence properties of the cross-entropy method for discrete optimization," *Elsevier Oper. Res. Lett.*, vol. 35, no. 7, pp. 573–580, Sep. 2007.

[39] Z. Wu and M. Kolonko, "Asymptotic properties of a generalized cross-entropy optimization algorithm," *IEEE Trans. Evol. Comput.*, vol. 18, no. 5, pp. 658–673, Oct. 2014.

[40] P. Chen, C. Huang, C. Lien, and Y. Tsai, "An efficient hardware implementation of HOG feature extraction for human detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 656–662, Apr. 2014.

[41] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[42] K. Knopp, *Infinite Sequences and Series*. New York, NY, USA: Dover, 1956.

Pietro Cassará received the M.Sc. degrees in telecommunication and electronic engineering from University of Palermo, in 2005, and the Ph.D. degree from the State University of New York, in 2010. He is currently a Staff Member with the Institute of Science and Information Technologies (ISTI), National Research Council (CNR), Pisa, Italy, and since 2017, he has been a Temporary Staff Member with the CMRE lab, NATO of La Spezia. He has been participating in European, ESA, and National funded projects. His research interests include wireless sensor network and machine learning for communication and networking.

He is currently a member of the IEEE ComSoc and VTS, vice-chair of the Italian Council of IPV6, and member of IETF for the ISG IPE. He serves in the TPCs of flagship ComSoc and VTS conferences, he also contribute as a Reviewer and into the editorial board of IEEE and MDPI journals.

Alberto Gotta (Member, IEEE) received the M.Sc. and Ph.D. degrees, in 2002 and 2007, respectively. He is currently a Researcher with the Institute of Information Science and Technologies (ISTI), National Research Council (CNR), Italy. He has been Principal investigator of several EU, national, regional, and ESA funded R&D projects. He coauthored more than 100 papers among which the most cited article in 2012 and 2013 on Elsevier Computer Communications and the second most cited article from 2019 in Elsevier Array. His research interests include traffic engineering, satellite and nonterrestrial networks, aerial networks, IoT, sensor networks, and machine learning for communications. He serves the TPCs of flagship ComSoc conferences and symposia and also the editorial board of MDPI Sensors, MDPI Network, the *International Journal of Informatics and Communication Technology* (IJ-ICT), the *International Journal of Power Electronics and Drive Systems* (IJPEDS), and the *Journal of Computer Networks and Communications* (CNC).

Lorenzo Valerio is currently a Researcher with IIT-CNR. His research activity focuses on the design of communication efficient distributed learning solutions, machine learning solutions for resource-constrained devices and machine learning based cellular traffic offloading solutions for the Future Internet. He has authored or coauthored in journals and conference proceedings more than 40+ papers. He has served as Workshop co-chair for IEEE AOC'15 and IEEE PerConAI'22. He has also been Guest Editor for the *Elsevier Computer Communications* and *Elsevier Pervasive and Mobile Computing*. He has been recipient for one Best Paper Award at IEEE WoWMoM 2013 and one Best Paper Nomination at IEEE SMARTCOMP 2016. He is currently in the editorial board of *Elsevier Computer Communications*.

Open Access provided by 'Consiglio Nazionale delle Ricerche' within the CRUI CARE Agreement

E Nonlinear Gradient Mappings and Stochastic Optimization: A General Framework with Applications to Heavy-Tail Noise

The appended paper follows.

NONLINEAR GRADIENT MAPPINGS AND STOCHASTIC OPTIMIZATION: A GENERAL FRAMEWORK WITH APPLICATIONS TO HEAVY-TAIL NOISE

DUŠAN JAKOVETIĆ*, DRAGANA BAJOVIĆ†, ANIT KUMAR SAHU‡, SOUMMYA KAR§, NEMANJA MILOŠEVIĆ*, AND DUŠAN STAMENKOVIĆ*

Abstract. We introduce a general framework for nonlinear stochastic gradient descent (SGD) for the scenarios when gradient noise exhibits heavy tails. The proposed framework subsumes several popular nonlinearity choices, like clipped, normalized, signed or quantized gradient, but we also consider novel nonlinearity choices. We establish for the considered class of methods strong convergence guarantees assuming a strongly convex cost function with Lipschitz continuous gradients under very general assumptions on the gradient noise. Most notably, we show that, for a nonlinearity with bounded outputs and for the gradient noise that may not have finite moments of order greater than one, the nonlinear SGD's mean squared error (MSE), or equivalently, the expected cost function's optimality gap, converges to zero at rate $O(1/t^\zeta)$, $\zeta \in (0,1)$. In contrast, for the same noise setting, the linear SGD generates a sequence with unbounded variances. Furthermore, for general nonlinearities that can be decoupled component wise and a class of joint nonlinearities, we show that the nonlinear SGD asymptotically (locally) achieves a $O(1/t)$ rate in the weak convergence sense and explicitly quantify the corresponding asymptotic variance. Experiments show that, while our framework is more general than existing studies of SGD under heavy-tail noise, several easy-to-implement nonlinearities from our framework are competitive with state-of-the-art alternatives on real data sets with heavy tail noises.

Key words. Stochastic optimization; stochastic gradient descent; nonlinear mapping; heavy-tail noise; convergence rate; mean square analysis; asymptotic normality; stochastic approximation.

AMS subject classifications. 90C15, 90C25, 65K05, 62L20, 68T05

1. Introduction.

Stochastic gradient descent (SGD) and its variants, e.g., [27, 16, 23, 35, 25, 12, 24, 7], are popular and standard methods for large scale optimization and training of various machine learning models, e.g., [5, 6, 31, 8]. Recently, there have been several studies that demonstrate that the gradient noise in SGD is heavy-tailed, e.g., when training deep learning models [[illegible]2, 17, 37].

Motivated by these studies, we introduce a general analytical framework for *nonlinear* SGD when the gradient evaluation is subject to a heavy-tailed noise. We combat the gradient noise with a generic nonlinearity that is applied on the noisy gradient to effectively reduce the noise effect. The resulting class of nonlinear methods subsumes several popular choices in training machine learning models, including normalized gradient descent and clipped gradient descent, e.g., [28, 36], the sign gradient, e.g., [4, 2], and (component-wise) quantized gradient, e.g., [1, 18].[1]

We establish for the considered class of methods several results that demonstrate a high degree of robustness to noise under very general assumptions on the nonlinearity

*University of Novi Sad, Faculty of Sciences, Department of Mathematics and Informatics (dusan.jakovetic@dmi.uns.ac.rs, nmilosev@dmi.uns.ac.rs, dusan.stamenkovic@dmi.uns.ac.rs)

†University of Novi Sad, Faculty of Technical Sciences, Department of Power, Electronic and Communication Engineering, (dbajovic@uns.ac.rs)

‡Amazon Alexa AI (anit.sahu@gmail.com)

§Department of Electrical and Computer Engineering, Carnegie Mellon University (soummyak@andrew.cmu.edu). The work of D. Bajovic and D. Jakovetic is partially supported by the European Union's Horizon 2020 Research and Innovation program under grant agreement No 957337. The paper reflects only the view of the authors and the Commission is not responsible for any use that may be made of the information it contains.

[1]Interestingly, some of these nonlinear methods are usually introduced with a different motivation than robustness, like, e.g., speeding up training, see, e.g., [36], or communication efficiency, [2, 4].

and on the gradient noise, assuming a strongly convex cost with Lipschitz continuous gradient. First, for a nonlinearity with bounded outputs (e.g., a sign, normalized, or clipped gradient) and the gradient noise that may have infinite moments of order greater than one, assuming that the noise probability density function (pdf) is symmetric, we show that the nonlinear SGD converges almost surely to the solution, and, moreover, achieves a global $O(1/t^{\zeta})$ mean squared error (MSE) convergence rate, where we explicitly quantify the degree $\zeta \in (0,1)$. In the same setting, the linear SGD generates a sequence with unbounded variances at each iteration t. Furthermore, assuming the gradient noise with finite variance, we show – for the unbounded nonlinearities that are lower bounded by a linear function – almost sure convergence and the $O(1/t)$ global MSE rate.

Next, for the general nonlinearities with bounded outputs that can be decoupled component-wise and a restricted class of joint nonlinearities with bounded outputs, we show under the heavy-tail noise a local (asymptotic) $O(1/t)$ rate in the weak convergence sense. More precisely, we show that the sequence generated by the nonlinear SGD is asymptotically normal and explicitly quantify the asymptotic variance. Finally, we illustrate the results on several examples of the nonlinearity and the gradient noise pdf, highlighting and quantifying the noise regimes and the corresponding gains of the nonlinear SGD over the linear SGD scheme. In more detail, the asymptotic variance expression reveals an interesting tradeoff that the nonlinearity makes on the algorithm performance: on the one hand, the nonlinearity suppresses the noise effect to a certain degree, but on the other hand it also reduces the "useful information flow" and hence slows down convergence with respect to the noiseless case. We explicitly quantify this tradeoff and demonstrate through examples that an appropriately chosen nonlinearity strictly improves performance over the linear scheme in a high noise setting. Finally, we carry out numerical experiments on several real data sets that exhibit heavy tail gradient noise effects. The experiments show that, while our analytical framework is more general than usual studies of SGD under heavy-tail noise, several easy-to-implement example nonlinearities of our framework – including those not previously used – are competitive with state-of-the-art alternatives.

Technically, for component-wise nonlinearities and the asymptotic analysis, we develop proofs based on stochastic approximation arguments, e.g., [26], following the noise and nonlinearities assumptions framework similar to [30]. The paper [30] is concerned with a related but different problem than ours: it considers linear estimation of a vector parameter observed through a sequence of scalar observation equations, and it is not concerned with a global MSE rate analysis that we provide here. For the MSE analysis and for the nonlinearities that cannot be expressed component-wise, like the clipped and normalized gradient, we develop novel analysis techniques.

There have been several works that study robustness of stochastic gradient descent under certain variants of heavy-tailed noises. Reference [37] consider an adaptive gradient clipping method and establish convergence rates in expectation for the considered method under a heavy-tailed noise. For this, the authors assume that the expected value of the norm of the gradient noise raised to power α is finite, for $\alpha \in (1,2]$. They also provide lower complexity bounds for SGD methods assuming in addition that the expected α-power of the norm of the *stochastic gradient* is finite. The paper [32] establishes convergence of the *linear* SGD assuming that the gradient noise follows a heavy-tailed α-stable distribution.

It is worth noting that, in addition to the MSE (expected optimality gap) results achieved here, it is also of interest to derive high probability bounds. Specifically, given a target accuracy $\epsilon > 0$ and a confidence level $1-\beta$, $\beta \in (0,1)$, we would like to find $T(\epsilon, \beta)$ such that $f(\mathbf{x}^t) - f(\mathbf{x}^\star) \leq \epsilon$ with probability at least $1-\beta$, for all iterations $t \geq T(\epsilon, \beta)$. Application of the Markov inequality to our result $\mathbb{E}[\, f(\mathbf{x}^t) - f(\mathbf{x}^\star)\,] = O(1/t^\zeta)$ yields, abstracting dependencies on other system parameters, a bound $T(\epsilon,\beta) \sim \frac{1}{(\beta\, \epsilon)^{1/\zeta}}$. This involves a strong dependence on β, on the order $1/\beta^{1/\zeta}$. Several works, e.g., [13, 14, 19, 15, 11], establish high probability bounds where $T(\epsilon, \beta)$ depends *logarithmically* on β for the settings therein. For example, references [13, 14] establish high probability bounds for the stochastic gradient methods therein assuming that the gradient noise has light tails (sub-Gaussian noise). The authors of [19] establish the corresponding bounds for the basic SGD and the mirror descent that utilize a gradient truncation technique. They relax the noise sub-Gaussianity assumption and assume a finite noise variance. Very recently, [15] establishes high probability bounds for accelerated SGD with a clipping nonlinearity, but assuming a finite variance of the gradient noise. Reference [11] proposes a procedure called proxBoost and establishes for the procedure high probability bounds, again assuming a finite noise variance (without the sub-Gaussianity assumption). It is highly relevant to investigate high probability bounds for the problem setting and the algorithmic class considered in this paper. Of special interest is to provide high probability bounds for a broader class of nonlinearities than the usually studied clipping-type nonlinearities; this is an interesting future work direction.

In summary, with respect to existing work, our framework is more general with respect to both the adopted nonlinearity in SGD and the "thickness" of the gradient noise tail, assuming in addition that the noise pdf is a symmetric function. For example, current works usually assume a single choice for the nonlinearity, e.g., gradient clipping, while we consider a general nonlinearity that subsumes many popular choices. Also, provided that the nonlinearity's output is bounded (which is true for many popular choices like the clipped, signed and normalized gradient), we establish a sublinear MSE convergence rate $O(1/t^\zeta)$ assuming only that the expected norm of the gradient noise is finite, an assumption weaker than those considered in the works of [15, 37, 11, 32]. On the other hand, we assume a strongly convex smooth cost function, which is equivalent to or stronger than the assumptions made in these works. See also Examples 3.2 and 3.3. ahead for further rate comparisons with existing work.

The idea of employing a nonlinearity into a "baseline" linear scheme has also been used in other contexts. Most notably, several works consider nonlinear versions of the standard consensus algorithm to evaluate average of scalar values in a distributed fashion, e.g., [22, 33, 10]. The paper [22] introduces a trigonometric nonlinearity into a standard linear consensus dynamics and shows an improved dependence of the method on initial conditions. References [33] and [10] employ a general nonlinearity in the linear consensus dynamics and show that it improves the method's resilience to additive communication noise. The authors of [34] modify the linear consensus by taking out from the averaging operation the maximal and minimal estimates among the estimates from all neighbors of a node. The above works are different from ours as they focus on the specific consensus problem that can be translated into minimizing a convex quadratic cost function in a distributed way over a generic, connected network. In contrast, we consider general strongly convex costs, and we are not directly concerned with distributed systems.

Paper organization. Section 2 describes the problem model and the nonlinear SGD framework that we assume. Section 3 and Section 4 explain our results on nonlinear SGD for component-wise and joint nonlinearities, respectively. Section 5 and Section 6 then provide proofs of the corresponding results. Section 7 illustrates the performance of several example methods from our nonlinear SGD framework on real data sets that have heavy-tail gradient noise. Finally, Section 8 concludes the paper. Some auxiliary results and proofs are delegated to the Appendix.

Notation. We denote by $\mathbb{R}$ and $\mathbb{R}_+$, respectively, the set of real numbers and real nonnegative numbers, and by $\mathbb{R}^m$ the m-dimensional Euclidean real coordinate space. We use normal (lower-case or upper-case) letters for scalars, lower-case boldface letters for vectors, and upper case boldface letters for matrices. Further, we denote by: a_i or $[\mathbf{a}]_i$, as appropriate, the i-th element of vector $\mathbf{a}$; $\mathbf{A}_{ij}$ or $[\mathbf{A}]_{ij}$, as appropriate, the entry in the i-th row and j-th column of a matrix $\mathbf{A}$; $\mathbf{A}^\top$ the transpose of a matrix $\mathbf{A}$; and trace($\mathbf{A}$) the sum of diagonal elements of $\mathbf{A}$. Further, we use either $\mathbf{a}^\top\mathbf{b}$ or $\langle \mathbf{a}, \mathbf{b}\rangle$ for the inner product of vectors $\mathbf{a}$ and $\mathbf{b}$. Next, we let $\mathbf{I}$ and $\mathbf{0}$ be, respectively, the identity matrix and the zero matrix; $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument; $\phi'(w)$ the first derivative evaluated at w of a function $\phi : \mathbb{R} \to \mathbb{R}$; $\nabla h(\mathbf{w})$ and $\nabla^2 h(\mathbf{w})$ the gradient and Hessian, respectively, evaluated at $\mathbf{w}$ of a function $h : \mathbb{R}^m \to \mathbb{R}$; $\mathbb{P}(\mathcal{A})$ and $\mathbb{E}[u]$ the probability of an event $\mathcal{A}$ and expectation of a random variable u, respectively; and by sign(a) the sign function, i.e., sign(a) $= 1$, for $a > 0$, sign(a) $= -1$, for $a < 0$, and sign(0) $= 0$. Finally, for two positive sequences η_n and χ_n, we have: $\eta_n = O(\chi_n)$ if $\limsup_{n\to\infty} \frac{\eta_n}{\chi_n} < \infty$; $\eta_n = \Omega(\chi_n)$ if $\liminf_{n\to\infty} \frac{\eta_n}{\chi_n} > 0$; and $\eta_n = \Theta(\chi_n)$ if $\eta_n = O(\chi_n)$ and $\eta_n = \Omega(\chi_n)$.

2. Problem Model and the nonlinear SGD Framework. We consider the following unconstrained problem:

$$\text{minimize} \quad f(\mathbf{x}), \tag{2.1}$$

where $f : \mathbb{R}^d \mapsto \mathbb{R}$ is a convex function.

We make the following standard assumption.

Assumption 1. *Function $f : \mathbb{R}^d \mapsto \mathbb{R}$ is strongly convex with strong convexity parameter $\mu > 0$, and it has Lipschitz continuous gradient with Lipschitz constant $L \geq \mu$.*

For asymptotic results (see ahead Theorems 3.1 and 3.3), we will also require the following assumption.

Assumption 2. *Function $f : \mathbb{R}^d \mapsto \mathbb{R}$ is twice continuously differentiable.*

Under Assumption 1, problem (2.1) has a unique solution, which we denote by $\mathbf{x}^\star \in \mathbb{R}^d$.

In machine learning settings, f can correspond to the risk function, i.e.,

$$f(\mathbf{x}) = \mathbb{E}_{\mathbf{d}\sim P}\left[\,\ell\,(\mathbf{x};\mathbf{d})\,\right] + \mathcal{R}(\mathbf{x}). \tag{2.2}$$

Here, P is the (unknown) distribution from which the data samples $\mathbf{d} \in \mathbb{R}^q$ are drawn; $\ell(\cdot;\cdot)$ is a loss function that is smooth and convex in its first argument for any fixed value of the second argument; and $\mathcal{R} : \mathbb{R}^d \mapsto \mathbb{R}$ is a smooth strongly convex regularizer. Similarly, f can be empirical risk, i.e., $f(\mathbf{x}) = \frac{1}{n}\left(\sum_{j=1}^n \ell\,(\mathbf{x};\mathbf{d}_j)\right) + \mathcal{R}(\mathbf{x})$, where $\mathbf{d}_j$, $j = 1, ..., n$, is the set of training data points. Several machine learning models fall within the described framework under Assumptions 1–2, including, e.g., ℓ_2-regularized quadratic and logistic losses.

We introduce a general framework for *nonlinear* SGD methods to solve problem (1); an algorithm within the framework takes the following form:

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \alpha_t \boldsymbol{\Psi}(\nabla f(\mathbf{x}^t) + \boldsymbol{\nu}^t). \tag{2.3}$$

Here, $\mathbf{x}^t$ denotes the solution estimate at iteration t, $t = 0, 1, ...$; $\mathbf{\Psi} : \mathbb{R}^d \mapsto \mathbb{R}^d$ is a general nonlinear map; $\alpha_t > 0$ is the employed step size; $\boldsymbol{\nu}^t \in \mathbb{R}^d$ is a zero-mean gradient noise; and $\mathbf{x}^0$ is an arbitrary deterministic point in $\mathbb{R}^d$.

We will specify further ahead the assumptions that we make on the step size α_t, the map $\mathbf{\Psi}$ and the noise $\boldsymbol{\nu}^t$. Some examples of commonly used maps $\mathbf{\Psi}$ that fall within our framework are the following:

1. Sign gradient: $[\mathbf{\Psi}(\mathbf{w})]_i = \text{sign}(w_i)$, $i = 1, ..., d$;
2. Component-wise clipping: $[\mathbf{\Psi}(\mathbf{w})]_i = w_i$, for $|w_i| \leq m$; $[\mathbf{\Psi}(\mathbf{w})]_i = m$, for $w_i > m$, and $[\mathbf{\Psi}(\mathbf{w})]_i = -m$, for $w_i < -m$, for some constant $m > 0$.
3. Component-wise quantization: for each $i = 1, ..., d$, we let $[\mathbf{\Psi}(\mathbf{w})]_i = r_j$, for $w_i \in (q_{j-1}, q_j]$, $j = 1, ..., J$, where $-\infty = q_0 < q_1 < ... < q_J = +\infty$, J is a positive integer, and the r_j's and q_j's are chosen such that each component nonlinearity is an odd function, i.e., $[\mathbf{\Psi}(\mathbf{w})]_i = -[\mathbf{\Psi}(-\mathbf{w})]_i$, for each i and for each $\mathbf{w}$;
4. Normalized gradient: $\mathbf{\Psi}(\mathbf{w}) = \frac{\mathbf{w}}{\|\mathbf{w}\|}$, for $\mathbf{w} \neq 0$, and $\mathbf{\Psi}(0) = 0$;
5. Clipped gradient: $\mathbf{\Psi}(\mathbf{w}) = \mathbf{w}$, for $\|\mathbf{w}\| \leq M$, and $\mathbf{\Psi}(\mathbf{w}) = \frac{\mathbf{w}}{\|\mathbf{w}\|} M$, for $\|\mathbf{w}\| > M$, for some constant $M > 0$.

Other nonlinearity choices are also introduced ahead (see Section 7).

We next discuss the various possible sources of the gradient noise $\boldsymbol{\nu}^t$. First, the noise may arise due to utilizing a search direction with respect to a data sample. That is, a common search direction in machine learning algorithms is the gradient of the loss with respect to a single data point $\mathbf{d}_i$[2]: $\mathbf{g}_i(\mathbf{x}) = \nabla \ell(\mathbf{x}; \mathbf{d}_i) + \nabla$[illegible]. In case of the risk function (2.2), $\boldsymbol{d}_i$ is drawn from distribution P; in case of the empirical risk, $\boldsymbol{d}_i$ can be, e.g., drawn uniformly at random from the set of data points $\boldsymbol{d}_j$, $j = 1, ..., n$, with repetition along iterations. In both cases, the corresponding gradient noise equals $\boldsymbol{\nu} = \mathbf{g}_i(\mathbf{x}) - \nabla f(\mathbf{x})$. Several recent studies indicate that noise $\boldsymbol{\nu}$ exhibits heavy tails on many real data sets, e.g, [32, 17, 37]. (See also Section 7).

We also comment on other possible sources of gradient noise. The noise may be added on purpose to the gradient [illegible]$\mathbf{x})$ for improving privacy of an SGD-based learning process, e.g., [illegible]. Also, the noise $\boldsymbol{\nu}^t$ may model random computational perturbations or inexact calculations in evaluating a gradient $\nabla f(\mathbf{x})$.

3. Main results: Component-wise Nonlinearities. Section 3 provides analysis of the nonlinear SGD method for component-wise nonlinearities. That is, we consider here maps $\mathbf{\Psi} : \mathbb{R}^d \mapsto \mathbb{R}^d$ of the form $\mathbf{\Psi}(w_1, ..., w_d) = (\Psi(w_1), ..., \Psi(w_d))^\top$, for any $\mathbf{w} \in \mathbb{R}^d$, where (somewhat abusing notation) we denote by $\Psi : \mathbb{R} \mapsto \mathbb{R}$ the component-wise nonlinearity. In this setting, we establish for (2.3) almost sure convergence and evaluate the MSE convergence rate and the asymptotic covariance of the method. In more detail, we consider a probability space $(\Omega, \mathcal{F}, P)$, where $\omega \in \Omega$ is a canonical element. For each $t = 0, 1, ...$, $\boldsymbol{\nu}^t : \Omega \mapsto \mathbb{R}^d$ is a random vector defined on $(\Omega, \mathcal{F}, P)$. We also denote by $\mathcal{F}_t$, $t = 0, 1, ...$, the σ-algebra generated by random vectors $\{\boldsymbol{\nu}^s\}$, $s = 0, ..., t$. Clearly, in view of (2.3), $\mathbf{x}^{t+1}$ is measurable with respect to $\mathcal{F}_t$, $t = 0, 1, ...$ We make the following assumptions; they follow the noise and nonlinearity framework similar to [30].

ASSUMPTION 3 (Gradient noise). *For the gradient noise random vector sequence $\{\boldsymbol{\nu}^t\}$ in (2.3), $t = 0, 1, ...$, $\boldsymbol{\nu}^t \in \mathbb{R}^d$, we assume the following.*

1. *The sequence of random vectors $\{\boldsymbol{\nu}^t\}$ is independent identically distributed*

[2]Similar considerations hold for a loss with respect to a mini-batch of data points; this discussion is abstracted for simplicity.

(i.i.d.) Also, random variables ν_i^t are mutually independent across $i = 1, ...d$;

2. *Each component ν_i^t, $i = 1, ..., d$, of vector $\boldsymbol{\nu}^t = (\nu_1^t, ..., \nu_d^t)^\top$ has a probability density function $p(u)$, $p : \mathbb{R} \mapsto \mathbb{R}_+$.*
3. *The pdf p is symmetric, i.e., $p(u) = p(-u)$, for any $u \in \mathbb{R}$ with $\int |u| p(u) du < +\infty$, and $p(u) > 0$ for $|u| \leq c_p$, for some constant $c_p > 0$.*

Note that Assumption 3 implies that $\boldsymbol{\nu}^t$ is zero-mean, for all t, and that $\boldsymbol{\nu}^t$ and $\mathbf{x}^t$ are mutually independent, for all t. For a class of unbounded nonlinearities Ψ that obey Assumption 6 ahead, we will additionally require the following.

Assumption 4. *The gradient noise variance $\sigma_\nu^2 = \int_{-\infty}^{+\infty} u^2 p(u) du < +\infty$.*

Assumption 3 requires that the noise vector is i.i.d. across its components $i = 1, ..., d$ which may be restrictive in certain scenarios. For the global MSE analysis, these assumptions can be relaxed; see ahead the remark after Theorem 3.2 and Appendix C.

Regarding noise pdf $p(u)$, except for strictly positivite values in the vicinity of zero (a very mild assumption), we require that the noise pdf is symmetric. Examples of the distributions that satisfy Assumption 3 include, e.g., a Gaussian zero-mean pdf or a Laplace zero-mean pdf with strictly positive variances, and heavy-tail zero-mean symmetric α-stable distributions [3]. [3] On the other hand, $p(u)$ may not be symmetric if, e.g., it is a mixture of some standard distributions. For example, consider random variable ν that is sampled from $\mathbb{N}(-m_1, \sigma^2)$ with probability $p = \frac{m_2}{m_1 + m_2}$ and it is sampled from $\mathbb{N}(m_2, \sigma^2)$ with probability $1 - p$, for some $m_1 \neq m_2$, $m_1, m_2 > 0$, and $\sigma > 0$. Then, clearly, ν is zero-mean but does not have a symmetric pdf.

Assumption 5 (Nonlinearity Ψ). *Function $\Psi : \mathbb{R} \mapsto \mathbb{R}$ is continuous (except possibly on a point set with Lebesgue measure of zero), monotonically non-decreasing and odd function, i.e., $\Psi(-w) = -\Psi(w)$, for any $w \in \mathbb{R}$. Moreover, Ψ is piece-wise differentiable. Finally, Ψ is either discontinuous at zero, or $\Psi(u)$ is strictly increasing for $u \in (-c_\Psi, c_\Psi)$, for some $c_\Psi > 0$.*

In addition, we impose one of the Assumptions 6 or 7 below.

Assumption 6. *$|\Psi(w)| \leq C_1 (1 + |w|)$, for any $w \in \mathbb{R}$, for some constant $C_1 > 0$.*

Assumption 7. *$|\Psi(w)| \leq C_2$, for some constant $C_2 > 0$.*

Assumption 3 and Assumption 5 are imposed throughout the paper. Assumption 4 is imposed when Assumption 6 holds, i.e., for the nonlinearities Ψ that can have unbounded outputs. When Assumption 7 is imposed, then Assumption 4 is not required.

Note that, provided that Assumption 7 holds, we require only a finite first moment of the gradient noise, while the moments of α-order, $\alpha > 1$, may be infinite, hence allowing for heavy-tail noise distributions. For example, the gradient noise variance can be infinite. Assumption 5 holds for several interesting component-wise nonlinearities, like, e.g., the sign gradient, component-wise clipping, and quantization schemes introduced in Section 2. Note also that Assumption 5 encompasses a broad range of component-wise nonlinearities, beyond the examples in Section 2. (For example, see Section 7 for the tanh and a bi-level quantization nonlinearity.)

Let us define function $\phi : \mathbb{R} \mapsto \mathbb{R}$, as follows. For a fixed (deterministic) point $w \in \mathbb{R}$, $\phi(w)$ is defined by:

$$\phi(w) = \mathbb{E}\left[\Psi(w + \nu_1^0)\right] = \int \Psi(w + u) p(u) du, \tag{3.1}$$

[3]A random variable Z has a symmetric α-stable zero-mean distribution with scale parameter $\sigma > 0$ if its characteristic function takes the form: $\mathbb{E}\left[\exp(i\, u\, Z)\right] = \exp(-\sigma^\alpha |u|^\alpha)$, $u \in \mathbb{R}$, $\alpha \in [0, 2]$.

where the expectation is taken with respect to the distribution of a single entry of the gradient noise at any iteration, i.e., with respect to pdf $p(u)$. Intuitively, the nonlinearity ϕ is a convolution-like transformation of the nonlinearity Ψ, where the convolution is taken with respect to the gradient noise pdf $p(u)$. As we will see ahead, the nonlinearity ϕ plays an effective role in determinining the performance of algorithm (2.3). We now state the main results on (2.3) with component-wise nonlinearities, including the results on a.s. convergence, MSE rate, and asymptiotic normality. We start with the following Theorem that establishes a.s. convergence.

Theorem 3.1 (Almost sure convergence: Component-wise nonlinearity). *Consider algorithm (2.3) for solving optimization problem (2.1), and let Assumptions 1, 2, 3, 5, and 7 hold. Further, let the positive step-size sequence $\{\alpha_t\}$ be square summable, non-summable: $\sum \alpha_t = +\infty$; $\sum \alpha_t^2 < +\infty$. Then, the sequence of iterates $\{\mathbf{x}^t\}$ generated by algorithm (2.3) converges almost surely to the solution $\mathbf{x}^\star$ of the optimization problem (2.1). Moreover, the result holds if Assumption 7 is replaced with Assumption 6, and Assumption 4 is additionally imposed.*

Theorem 3.1 establishes a.s. convergence of the nonlinear SGD scheme (2.3) under a general setting for the component-wise nonlinearities and gradient noise. For example, provided that the output of the nonlinearity Ψ is bounded, algorithm (2.3) converges even when the gradient noise may not have a finite α-moment, for any $\alpha > 1$. (Hence it may have an infinite variance). In contrast, as shown in Appendix B, the linear SGD (algorithm (2.3) with $\boldsymbol{\Psi}$ being the identity function) generates a sequence of solution estimates with infinite variances, provided that the variance of $p(u)$ is infinite.

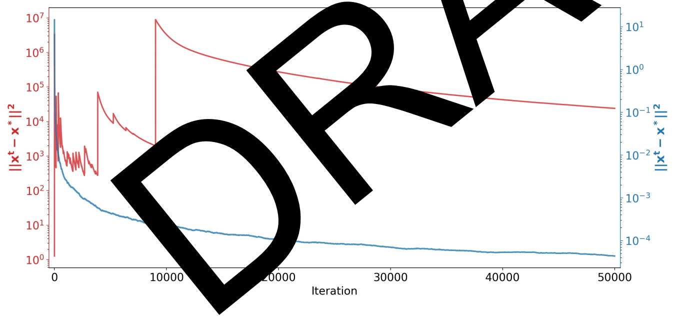


Fig. 3.1: Illustration of Theorem 3.1: estimated MSE versus iteration counter for the nonlinear SGD in (2.3) with component-wise sign nonlinearity (blue line) and the linear SGD (red line).

Example 3.1. Figure 3.1 illustrates Theorem 3.1 with a simulation example. We consider a strongly convex quadratic function $f : \mathbb{R}^d \mapsto \mathbb{R}$, $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b}^\top \mathbf{x}$, where $\mathbf{A} \in \mathbb{R}^{d\times d}$ is a (symmetric) positive definite matrix, $d = 16$, and quantities $\mathbf{A}, \mathbf{b}$ are generated at random. We consider algorithm (2.3) with the component-wise sign nonlinearity and the linear SGD. The gradient noise has a heavy-tailed pdf given by:

$$p(u) = \frac{\alpha - 1}{2(1 + |u|)^\alpha}, \tag{3.2}$$

for $u \in \mathbb{R}$ and $\alpha > 2$. Note that the distribution (3.2) does not have a finite $\alpha - 1$ moment and has finite moments of r-th order for $r < \alpha - 1$. We set in simulation $\alpha = 2.05$. Note that, in this case, the gradient noise has infinite variance. We initialize both the linear and nonlinear algorithm with $\mathbf{x}^0 = 0$, and we let step size $\alpha_t = \frac{1}{t+1}$. Figure 3.1 shows an estimate of MSE, i.e., of the quantity $\mathbb{E}[\|\mathbf{x}^t - \mathbf{x}^\star\|^2]$, obtained by averaging results from 100 sample paths. The red line corresponds to the linear SGD, while the blue line corresponds to the nonlinear SGD with the component-wise sign nonlinearity. As predicted by Theorem 3.1, the nonlinear SGD drives the MSE to zero, while the linear SGD does not seem to provide a meaningful solution estimate sequence.
We next establish the mean square error (MSE) convergence rate of algorithm (2.3).

THEOREM 3.2 (MSE convergence: Component-wise nonlinearity). *Consider algorithm (2.3) for solving optimization problem (2.1), and let Assumptions 1, 3, 5, and 7 hold. Further, let the step-size sequence $\{\alpha_t\}$ be $\alpha_t = a/(t+1)^\delta$, $a > 0$, $\delta \in (0.5, 1)$. Then, for the sequence of iterates $\{\mathbf{x}^t\}$ generated by algorithm (2.3), it holds that $\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^\star\|^2\right] = O(1/t^\zeta)$, or equivalently, $\mathbb{E}\left[f(\mathbf{x}^t) - f^\star\right] = O(1/t^\zeta)$. Here, $\zeta < 1$ is any positive number such that $\zeta < \min\left(2\delta - 1, \frac{a\,(1-\delta)\xi\,\phi'(0)\mu}{L\,(c\,[illegible]\,2\sqrt{d} + \|\mathbf{x}^0 - \mathbf{x}^\star\|)}\right)$, and constant $\xi > 0$ is such that $\phi(a) \geq \frac{\phi'(0)}{2}\,a$, for any $a \in [0, [illegible])$. Furthermore, if Assumptions 1, 3, 5, and 6, and 4 hold, let $\alpha_t = \frac{a}{(t+1)^\delta}$, $\delta \in (0.5, 1]$, and assume that $\inf_{a \neq 0} \frac{|\Psi(a)|}{|a|} > 0$. Then, there holds that $\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^\star\|^2\right] = O(1/t^\delta)$, or equivalently, $\mathbb{E}\left[f(\mathbf{x}^t) - f^\star\right] = O(1/t^\delta)$. In particular, for $\delta = 1$, we obtain the $O(1/t)$ MSE rate.*

Remark. The MSE convergence $O(1/t^{\zeta'})$, for some $\zeta' \in (0,1)$, continues to hold under the same set of assumptions as in Theorem 3.2 but with a relaxed version of Assumption 3, where we no longer require that the gradient noise vector has mutually independent components. More precisely, we allow for an i.i.d. noise vector sequence $\{\boldsymbol{\nu}^t\}$, $\boldsymbol{\nu}^t \in \mathbb{R}^d$, that has a symmetric joint pdf $p : \mathbb{R}^d \mapsto \mathbb{R}$, $p(\mathbf{u}) = p(-\mathbf{u})$, for any $\mathbf{u} \in \mathbb{R}^d$, that is strictly positive for $\|\mathbf{u}\| \leq u_0$ for some $u_0 > 0$. In that case, effectively, the role of function ϕ in Theorem 3.2 is replaced by functions $w \mapsto \phi_i(w)$, $w \in \mathbb{R}$, $i = 1, ..., d$, where $\phi_i(w) = \int \Psi(w + u)\,p_i(u)du$, and $p_i : \mathbb{R} \mapsto \mathbb{R}$ is the marginal pdf of the i-th component associated with the joint pdf $p : \mathbb{R}^d \mapsto \mathbb{R}$. (See Appendix C.)

For the bounded nonlinearity case (e.g., sign gradient, component-wise clipping, quantization nonlinearity) and the heavy-tail noise (only the first noise moment assumed to be finite), the nonlinear SGD (2.3) achieves a global sublinear MSE rate $O(1/t^\zeta)$, $\zeta \in (0,1)$. On the other hand, for the finite variance case and an unbounded nonlinearity, the nonlinear SGD (2.3) achieves a global MSE rate $O(1/t)$ provided that $\inf_{w \neq 0} \frac{|\Psi(w)|}{|w|} > 0$. This is the best achievable rate and equal to that of the linear SGD in the same setting. Furthermore, by Theorem 3.3 ahead, the nonlinear SGD (2.3) with bounded outputs under the heavy-tail noise achieves *locally*, in the weak convergence sense, the faster $O(1/t)$ rate. This is again in the setting where the linear SGD fails.

Example 3.2. We next illustrate the value ζ in Theorem 3.2 on the family of heavy-tailed pdfs given in (3.2). To be specific, consider the sign nonlinearity $\Psi(w) = \text{sign}(w)$. Then, it is easy to show that: $\phi(w) = 2\int_0^w p(u)du$, $\phi'(0) = 2\,p(0)$, $\xi \geq 2^{1/\alpha} - 1 \approx \frac{1}{\alpha}$. Using the above calculations, we can see that, for a large a, ζ can be approximated as $\min\{2\delta - 1,\ \frac{\mu}{L}\frac{1-\delta}{\sqrt{d}}\frac{\alpha-1}{\alpha}\}$.

We also compare the rate ζ with the analysis in [37] that is closest to our setting with respect to existing work. Modulo the differences in the assumptions of the assumed settings here and in [37], the rate in [37], when adapted to the noise pdf in Example 3.1, reads as follows: $\frac{2(r-1)}{r}$, where r is any number such that $r \leq \min\{\alpha - 1, 2\}$. When compared with ζ, the rate in [37] is clearly better for α above a threshold. However, as α decreases and approaches the value 2, the rate achieved here stays bounded away from zero and approaches the quantity: $\min\left\{2\delta - 1, \frac{1}{2}\frac{\mu}{L}\frac{1-\delta}{\sqrt{d}}\right\}$. In contrast, the rate in in [37] approaches zero as α approaches 2. [4]

Example 3.3. We continue to assume the noise pdf in (3.2), but here we consider the component-wise clipping nonlinearity Ψ with saturation value m. For simplicity, we take $m > 1$, while similar bounds can be obtained for $m \leq 1$ as well. It can be shown that the rate ζ can be estimated as (see Appendix E):

$$\min\left\{2\delta - 1, \frac{\mu}{L\sqrt{d}}\frac{(1-\delta)(m-1)(1-(m+1)^{-\alpha})}{m}\right\}. \tag{3.3}$$

The above α-dependent estimate can be replaced with a more conservative rate that holds for any $\alpha > 2$: $\min\{2\delta - 1, \frac{\mu}{L\sqrt{d}} \frac{(1-\delta)(m-1)(1-(m+1)^{-2})}{m}\}$. We again compare the rate achieved by the proposed method with the rate from [37] that equals $\frac{2(r-1)}{r}$, $r < \min\{\alpha - 1, 2\}$. We can see that the rate in [37] is better than (3.3) for α above a threshold. On the other hand, when α decreases to 2, the rate of [37] approaches zero, while (3.3) becomes better and stays bounded away from zero.

We next establish asymptotic normality of (2.3).

THEOREM 3.3 (Asymptotic normality: Component-wise nonlinearity). *Consider algorithm* (2.3) *for solving optimization problem* (2.1)*, and let Assumptions* 1*,* 2*,* 3*,* 5*, and* 7 *hold. Further, let the step-size sequence* $\{\alpha_t\}$ *equal:* $\alpha_t = a/(t+1)$*,* $t = 0, 1, ...,$ *with parameter* $a > \frac{1}{2\phi'(0)\mu}$*. Then the sequence of iterates* $\{\mathbf{x}^t\}$ *generated by algorithm* (2.3) *is asymptotically normal, and there holds:*

$$\sqrt{t+1}(\mathbf{x}^t - \mathbf{x}^\star) \xrightarrow{d} \mathbb{N}(0, \mathcal{S}), \tag{3.4}$$

where $\xrightarrow{d}$ *designates convergence in distribution. The asymptotic covariance* $\mathcal{S}$ *of the multivariate normal distribution* $\mathbb{N}(0, \mathcal{S})$ *is given by:*

$$\mathcal{S} = a^2 \int_{\nu=0}^{\infty} e^{\nu\mathbf{\Sigma}} \mathcal{S}_0 e^{\nu\mathbf{\Sigma}} d\nu = a^2 \sigma_\psi^2 \left[2a\phi'(0)\nabla^2 f(x^\star) - \mathbf{I}\right]^{-1},$$

where:

$$\mathcal{S}_0 = \sigma_\Psi^2 \,\mathbf{I}, \;\; \sigma_\Psi^2 = \int |\Psi(v)|^2 p(v) dv, \;\; \Sigma = \frac{1}{2}\mathbf{I} - a\,\phi'(a)\nabla^2 f(\mathbf{x}^\star). \tag{3.5}$$

Moreover, the same result holds when Assumption 7 *is replaced with Assumption* 6*, and Assumption* 4 *is additionally imposed.*

[4]It is worth noting that reference [37] establishes certain tightness results on the rate achieved therein, by providing a "hard" problem example where the mean squared error after t iterations is $\Omega(1/t^{\frac{2(r-1)}{r}})$. However, this does not contradict our results due to the different sets of Assumptions made here and in [37]. Most notably, [37] assumes bounded moments of *gradients* and allow for *dependence* between the current point $\mathbf{x}^t$ and the gradient noise $\boldsymbol{\nu}^t$. In fact, the "hard example" construction in the proof of Theorem 5 in [37] constructs $\boldsymbol{\nu}^t$ as an explicit function of $\mathbf{x}^t$.

Theorem 3.3 establishes asymptotic normality of (2.3) and, moreover, it gives an exact expression for the asymptotic covariance $\mathcal{S}$ in (3.3), that basically corresponds to the constant in the $1/t$ variance decay near the solution. The asymptotic covariance value (3.3) reveals an interesting tradeoff with respect to the effect of the nonlinearity Ψ. We provide some insights into the tradeoff through examples below.

Example 3.4 Figure 3.2 illustrates Theorem 3.3 for the nonlinear SGD in (2.3) with component-wise sign nonlinearity and the same simulation setting used for the numerical illustration of Theorem 3.1 and step-size $\alpha_t = \frac{10}{t+1}$. The red line plots quantity $\frac{t}{d}\|\mathbf{x}^t - \mathbf{x}^\star\|^2$ estimated through 100 sample path runs. This quantity estimates the constant in the $1/t$ per-entry asymptotic variance decay, i.e., it is a numerical estimate of the per-entry asymptotic variance $\frac{\mathrm{trace}(\mathcal{S})}{d}$, where $\mathcal{S}$ is given in Theorem 3.3. The blue horizontal line marks the value $\frac{\mathrm{trace}(\mathcal{S})}{d}$. We can see that the simulation matches well the theory.

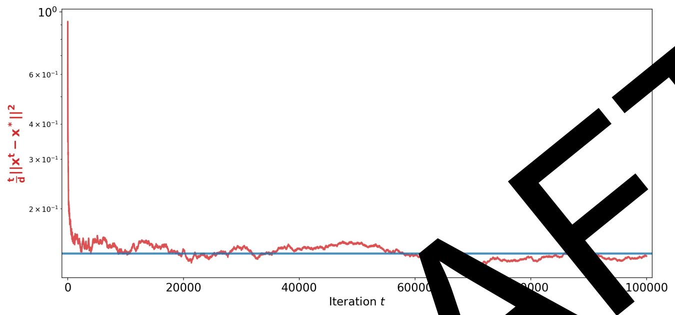


Fig. 3.2: Illustration of Theorem 3.3: Monte Carlo estimate of per-entry asymptotic variance (red line) and the theoretical per-entry asymptotic variance in Theorem 3.3 (blue line).

Example 3.5. We compare the linear SGD and the nonlinear SGD with component-wise clipping. For illustration and simplification of calculations, we consider the special case when $\nabla^2 f(\mathbf{x}^\star)$ is a symmetric matrix with all eigenvalues equal to one. Then, it is straightforward to show that the per-entry asymptotic variance for the best choice of parameter a over the admissible set of values equals:

$$\inf_{a > \frac{1}{2\phi'(0)}} \mathrm{trace}\,(\mathcal{S}) = \frac{\sigma_\Psi^2}{(\phi'(0))^2}. \tag{3.6}$$

Here, for the linear SGD i.e., when $\Psi(a) = a$, we have that $\sigma_\Psi^2 = \int a^2 p(a) da$ equals the gradient noise (per component) variance σ_ν^2, and $\phi'(0) = 1$, and so (3.6) equals σ_ν^2. Now, consider the coordinate-wise clipping, with $\Psi(a) = a$ for $|a| \le m$ and $\Psi(a) = \mathrm{sign}(a)\, m$, for $|a| > m$, for some $m > 0$. Then, we have: $\sigma_\Psi^2 = m^2 - 2\int_0^m (m^2 - v^2) p(v) dv$, and $\phi'(0) = 2\int_0^m p(v)dv$. (See Appendix F for the derivation.) Note that the case $m \to \infty$ corresponds to the linear SGD case. Consider now the tradeoff with respect to the choice of m. Clearly, taking a smaller m has a positive effect on the numerator in (3.6) (it suppresses the noise effect). On the other hand, reducing m has a negative effect on the denominator in (3.6); that is, it reduces the value $\phi'(0)$ – intuitively, it "lowers the quality" of the search direction utilized with (2.3). One needs to choose the nonlinearity, i.e., the parameter m, optimally, to strike the best balance here. Clearly, for larger gradient noise σ_ν^2, we should pick a smaller value of m. Note also that, when σ_ν^2 is infinite, the linear SGD has an infinite asymptotic variance in (3.6), while the nonlinear SGD with any $m \in (0, \infty)$ has a finite asymptotic variance.

Example 3.6. We continue to assume the simplified setting when the per-entry asymptotic variance equals (3.6). We consider the sign gradient nonlinearity and the class of heavy-tail gradient noise distributions in (3.2). It can be shown that here: $\sigma_\Psi^2 = 1$; $\sigma_\nu^2 = \frac{2}{(\alpha-3)(\alpha-2)}$, for $\alpha > 3$ and $\sigma_\nu^2 = \infty$, else; and $\phi'(0) = \alpha - 1$. (See Appendix G.) Therefore, for the sign gradient, the best achievable per entry asymptotic variance equals $\frac{1}{(\alpha-1)^2}$, while for the linear SGD it equals $\frac{2}{(\alpha-2)(\alpha-3)}$ for $\alpha > 3$, and is infinite for $\alpha \in (2, 3]$. Hence, we can see for the considered example that the sign gradient outperforms the linear SGD for any $\alpha > 2$, and the gap becomes larger as α gets smaller.

Example 3.7. We still consider the simplified setting of (3.6). If the noise pdf $p(u)$ is known, then, following [30], we can find a globally optimal nonlinearity that minimizes (3.6) that takes the form: $\Psi(a) = -\frac{d}{da}\ln(p(a))$. The corresponding optimal asymptotic variance equals the Fisher information associated with the pdf $p(u)$.

4. Main results: Joint Nonlinearities.

We now consider algorithm (2.3) for a nonlinearity $\mathbf{\Psi} : \mathbb{R}^d \mapsto \mathbb{R}^d$ that cannot be decoupled into (equal) component-wise nonlinearities $\Psi : \mathbb{R} \mapsto \mathbb{R}$, as it was possible before. More precisely, we make the following assumptions on the gradient noise $\boldsymbol{\nu}^t$ and the nonlinear map $\mathbf{\Psi} : \mathbb{R}^d \mapsto \mathbb{R}^d$. Recall also filtration $\mathcal{F}_t$ in Section 3.

ASSUMPTION 8. *[Gradient noise] For the gradient noise sequence $\{\boldsymbol{\nu}^t\}$, we assume the following:*

1. *The sequence of random vectors $\{\boldsymbol{\nu}^t\}$ is i.i.d. Moreover, $\boldsymbol{\nu}^t$ has a joint symmetric pdf $p(\mathbf{u})$, $p : \mathbb{R}^d \mapsto \mathbb{R}$, i.e., $p(\mathbf{u}) = p(-\mathbf{u})$, for any $\mathbf{u} \in \mathbb{R}^d$ with $\int \|\mathbf{u}\| p(\mathbf{u}) d\mathbf{u} < \infty$;*
2. *There exists a positive constant B_0 such that, for any $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{x} \neq 0$, for any $A \in (0, 1]$, there exists $\lambda = \lambda(A) > 0$, such that $\int_{\mathcal{J}_A} p(\mathbf{u}) d\mathbf{u} > \lambda(A)$, where $\mathcal{J}_A = \{\mathbf{u} \in \mathbb{R}^d : \frac{\mathbf{u}^\top \mathbf{x}}{\|\mathbf{u}\|\|\mathbf{x}\|} \in [0, A], \|u\| \leq B_0\}$.*[5]

Assumption 8 allows for a heavy-tailed noise vector whose components can be mutually dependent. Condition 2 in Assumption 8 is mild; it says that the joint pdf $p(\mathbf{u})$ is "non-degenerate" in the sense that, along each "direction" (determined by arbitrary nonzero vector $\mathbf{x}$), the intersection of the set $\{\frac{\mathbf{u}^\top \mathbf{x}}{\|\mathbf{u}\|\|\mathbf{x}\|} \in [0, A]\}$ and the ball $\{\|\mathbf{u}\| \leq B_0\}$ consumes a positive mass of the joint pdf $p(\mathbf{u})$.

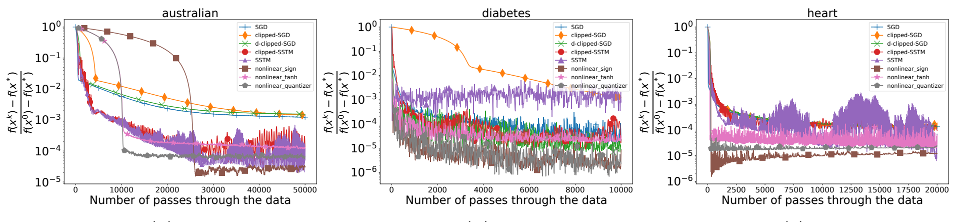
We make the following assumption on the joint nonlinearity.

ASSUMPTION 9 (Nonlinearity $\mathbf{\Psi}$). *The nonlinear map $\mathbf{\Psi} : \mathbb{R}^d \mapsto \mathbb{R}^d$ takes the following form: $\mathbf{\Psi}(\mathbf{w}) = \mathbf{w}\mathcal{N}(\|\mathbf{w}\|)$, where function $\mathcal{N} : \mathbb{R}_+ \mapsto \mathbb{R}_+$ satisfies the following: $\mathcal{N}$ is non-increasing and continuous except possibly on a point set with Lebesgue measure of zero with $\mathcal{N}(q) > 0$, for any $q > 0$. The function $q\mathcal{N}(q)$ is non-decreasing.*

In addition, we assume that either Assumption 10 or Assumption 11 holds.

ASSUMPTION 10. *$\|\mathbf{\Psi}(\mathbf{w})\| \leq C_2'$, for any $\mathbf{w} \in \mathbb{R}^d$, for some $C_2' > 0$.*

[5]The integration set $\mathcal{J}_A$ also includes the point $\mathbf{u} = 0$. In other words, for compact notation here and throughout the paper, we write $\frac{\mathbf{u}^\top \mathbf{x}}{\|\mathbf{u}\|\|\mathbf{x}\|} \in [0, A]$ instead of $0 \leq \mathbf{u}^\top \mathbf{x} \leq A\,\|\mathbf{u}\|\|\mathbf{x}\|$.



(a) (b) (c)

Fig. 4.1: Comparison of the optimization algorithms across different datasets

Assumption 11. $\|\mathbf{\Psi}(\mathbf{w})\| \leq C_1'(1+\|\mathbf{w}\|)$, *for any* $\mathbf{w} \in \mathbb{R}^d$, *for some* $C_1' > 0$

There are many nonlinearities that satisfy the above Assumptions, including the normalized gradient and the clipped gradient discussed in Section 2. If Assumption 11 holds, then we additionally require the following.

Assumption 12. *There holds:* $\int \|\mathbf{u}\|^2 p(\mathbf{u}) d\mathbf{u} < \infty$.

For asymptotic normality in the joint nonlinearity case, we additionally impose the following.

Assumption 13. *Function* $\mathcal{N} : \mathbb{R}_+ \mapsto \mathbb{R}$ *is differentiable for any positive argument, i.e.,* $\mathcal{N}'(a)$ *exists for any* $a > 0$. *Furthermore,* $\sup_{a>0} \mathcal{N}(a) < +\infty$.

We first state Theorem 4.1 and Theorem 4.2 on the a.s. convergence and the MSE rate of algorithm (2.3), respectively; we then illustrate the results with examples.

Theorem 4.1 (A.s. convergence: Joint nonlinearity). *Consider algorithm* (2.3) *for solving optimization problem* (2.1), *and let Assumptions* 1, 2, 8, 9, *and* 10 *hold. Further, let the step-size sequence* $\{\alpha_t\}$ *be square-summable, non-summable. .Then, for the sequence of iterates* $\{\mathbf{x}^t\}$ *generated by algorithm* (2.3), *it holds that* $\mathbf{x}^t \to \mathbf{x}^\star$, *a.s. Moreover, the result continues to hold if Assumption* 10 *is replaced with Assumption* 11, *and Assumption* 12 *is additionally imposed.*

We now state our MSE rate result for the joint nonlinearity case.

Theorem 4.2 (MSE convergence rate: Joint nonlinearity). *Consider algorithm* (2.3) *for solving optimization problem* (2.1), *and let Assumptions* 1, 8, 9, *and* 10 *hold. Further, let the step-size sequence* $\{\alpha_t\}$ *be* $\alpha_t = a/(t+1)$, $a > 0$, $\delta \in (0.5, 1)$. *Then,* $\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^\star\|^2\right] = O(1/t^\zeta)$, *or equivalently,* $\mathbb{E}\left[f(\mathbf{x}^t) - f^\star\right] = O(1/t^\zeta)$. *Here,* $\zeta \in (0,1)$ *is any positive number smaller than:* $\min\left\{2\delta - 1, \frac{4\,a\,\mu\,(1-\kappa)\lambda(\kappa)(1-\delta)\mathcal{N}(1)}{L\,(\,a\,C_2'+\|\mathbf{x}^0\|+\|\mathbf{x}^\star\|\,)+B_0}\right\}$, *where* κ *is an arbitrary constant in* $(0,1)$, *and we recall quantities* B_0 *and* $\lambda(\kappa)$ *in Assumption* 8; μ *and* L *in Assumption* 1; *and* C_2' *in Assumption* 9. *In alternative, let Assumptions* 1, 8, 9, 11, *and* 12 *hold. Let* $\alpha_t = \frac{a}{(t+1)^\delta}$, $\delta \in (0.5, 1]$, *and assume that* $\inf_{\mathbf{w}\neq 0} \frac{\|\mathbf{\Psi}(\mathbf{w})\|}{\|\mathbf{w}\|} > 0$. *Then,* $\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^\star\|^2\right] = O(1/t^\delta)$, *or equivalently,* $\mathbb{E}\left[f(\mathbf{x}^t) - f^\star\right] = O(1/t^\delta)$. *In particular, for* $\delta = 1$ *and a sufficiently large parameter* a, *we obtain the* $O(1/t)$ *MSE rate.*

Example 4.1. We illustrate the rate ζ in Theorem 4.2 for the gradient clipping nonlinearity with floor level $M > 0$. We consider an arbitrary joint pdf $p : \mathbb{R}^d \mapsto \mathbb{R}_+$ that has "radial symmetry", i.e., $p(\mathbf{u}) = q(\|\mathbf{u}\|)$, where $q : \mathbb{R}_+ \mapsto \mathbb{R}_+$ is a given function. For example, we let:

$$p(\mathbf{u}) = q(\|\mathbf{u}\|),\ q(\rho) = \frac{(\alpha-2)(\alpha-1)}{2\pi}\frac{1}{(1+\rho)^{\alpha}},\ \rho \geq 0,\ \ \alpha > 3. \tag{4.1}$$

It can be shown that $p(\mathbf{u})$ in (4.1) has finite moments of order r, $r < \alpha - 2$, and it has infinite moments for $r \geq \alpha - 2$. It holds that (see Appendix H for derivations) the rate ζ can be estimated as: $\min\left\{2\delta - 1,\, (1-\delta)\frac{0.68\,\mu}{L}\right\}$. Hence, up to universal constants, the rate ζ is approximated as $\min\left\{2\delta-1,\, (1-\delta)\frac{\mu}{L}.\right\}$. It is easy to see that the same rate estimate can be obtained for the normalized gradient nonlinearity, under the same gradient noise setting.

We compare the rate estimate here with the rate for component-wise nonlinearities (e.g., component-wise clipping in Example 3.3) that is, up to universal constants, of order $\min\left\{2\delta-1,\, (1-\delta)\frac{\mu}{\sqrt{d}\,L}.\right\}$. We can see that, with the joint nonlinearity examples here, the rate is improved with respect to the component-wise nonlinearities by a factor $\sqrt{d}$. In other words, the rate estimate for the joint nonlinearities does not deteriorate with the dimension d increase. This may be intuitively explained by considering the sign component-wise nonlinearity and the normalized gradient. These two functions coincide for $d = 1$ (and this is reflected by the identical rate estimates we obtain here), but they become different for $d > 1$ (as also reflected by our obtained rate estimates). Intuitively, in the noiseless case the normalized gradient preserves "more information" about the exact gradient ("true search direction") than the component-wise sign function; hence, the difference in the estimated rates.

We now examine asymptotic normality for the joint nonlinearities case. We have the following theorem.

THEOREM 4.3 (Asymptotic normality: Joint nonlinearity). *Consider algorithm* (2.3) *for solving optimization problem* (2.1) *and let Assumptions 1, 2, 8, 9, 10, and 13 hold. Further, let the step-size sequence* $\{\alpha_t\}$ *equal* $\alpha_t = a/(t+1)$, $a > 0$. *Then:* $\sqrt{t+1}(\mathbf{x}^t - \mathbf{x}^\star) \xrightarrow{d} \mathcal{N}(0, \mathcal{S})$. *The asymptotic covariance* $\mathcal{S}$ *is given by* $\mathcal{S} = a^2 \int_0^\infty e^{v\,\boldsymbol{\Sigma}} \mathcal{S}_0 e^{v\,\boldsymbol{\Sigma}} dv$, *where* $\mathcal{S}_0 = \int \mathbf{u}\mathbf{u}^\top (\mathcal{N}(\|\mathbf{u}\|))^2 p(\mathbf{u}) d\mathbf{u}$; $\boldsymbol{\Sigma} = 1_{\overline{2\mathbf{I}+a\,\mathbf{B};}}$ $\mathbf{B} = -(\int \mathcal{N}(\|\mathbf{u}\|)p(\mathbf{u})d\mathbf{u} + \int_{\mathbf{u}\neq 0} \frac{\mathbf{u}\mathbf{u}^\top}{\|\mathbf{u}\|}\mathcal{N}'(\|\mathbf{u}\|)p(\mathbf{u})d\mathbf{u})\nabla^2 f(\mathbf{x}^\star)$, *and constant* $a > 0$ *in the step-size sequence is taken large enough such that matrix* $\boldsymbol{\Sigma}$ *is stable. Moreover, the result continues to hold if Assumption* 10 *is replaced with Assumption* 11, *and Assumption* 12 *is additionally imposed.*

Theorem 4.3 shows that asymptotic normality continues to hold for the joint nonlinearity case as well, provided that $\mathcal{N}(a)$ is differentiable for any $a > 0$ and that $\mathcal{N}$ is uniformly bounded from above.

5. Intermediate results and proofs: Component-wise nonlinearities.

This section provides proofs of Theorem 3.1, Theorem 3.2, and Theorem 3.3, accompanied with the required intermediate results. Subsection 5.1 presents some useful intermediate results on stochastic approximation and deterministic time-varying sequences; Subsection 5.2 deals with the asymptotic analysis (Theorem 3.1 and Theorem 3.3); and Subsection 5.3 considers MSE analysis (Theorem 3.2).

5.1. Stochastic approximation and time-varying sequences.

We present a useful result on single time scale stochastic approximation; see [26], Theorems 4.4.4 and 6.6.1.

THEOREM 5.1. *Let $\left\{\mathbf{x}^t \in \mathbb{R}^d\right\}$ be a random sequence that satisfies:*

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \alpha_t \left[\mathbf{r}(\mathbf{x}^t) + \boldsymbol{\gamma}\left(t+1, \mathbf{x}^t, \omega\right)\right], \tag{5.1}$$

where, $\mathbf{r}(\cdot) : \mathbb{R}^d \longmapsto \mathbb{R}^d$ is Borel measurable and $\{\boldsymbol{\gamma}(t,\mathbf{x},\omega)\}_{t\geq 0,\ \mathbf{x}\in\mathbb{R}^d}$ is a family of random vectors in $\mathbb{R}^d$, defined on a probability space $(\Omega, \mathcal{F}, \mathcal{P})$, and $\omega \in \Omega$ is a canonical element. Let the following sets of assumptions hold:
(B1)*: The function $\boldsymbol{\gamma}(t,\cdot,\cdot) : \mathbb{R}^d \times \Omega \longrightarrow \mathbb{R}^d$ is $\mathcal{B}^d \otimes \mathcal{F}$ measurable for every t; $\mathcal{B}^d$ is the Borel algebra of $\mathbb{R}^d$.*
(B2)*: There exists a filtration $\{\mathcal{F}_t\}_{t\geq 0}$ of $\mathcal{F}$, such that, for each t, the family of random vectors $\{\boldsymbol{\gamma}\left(t,\mathbf{x},\omega\right)\}_{\mathbf{x}\in\mathbb{R}^d}$ is $\mathcal{F}_t$ measurable, zero-mean and independent of $\mathcal{F}_{t-1}$.*
(B3)*: There exists a twice continuously differentiable function $V\left(\mathbf{x}\right)$ with bounded second order partial derivatives and a point $\mathbf{x}^\star \in \mathbb{R}^d$ satisfying: $V\left(\mathbf{x}^\star\right) = 0$, $V\left(\mathbf{x}\right) > 0$, $\mathbf{x} \neq \mathbf{x}^\star$, $\lim_{\|\mathbf{x}\|\to\infty} V\left(\mathbf{x}\right) = \infty$, $\sup_{\epsilon<\|\mathbf{x}-\mathbf{x}^\star\|<\frac{1}{\epsilon}} \left\langle \mathbf{r}\left(\mathbf{x}\right), \nabla V\left(\mathbf{x}\right)\right\rangle < 0$, for any $\epsilon > 0$.*
(B4)*: There exist constants $k_1, k_2 > 0$, such that,*

$$\left\|\mathbf{r}\left(\mathbf{x}\right)\right\|^2 + \mathbb{E}\left[\left\|\boldsymbol{\gamma}\left(t+1,\mathbf{x},\omega\right)\right\|^2\right] \leq k_1\left(1 + V\left(\mathbf{x}\right)\right) - \\ - k_2\langle\mathbf{r}\left(\mathbf{x}\right), \nabla V\left(\mathbf{x}\right)\rangle.$$

(B5)*: The weight sequence $\{\alpha_t\}$ satisfies $\alpha_t > 0$, $\sum_{t\geq 0}\alpha_t = \infty$, $\sum_{t\geq 0}\alpha_t^2 < \infty$.*
(C1)*: The function $\mathbf{r}\left(\mathbf{x}\right)$ admits the representation*

$$\mathbf{r}\left(\mathbf{x}\right) = \boldsymbol{B}\left(\mathbf{x} - \mathbf{x}^\star\right) + \delta\left(\mathbf{x}\right), \tag{5.2}$$

where $\lim_{\mathbf{x}\to\mathbf{x}^\star} \frac{\|\delta(\mathbf{x})\|}{\|\mathbf{x}-\mathbf{x}^\star\|} = 0$.
(C2)*: The step-size sequence, $\{\alpha_t\}$ is of the form, $\alpha_t = \frac{a}{t+1}$, for any $t \geq 0$, where $a > 0$ is a constant.*
(C3)*: Let $\mathbf{I}$ be the $d\times d$ identity matrix and a, $\mathbf{B}$ as in C2 and C1, respectively. Then, the matrix $\Sigma = a\mathbf{B} + \frac{1}{2}\mathbf{I}$ is stable.*
(C4)*: The entries of the matrices, for any $t \geq 0, \mathbf{x} \in \mathbb{R}^d$, $\mathbf{A}\left(t,\mathbf{x}\right) = \mathbb{E}[\,\boldsymbol{\gamma}\left(t+1,\mathbf{x},\omega\right)$ $\boldsymbol{\gamma}^\top\left(t+1,\mathbf{x},\omega\right)]$ are finite, and the following limit exists: $\lim_{t\to\infty,\ \mathbf{x}\to\mathbf{x}^\star}\mathbf{A}\left(t,\mathbf{x}\right) = \mathcal{S}_0$.*
(C5)*: There exists $\epsilon > 0$, such that*

$$\lim_{R\to\infty}\ \sup_{\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}\ \sup_{t\geq 0}\int_{\|\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)\|>R}\left\|\boldsymbol{\gamma}\left(t+1,\mathbf{x},\omega\right)\right\|^2 dP = 0. \tag{5.3}$$

Then we have the following:

*Let Assumptions **(B1)-(B5)** hold for $\{\mathbf{x}^t\}$ in (5.1). Then, starting from an arbitrary initial state, the process $\{\mathbf{x}^t\}$ converges a.s. to $\mathbf{x}^\star$.*

*The normalized process, $\left\{\sqrt{t}\left(\mathbf{x}^t - \mathbf{x}^\star\right)\right\}$, is asymptotically normal if, besides Assumptions **(B1)-(B5)**, Assumptions **(C1)-(C5)** are also satisfied. In particular, as $t\to\infty$, we have: $\sqrt{t}\left(\mathbf{x}^t - \mathbf{x}^\star\right) \xrightarrow{d} \mathbb{N}(0,\mathcal{S})$. Also, the asymptotic covariance $\mathcal{S}$ of the multivariate Gaussian distribution $\mathbb{N}(0,\mathcal{S})$ is $\mathcal{S} = a^2\int_0^\infty e^{v\,\boldsymbol{\Sigma}}\ \mathcal{S}_0 e^{v\,\boldsymbol{\Sigma}^\top}dv$.*

Proof. For a proof see [26] (c.f. Theorems 4.4.4, 6.6.1). □

We also make use of the following Theorem, proved in Appendix A; see also Lemmas 4 and 5 in [21].

THEOREM 5.2. *Let z^t be a nonnegative (deterministic) sequence satisfying:*

$$z^{t+1} \leq \left(1 - r_1^t\right)z^t + r_2^t,$$

for all $t \geq t'$, for some $t' > 0$, with some $z^{t'} \geq 0$. Here, $\{r_1^t\}$ and $\{r_2^t\}$ are deterministic sequences with $\frac{a_1}{(t+1)^{\delta_1}} \leq r_1^t \leq 1$ and $r_2^t \leq \frac{a_2}{(t+1)^{\delta_2}}$, with $a_1, a_2 > 0$, and $\delta_2 > \delta_1 > 0$. Then, the following holds: (1) If $\delta_1 < 1$, then $z^t = O(\frac{1}{t^{\delta_2 - \delta_1}})$; (2) If $\delta_1 = 1$, then $z^t = O(\frac{1}{t^{\delta_2 - 1}})$ provided that $a_1 > \delta_2 - \delta_1$; (3) if $\delta_1 = 1$ and $a_1 \leq \delta_2 - 1$, then $z^t = O(\frac{1}{t^{\zeta}})$, for any $\zeta < a_1$.

5.2. Asymptotic analysis: Proofs of Theorem 3.1 and Theorem 3.3. The next Lemma, due to [30], establishes structural properties of function ϕ in (3.1). The Lemma says that essentially, the convolution-like transofrmation of the nonlinearity preserves the structural properties of the nonlinearity. For a proof of the Lemma, see Appendix D.

LEMMA 5.3. *[30] Consider function ϕ in (3.1), where function $\Psi : \mathbb{R} \mapsto \mathbb{R}$ satisfies Assumption 5, and noise pdf $p : \mathbb{R} \mapsto \mathbb{R}_+$ satisfies Assumption 3. Then, the following holds.*

1. *ϕ is odd;*
2. *If in addition Assumption 7 holds, then $|\phi(a)| \leq K_2$, for any $a \in \mathbb{R}$, for some constant $K_2 > 0$;*
3. *If in addition Assumption 6 holds, then $|\phi(a)| \leq K_1(1 + |a|)$, for any $a \in \mathbb{R}$, for some constant $K_1 > 0$;*
4. *$\phi(a)$ is monotonically nondecreasing;*
5. *If in addition either Assumption 6 or Assumption 7 holds, then ϕ is differentiable at zero, with a strictly positive derivative at zero, equal to:*

$$\phi'(0) = \sum_{i=1}^{s} \left(\Psi(\nu_i + 0) - \Psi(\nu_i - 0)\right) p(\nu_i) + \sum_{i=0}^{s} \int_{\nu_i}^{\nu_{i+1}} \Psi'(\nu) p(\nu) d\nu, \tag{5.4}$$

   where $\nu_i, i = 1, ..., s$ are points of discontinuity of Ψ such that $\nu_0 = -\infty$ and $\nu_{s+1} = +\infty$.

Remark. In view of (5.4), we highlight the need that $p(u)$ is strictly positive in the vicinity of zero and that Ψ is either discontinuous at zero or strictly increasing in the vicinity of zero, in order for $\phi'(0)$ to be strictly positive. (see Assumptions 3 and 5.) Consider the following counterexample: $\Psi(u) = \text{sign}(u)$, where p corresponds to the uniform distribution on the set $(-u_2, -u_1) \cup (u_1, u_2)$, for $0 < u_1 < u_2$. Note that p is zero in the vicinity of zero. Then, by (5.4), $\phi'(0) = 0$.

We proceed by setting up the proof of Theorem 3.1. The proof relies on convergence analysis of single-time scale stochastic approximation methods from [26]; more precisely, we utilize Theorem 5.1 in the Appendix; see also [20].

We first put algorithm (2.3) in the format that complies with Theorem 5.1. Namely, algorithm (2.3) can be written as:

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \alpha_t \left[\mathbf{r}(\mathbf{x}^t) + \boldsymbol{\gamma}(t+1, \mathbf{x}^t, \omega)\right]. \tag{5.5}$$

Here, ω denotes an element of the underlying probability space, and

$$\mathbf{r}(\mathbf{x}) = -\boldsymbol{\phi}(\nabla f(\mathbf{x})), \tag{5.6}$$

where, abusing notation, $\boldsymbol{\phi} : \mathbb{R}^d \mapsto \mathbb{R}^d$ equals $(\boldsymbol{\phi}(a_1, ..., a_d)) = (\phi(a_1), ..., \phi(a_d))^\top$. That is, we have that:

(5.7)

$$\mathbf{r}(\mathbf{x}) = -\left(\phi([\nabla f(x)]_1), ..., \phi([\nabla f(x)]_d)\right)^\top, \;\; \boldsymbol{\gamma}(t+1, \mathbf{x}, \omega) = \boldsymbol{\phi}(\nabla f(\mathbf{x})) - \Psi(\nabla f(\mathbf{x}) + \boldsymbol{\nu}^t).$$

We provide an intuition behind the algorithmic format (5.5). Quantity $\mathbf{r}(\mathbf{x})$ is a deterministic, "useful", progress direction with respect to the evolution of $\mathbf{x}^t$; quantity $\boldsymbol{\gamma}(t+1, x, \omega)$ is the stochastic component that plays a role of a noise in the system.

We adopt the following Lyapunov function: $V(x) = f(x) - f^\star$, $V : \mathbb{R}^d \mapsto \mathbb{R}$, where $f^\star = \inf_{x \in \mathbb{R}^d} f(x) = f(x^\star)$. By Assumptions 1 and 2, V is twice continuously differentiable and has uniformly bounded second order partial derivatives, as required by Theorem 5.1. We are ready to prove Theorem 3.1.

Proof (Proof of Theorem 3.1). We now verify conditions B1-B5 from Theorem 5.1. Recall from Section 3 $\mathcal{F}_t$, the σ-algebra generated with random vectors $\boldsymbol{\nu}^s$, $s = 0, ..., t$. Then, the family of random vectors $\{\boldsymbol{\gamma}\,(t+1, \mathbf{x}, \omega)\}_{\mathbf{x} \in \mathbb{R}^d}$ is $\mathcal{F}_t$-measurable, zero-mean and independent of $\mathcal{F}_{t-1}$. Also, clearly, function $\boldsymbol{\gamma}(t+1, \cdot, \cdot)$ is measurable, for all t. Thus, conditions Bi and B2 hold.

For B3, we need to prove that $\sup_{\mathbf{x}:\|\mathbf{x}-\mathbf{x}^\star\| \in (\epsilon, \frac{1}{\epsilon})} \langle \mathbf{r}(\mathbf{x}), \nabla V(\mathbf{x}) \rangle < 0$, for any $\epsilon > 0$, where $\nabla V(\mathbf{x}) = \nabla f(\mathbf{x})$. Let us fix an $\epsilon > 0$. Then, we have, for any $\mathbf{x} \in \mathbb{R}^d$:

$$\begin{aligned}\langle \mathbf{r}(\mathbf{x}), \nabla V(\mathbf{x}) \rangle &= -\boldsymbol{\phi}(\nabla f(\mathbf{x}))^\top (\nabla f(\mathbf{x})) \\ &= -\sum_{j=1}^d \phi([\nabla f(\mathbf{x})]_j)[\nabla f(\mathbf{x})]_j = -\sum_{j=1}^d |\phi([\nabla f(x)]_j)|\,|[\nabla f(x)]_j|,\end{aligned}$$

where the last equality holds because ϕ is an odd function. Consider arbitrary $\mathbf{x}$ such that $\|\mathbf{x} - \mathbf{x}^\star\| \geq \epsilon$. As $\|\nabla f(\mathbf{x})\|^2 \geq \mu^2 \|\mathbf{x} - \mathbf{x}^\star\|^2$ (due to strong convexity of f), we have $\|\nabla f(\mathbf{x})\| \geq \mu\epsilon$, where we recall that μ is the strong convexity constant of f. Therefore, there exists an index $i \in \{1, ..., d\}$ such that $|[\nabla f(\mathbf{x})]_i| \geq \frac{1}{d}\mu\epsilon =: \epsilon'$. Next, because $\phi'(0) > 0$, and ϕ is continuous at 0 and is non-decreasing (by Lemma 5.3), we have that $|\phi(b)| \geq \delta$ for some $\delta = \delta(\epsilon) > 0$, for all $b \in [\epsilon, 1/\epsilon]$. Finally, we have that: $\langle \mathbf{r}(\mathbf{x}), \nabla V(\mathbf{x}) \rangle \leq -\epsilon'\delta(\epsilon)$ for any $\mathbf{x}$ such that $\|\mathbf{x} - \mathbf{x}^\star\| \in [\epsilon, \frac{1}{\epsilon}]$, and therefore $\sup_{\mathbf{x}:\|\mathbf{x}-\mathbf{x}^\star\| \in (\epsilon, \frac{1}{\epsilon})} \langle \mathbf{r}(x), \nabla V(\mathbf{x}) \rangle \leq \sup_{\mathbf{x}:\|\mathbf{x}-\mathbf{x}^\star\| \in [\epsilon, \frac{1}{\epsilon}]} \langle \mathbf{r}(x), \nabla V(\mathbf{x}) \rangle \leq -\delta(\epsilon)\,\epsilon' < 0$, hence verifying condition B3.

We next verify condition B4. Consider quantity $\mathbf{r}(\mathbf{x})$ in (5.6). By Lemma 5.3 and the fact that f has Lipschitz gradient and is strongly convex (Assumption 1), it follows that: $\|\mathbf{r}(\mathbf{x})\|^2 \leq C_{r,1} + C_{r,2} V(\mathbf{x})$, for some positive constants $C_{r,1}$ and $C_{r,2}$. Also, since $\|\boldsymbol{\gamma}(\mathbf{x}, t+1, \omega)\|^2 \leq 2\|\phi(\nabla f(\mathbf{x}))\|^2 + 2\|\Psi(\nabla f(\mathbf{x}) + \boldsymbol{\nu}^t)\|^2$, and it holds that either 1) Ψ is bounded or 2) $|\Psi(a)| \leq C_2\,(1 + |a|)$ and ν_i^t has a finite variance, we have: $\mathbb{E}\left[\|\boldsymbol{\gamma}(\mathbf{x}, t+1, \omega)\|^2\right] \leq C_3 + C_4\, V(\mathbf{x})$, for some positive constants C_3, C_4. Now, we finally have:

$$\|\mathbf{r}(\mathbf{x})\|^2 + \mathbb{E}\left[\|\boldsymbol{\gamma}(\mathbf{x}, t+1, \omega)\|^2\right] \leq C_5 + C_6\, V(\mathbf{x}),$$

for some positive constants C_5, C_6, and hence condition B4 holds for a constant $k_1 > 0$ and $k_2 = 0$.[6] Condition B5 holds by the choice of the step size sequence $\{\alpha_t\}$ in the Theorem statement. Summarizing, all conditions B1-B5 hold true, and hence $\mathbf{x}^t \to \mathbf{x}^\star$, almost surely. □

We continue by proving Theorem 3.3.

Proof (Proof of Theorem 3.3). We prove the Theorem by verifying conditions C1-C5 in Theorem 5.1. To verify condition C1, consider $\mathbf{r}(\mathbf{x})$ in (5.6) and note that,

[6]Note that the term $-\langle \mathbf{r}(\mathbf{x}), \nabla V(\mathbf{x}) \rangle$ in condition B4 of Theorem 5.1 equals $\langle \boldsymbol{\phi}(\mathbf{x}), \nabla f(\mathbf{x}) \rangle$. This quantity is nonnegative, for any $\mathbf{x} \in \mathbb{R}^d$, and so k_2 can be taken to be any positive number. In other words, setting $k_2 = 0$ in B4 corresponds to a tighter inequality than the corresponding inequality for any $k_2 > 0$.

using the mean value theorem, it can be expressed as follows:

$$\begin{aligned}\mathbf{r}(\mathbf{x}) &= -\phi(\nabla f(\mathbf{x}) - \nabla f(\mathbf{x}^\star))\\ &= -\phi\left(\underbrace{\left[\int_0^1 \nabla^2 f(\mathbf{x}^\star + t(\mathbf{x}-\mathbf{x}^\star)dt\right]}_{\mathbf{H}}(\mathbf{x}-\mathbf{x}^\star)\right) \qquad (5.8)\\ &= -\phi\left(\mathbf{H}(\mathbf{x}-\mathbf{x}^\star)\right) = -\phi'(0)\nabla^2 f(\mathbf{x}^\star)(\mathbf{x}-\mathbf{x}^\star) + \delta(\mathbf{x}),\end{aligned}$$

where $\lim_{\mathbf{x}\to\mathbf{x}^\star} \frac{\|\delta(x)\|}{\|x-x^\star\|} = 0$. Hence, in the notation of Theorem 5.1, we have that $\mathbf{B} = -\phi'(0)\nabla^2 f(\mathbf{x}^\star)$. Therefore, C1 holds. Also, C2 holds, by assumptions of Theorem 3.3. Now, we consider C3, which requires that the matrix $\mathbf{\Sigma} = a\,\mathbf{B} + \frac{1}{2}\mathbf{I}$ is stable (all its eigenvalues have negative real parts), where $\mathbf{B} = -\phi'(0)\nabla^2 f(\mathbf{x}^\star)$. Note that $\mathbf{\Sigma} = \frac{1}{2}\mathbf{I} - a\,\phi'(0)\nabla^2 f(\mathbf{x}^\star)$. Clearly, $\mathbf{\Sigma}$ is stable for large enough a, because the matrix $\phi'(0)\nabla^2 f(\mathbf{x}^\star)$ is positive definite. More precisely, $\mathbf{\Sigma}$ is stable for $a > 1/(2\mu\phi'(0))$. Therefore, condition C3 holds, provided that $a > 1/(2\mu\phi'(0))$. We next consider condition C4. In the notation of Theorem 5.1, consider the following quantity:

$$\begin{aligned}\mathbf{A}(t,\mathbf{x}) &= \mathbb{E}\left[\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)^\top\right]\\ &= \mathbb{E}\left[\left(\phi(\nabla f(\mathbf{x})) - \Psi(\nabla f(\mathbf{x}) + \boldsymbol{\nu}^t)\right)\left((\phi(\nabla f(\mathbf{x})) - \Psi(\nabla f(\mathbf{x}) + \boldsymbol{\nu}^t)\right)^\top\right]\\ &= \mathbb{E}\left[\left(\phi(\nabla f(\mathbf{x})) - \Psi(\nabla f(\mathbf{x}) + \boldsymbol{\nu}^0)\right)\left((\phi(\nabla f(\mathbf{x})) - \Psi(\nabla f(\mathbf{x}) + \boldsymbol{\nu}^0)\right)^\top\right] \qquad (5.9)\\ &= \mathbb{E}\left[\boldsymbol{\gamma}(1,\mathbf{x},\omega)\boldsymbol{\gamma}(1,\mathbf{x},\omega)^\top\right]. \qquad (5.10)\end{aligned}$$

Consider the set $\Omega^\star$ of all outcomes $\omega \in \Omega$ such that $\mathbf{\Psi}$ is continuous at $\nu^0(\omega)$. Clearly, the set $\Omega^\star$ has the probability one. For every $\omega \in \Omega^\star$, we have $\mathbf{\Upsilon}(\omega) := \lim_{t\to\infty,\mathbf{x}\to\mathbf{x}^\star} \boldsymbol{\gamma}(1,\mathbf{x},\omega)\boldsymbol{\gamma}(1,\mathbf{x},\omega)^\top = \Psi(\boldsymbol{\nu}^0)\Psi(\boldsymbol{\nu}^0)^\top$. Note that, for any $\epsilon > 0$, the random family $\|\boldsymbol{\gamma}(1,\mathbf{x},\omega)\boldsymbol{\gamma}(1,\mathbf{x},\omega)^\top\|$, $\|\mathbf{x}-\mathbf{x}^\star\| < \epsilon$ is dominated by an integrable random variable. (See ahead (5.12)–(5.13).) Therefore, by the dominated convergence theorem, and the fact that the entries of $\boldsymbol{\nu}^0$ are mutually independent with pdf $p(u)$, we have that:

$$\lim_{t\to\infty,\mathbf{x}\to\mathbf{x}^\star} \mathbf{A}(t,\mathbf{x}) = \mathbf{A}_0 = \mathbb{E}\left[\Psi(\boldsymbol{\nu}^0)\cdot\Psi(\boldsymbol{\nu}^0)^\top\right] = \sigma_\Psi^2 \cdot \mathbf{I}, \qquad (5.11)$$

where $\sigma_\Psi^2 = \int |\Psi(a)|^2 p(a)da$. Therefore, condition C4 holds. We finally verify condition C5. We follow the arguments analogous to those in Theorem 10 in [20]. Condition C5 means uniform integrability of the family $\{\|\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)\|^2\}_{t=0,1,\dots,\,\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}$. We have: $\|\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)\|^2 \le 2\|\phi(\nabla f(\mathbf{x}))\|^2 + 2\|\psi(\nabla f(\mathbf{x}) + \boldsymbol{\nu}^t)\|^2$. First, consider the case when Assumptions 6 and 4 hold. Then:

$$\begin{aligned}\|\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)\|^2 &\le C_7 + C_8\|\mathbf{x}-\mathbf{x}^\star\|^2 + C_9\|\boldsymbol{\nu}^t\|^2\\ &\le C_7 + C_8\,\epsilon^2 + C_9\|\boldsymbol{\nu}^t\|^2, \qquad (5.12)\end{aligned}$$

for some positive constants C_7, C_8, C_9. Consider next the family $\{\widetilde{\gamma}(t+1,\mathbf{x},\omega)\}_{t=0,1,\dots,\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}$, with $\widetilde{\gamma}(t+1,\mathbf{x},\omega) = C_7 + C_8\,\epsilon^2 + \mathrm{C}_9\|\boldsymbol{\nu}^t\|^2$. The family $\{\widetilde{\gamma}(t+1,\mathbf{x},\omega)\}_{t=0,1,\dots,\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}$ is i.i.d. and hence it is uniformly integrable. The family $\{\|\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)\|^2\}_{t=0,1,\dots,\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}$ is dominated by $\{\widetilde{\gamma}(t+1,\mathbf{x},\omega)\}_{t=0,1,\dots,\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}$ that is uniformly integrable, and hence $\{\|\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)\|^2\}_{t=0,1,\dots,\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}$ is also uniformly integrable. Hence, C5 holds.

Now, let Assumption 7 hold. Then:

$$\|\boldsymbol{\gamma}(t+1,\mathbf{x},\omega)\|^2 \le C_{10} + C_{11}\|\mathbf{x}-\mathbf{x}^\star\|^2 \le C_{10} + C_{11}\,\epsilon^2. \tag{5.13}$$

Consider the family $\{\widehat{\gamma}(t+1,\mathbf{x},\omega)\}_{t=0,1,\dots,\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}$, with $\widehat{\gamma}(t+1,\mathbf{x},\omega) = C_{10} + C_{11}\,\epsilon^2$. The family $\{\widehat{\gamma}(t+1,\mathbf{x},\omega)\}_{t=0,1,\dots,\|\mathbf{x}-\mathbf{x}^\star\|<\epsilon}$ is uniformly integrable, and condition C5 is verified analogously to the previous case. Summarizing, we have established that all conditions C1-C5 of Theorem 5.1 hold true, thus the proof of Theorem 3.3 □.

5.3. MSE analysis: Proof of Theorem 3.2. We start with the following Lemma that upper boounds $\|\nabla f(\mathbf{x}^t)\|$.

LEMMA 5.4. *Let Assumptions 1, 3, 5, and 7 hold. Further, let the step-size sequence $\{\alpha_t\}$ be $\alpha_t = a/(t+1)^\delta$, $a>0$, $\delta \in (0.5,1)$. Then, for each $t=1,2,\dots$, we have, a.s.:*

$$\|\nabla f(\mathbf{x}^t)\| \le G_t := L\left(a\,C_2\sqrt{d}\,\frac{t^{1-\delta}}{1-\delta} + \|\mathbf{x}^0-\mathbf{x}^\star\|\right). \tag{5.14}$$

Proof. Consider (2.3). Because the output of each component nonlinearity Ψ is bounded in the absolute value by C_2 (Assumption 7), we have, for each $t \ge 1$:

$$\begin{aligned}\|\mathbf{x}^t-\mathbf{x}^\star\| &\le \|\mathbf{x}^0-\mathbf{x}^\star\| + a\sqrt{d}\,C_2\sum_{s=0}^{t-1}\frac{1}{(s+1)^\delta}\\ &\le \|\mathbf{x}^0-\mathbf{x}^\star\| + a\,C_2\sqrt{d}\left(\frac{t^{1-\delta}}{1-\delta}\right).\end{aligned} \tag{5.15}$$

Next, because ∇f is L-Lipschitz, we have $\|\nabla f(\mathbf{x}^t)\| \le L\,\|\mathbf{x}^t-\mathbf{x}^\star\|$. Applying this inequality to (5.15), the result follows. □

We will also make use of the following Lemma.

LEMMA 5.5. *There exists a positive constant ξ such that, for any $t=1,2,\dots$, there holds, almost surely, for each $j=1,\dots,d$, that: $|\phi([\nabla f(\mathbf{x}^t)]_j)| \ge |[\nabla f(\mathbf{x}^t)]_j|\;\frac{\phi'(0)\,\xi}{2\,G_t}$, where G_t is defined in (5.14).*

Proof. Consider function ϕ in (2.1). By Lemma 5.3, we have that $\phi'(0)>0$ and ϕ is continuous at zero[7]. Because ϕ is differentiable at zero, using first order Taylor series, there holds: $\phi(u) = \phi(0) + \phi'(0)u + h(u)u = \phi'(0)u + h(u)u$, $u \in \mathbb{R}$, where $h:\mathbb{R}\mapsto\mathbb{R}$ is a function such that $\lim_{u\to 0} h(u)=0$. Due to the latter property of h, there exists a positive number ξ such that $|h(u)| \le \frac{\phi'(0)}{2}$, for all $u\in[0,\xi)$. Using the latter bound, we obtain that $\phi(u) \ge \frac{1}{2}\phi'(0)u$, $u\in[0,\xi)$. Now, because ϕ is non-decreasing (by Lemma 5.3), it holds for any $a'>\xi$ that $\phi(a) \ge \frac{\phi'(0)\,\xi\,a}{2\,a'}$, for any $a\in[0,a')$. Consider now $\nabla f(\mathbf{x}^t)$. By Lemma 5.4, we have that $\|\nabla f(\mathbf{x}^t)\| \le G_t$, a.s., and so, for any $j=1,\dots,d$, $|[\nabla f(\mathbf{x}^t)]_j| \le G_t$. Therefore, setting $a'=G_t$, the Lemma follows. □

We are now ready to prove Theorem 3.2.

Proof (Proof of Theorem 3.2). Consider algorithm (2.3) under Assumptions 1, 3, 5, and 7. By the Lipschitz property of ∇f, we have, for any $\mathbf{x},\mathbf{y}\in\mathbb{R}^d$, that:

$$f(\mathbf{y}) \le f(\mathbf{x}) + \nabla f(\mathbf{x})^\top(\mathbf{y}-\mathbf{x}) + \frac{L}{2}\|\mathbf{x}-\mathbf{y}\|^2,$$

[7]As ϕ is an odd function, for simplicity, in the proof we consider only nonnegative arguments of ϕ, while analogous analysis applies for negative arguments of ϕ.

and so, almost surely:

$$
\begin{aligned}
f(\mathbf{x}^{t+1}) &\le f(\mathbf{x}^t) + \big(\nabla f(\mathbf{x}^t)\big)^\top (-\alpha_t \Psi(\nabla f(\mathbf{x}^t) + \boldsymbol{\nu}^t)) \\
&\quad + \frac{L}{2}\alpha_t^2 \|\Psi(\nabla f(\mathbf{x}^t) + \boldsymbol{\nu}^t)\|^2.
\end{aligned} \tag{5.16}
$$

Next, letting $\boldsymbol{\eta}^t = \Psi(\nabla f(\mathbf{x}^t) + \boldsymbol{\nu}^t) - \boldsymbol{\phi}(\nabla f(\mathbf{x}^t))$, and using the fact that Ψ has bounded outputs, we obtain:

$$
\begin{aligned}
f(\mathbf{x}^{t+1}) &\le f(\mathbf{x}^t) + \big(\nabla f(\mathbf{x}^t)\big)^\top (-\alpha_t \boldsymbol{\phi}(\nabla f(\mathbf{x}^t))) \\
&\quad + \frac{L}{2}\alpha_t^2\, d^2 C_2^2 - \alpha_t\, (\nabla f(\mathbf{x}^t))^\top \boldsymbol{\eta}^t, \text{ a.s.}
\end{aligned} \tag{5.17}
$$

Recall filtration $\mathcal{F}_t$. Taking conditional expectation, and using that $\mathbb{E}[\boldsymbol{\eta}^t \,|\, \mathcal{F}_t] = 0$, we get that, almost surely:

$$
\mathbb{E}[f(\mathbf{x}^{t+1}) \,|\, \mathcal{F}_t] \le f(\mathbf{x}^t) - \alpha_t \big(\nabla f(\mathbf{x}^t)\big)^\top \boldsymbol{\phi}(\nabla f(\mathbf{x}^t)) + \frac{L}{2}\alpha_t^2\, d^2 C_2^2. \tag{5.18}
$$

Next, using Lemma 5.5, and the fact that $\alpha_t = a/(t+1)^\delta$, we obtain that, a.s.:

$$
\mathbb{E}[f(\mathbf{x}^{t+1}) \,|\, \mathcal{F}_t] \le f(x^t) - \frac{c'}{(t+1)}\|\nabla f(\mathbf{x}^t)\|^2 + \frac{L}{2}\frac{a^2\, d^2\, C_2^2}{(t+1)^{2\delta}}, \tag{5.19}
$$

where $c' = \frac{a\,(1-\delta)\xi\,\phi'(0)}{2\,L\,(a\,C_2\,\sqrt{d} + \|\mathbf{x}^0 - \mathbf{x}^\star\|)}$. Next, by strong convexity of f, we have that $\|\nabla f(\mathbf{x}^t) - \nabla f(\mathbf{x}^\star)\|^2 \ge 2\,\mu\,(f(\mathbf{x}^t) - f^\star)$. Using the latter inequality, subtracting $f^\star$ from both sides of the inequality, taking expectation, and applying Theorem 5.2, claims (2) and (3), we obtain the desired MSE rate result.

We next consider the case when Assumption 7 is replaced with Assumption 6 and Assumption 4 is additionally imposed. Following analogous arguments as in the first part of the proof, it can be shown that, a.s.:

$$
\begin{aligned}
\mathbb{E}[f(\mathbf{x}^{t+1}) \,|\, \mathcal{F}_t] &\le f(\mathbf{x}^t) - \alpha_t\, \boldsymbol{\phi}(\nabla f(\mathbf{x}^t))^\top \nabla f(\mathbf{x}^t) \\
&\quad + \frac{L}{2}\alpha_t^2 \left(C_{13} + C_{14}\mathbb{E}[\,\|\boldsymbol{\nu}^t\|^2 \,|\, \mathcal{F}_t\,]\right),
\end{aligned} \tag{5.20}
$$

for some positive constants C_{13}, C_{14}. Next, because $\inf_{a \neq 0} \frac{|\phi(a)|}{|a|} > 0$, we have that $\phi(\nabla f(\mathbf{x}^t))^\top \nabla f(\mathbf{x}^t) \ge C_{15}\,\|\nabla f(\mathbf{x}^t)\|^2$, for some constant $C_{15} > 0$. Using the latter bound in (5.20), subtracting $f^\star$ from both sides of the inequality, taking expectation, and applying Theorem 5.2, claim (1) and (2), the result follows. □

6. Intermediate results and proofs: Joint nonlinearities.

Subsection 6.1 provides the required intermediate results, while Subsection 6.2 proves Theorem 4.1.

6.1. Intermediate results: Joint nonlinearities.

Recall function $\mathcal{N} : \mathbb{R}_+ \mapsto \mathbb{R}_+$ in Assumption 9. We first state and prove the following Lemma on the properties of function $\mathcal{N}$.

LEMMA 6.1. *Under Assumption 9, for any* $\mathbf{x}, \mathbf{u} \in \mathbb{R}^d$, *such that* $\|\mathbf{u}\| > \|\mathbf{x}\|$, *there holds:*

$$
\begin{aligned}
&|\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) - \mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)| \le \\
&\qquad \frac{\|\mathbf{x}\|}{\|\mathbf{u}\|}\left[\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) + \mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)\right].
\end{aligned} \tag{6.1}
$$

Proof. Fix a pair $\mathbf{x}, \mathbf{u} \in \mathbb{R}^d$, such that $\|\mathbf{u}\| > \|\mathbf{x}\|$, and assume without loss of generality that $\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) \geq \mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)$. Then, (6.1) is equivalent to:

$$(\|\mathbf{u}\| - \|\mathbf{x}\|)\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) \leq (\|\mathbf{u}\| + \|\mathbf{x}\|)\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|). \tag{6.2}$$

Denote by $\rho = \|\mathbf{u}\|$. Notice that: $\rho - \|\mathbf{x}\| \leq \|\mathbf{x}+\mathbf{u}\| \leq \|\mathbf{x}\| + \|\mathbf{u}\| = \|\mathbf{x}\| + \rho$, and similarly, $\rho + \|\mathbf{x}\| \geq \|\mathbf{x}-\mathbf{u}\| \geq \rho - \|\mathbf{x}\|$. As $\mathcal{N}$ is non-increasing, it follows that: $\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) \leq \mathcal{N}(\rho - \|\mathbf{x}\|)$, and $\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|) \geq \mathcal{N}(\rho + \|\mathbf{x}\|)$. Now, we have:

$$(\|\mathbf{u}\| - \|\mathbf{x}\|)\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) \leq (\rho - \|\mathbf{x}\|)\mathcal{N}(\rho - \|\mathbf{x}\|), \tag{6.3}$$

and similarly:

$$(\|\mathbf{u}\| + \|\mathbf{x}\|)\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|) \geq (\rho + \|\mathbf{x}\|)\mathcal{N}(\rho + \|\mathbf{x}\|). \tag{6.4}$$

By assumption, function $a \mapsto a\mathcal{N}(a)$, $a > 0$, is non-decreasing, and so $(\rho - \|\mathbf{x}\|)\mathcal{N}(\rho - \|\mathbf{x}\|) \leq (\rho + \|\mathbf{x}\|)\mathcal{N}(\|\mathbf{x}\| + \rho)$. Thus, combining (6.3) and (6.4), we have that (6.2) holds, which is in turn equivalent to the claim of the Lemma. ∎

We now define map $\boldsymbol{\phi} : \mathbb{R}^d \mapsto \mathbb{R}^d$, as follows. For a fixed (deterministic) point $\mathbf{w} \in \mathbb{R}^d$, we let:

$$\boldsymbol{\phi}(\mathbf{w}) = \int \boldsymbol{\Psi}(\mathbf{w}+\mathbf{u})p(\mathbf{u})d\mathbf{u} = \mathbb{E}[\boldsymbol{\Psi}(\mathbf{w}+\boldsymbol{\nu}^0)] \tag{6.5}$$

where the expectation is taken with respect to the joint pdf of the gradient noise at any iteration t, e.g., $t = 0$. The map $\boldsymbol{\phi} : \mathbb{R}^d \mapsto \mathbb{R}^d$ is abuse of notation, a counterpart of the component-wise map $\phi : \mathbb{R} \mapsto \mathbb{R}$ in ([illegible]). We have the following Lemma.

Lemma 6.2. *Under Assumptions 8 and 9, the following holds:*

$$\boldsymbol{\phi}(\mathbf{x})^\top \mathbf{x} \geq 2(1-\kappa)\|\mathbf{x}\|^2 \int_{\mathcal{J}(\mathbf{x})} \mathcal{N}(\|\mathbf{x}\| + \|\mathbf{u}\|)p(\mathbf{u})d\mathbf{u}, \tag{6.6}$$

where $\mathcal{J}(\mathbf{x}) = \{\mathbf{u} : \frac{\mathbf{u}^\top \mathbf{x}}{\|\mathbf{u}\|\|\mathbf{x}\|} \in [0, \kappa]\}$, *and* κ *is any constant in the interval* $(0,1)$.

Proof. Let us fix arbitrary $\mathbf{x} \in \mathbb{R}^d, \mathbf{x} \neq 0$. As $\boldsymbol{\Psi}(\mathbf{a}) = \mathbf{a}\mathcal{N}(\|\mathbf{a}\|)$, we have:

(6.7)

$$\boldsymbol{\phi}(\mathbf{x})^\top \mathbf{x} = \int_{\mathbf{u}\in\mathbb{R}^d} \underbrace{(\mathbf{x}+\mathbf{u})^\top \mathbf{x}\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)}_{:=\mathcal{M}(\mathbf{x},\mathbf{u})} p(\mathbf{u})d\mathbf{u}$$

$$= \int_{J_1(\mathbf{x})=\{\mathbf{u}:\,\mathbf{u}^\top\mathbf{x}\geq 0\}} \mathcal{M}(\mathbf{x},\mathbf{u})p(\mathbf{u})d\mathbf{u} + \int_{J_2(\mathbf{x})=\{\mathbf{u}:\,\mathbf{u}^\top\mathbf{x}<0\}} \mathcal{M}(\mathbf{x},\mathbf{u})p(\mathbf{u})d\mathbf{u}. \tag{6.8}$$

Note also that there holds: $\mathcal{M}(\mathbf{x},\mathbf{u}) = (\|\mathbf{x}\|^2 + \mathbf{u}^\top\mathbf{x})\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)$; and $\mathcal{M}(\mathbf{x},-\mathbf{u}) = (\|\mathbf{x}\|^2 - \mathbf{u}^\top\mathbf{x})\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)$. Therefore, using the fact that $p(\mathbf{u}) = p(-\mathbf{u})$, for all $u \in \mathbb{R}^d$, we obtain: $\boldsymbol{\phi}(\mathbf{x})^\top\mathbf{x} = \int_{J_1(\mathbf{x})} \mathcal{M}_2(\mathbf{x},\mathbf{u})\, p(\mathbf{u})d\mathbf{u}$, where $\mathcal{M}_2(\mathbf{x},\mathbf{u}) = [(\|\mathbf{x}\|^2 + \mathbf{u}^\top\mathbf{x})\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) + (\|\mathbf{x}\|^2 - \mathbf{u}^\top\mathbf{x})\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)]$. There holds:

$$\begin{aligned}\mathcal{M}_2(\mathbf{x},\mathbf{u}) \geq\ & \|\mathbf{x}\|^2[\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) + \mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)] - \\ & - \|\mathbf{u}\|\|\mathbf{x}\||\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|) - \mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)|.\end{aligned} \tag{6.9}$$

Since $\mathbf{u} \in J_1(\mathbf{x})$, there holds $\|\mathbf{x}+\mathbf{u}\| \geq \|\mathbf{x}-\mathbf{u}\|$. Now, using Lemma 6.1, we have:

$$\begin{aligned} \mathcal{M}_2(\mathbf{x},\mathbf{u}) &\geq \|\mathbf{x}\|^2[\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)+\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)]- \\ &\qquad \|\mathbf{u}\|\|\mathbf{x}\|\frac{\|\mathbf{x}\|}{\|\mathbf{u}\|}|\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)+\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)| = 0. \end{aligned} \tag{6.10}$$

Therefore, we have: $\mathcal{M}_2(\mathbf{x},\mathbf{u}) \geq 0$, for any $\mathbf{u} \in J_1(\mathbf{x})$, $\|\mathbf{u}\| > \|\mathbf{x}\|$. Now, consider $\mathcal{J}(\mathbf{x}) = \{\mathbf{u} \in \mathbb{R}^d : \mathbf{u}^\top\mathbf{x} \geq 0, \frac{\mathbf{u}^\top\mathbf{x}}{\|\mathbf{u}\|\|\mathbf{x}\|} \in [0,\kappa]\}$, where $\kappa \in (0,1)$. Let us consider $\mathbf{u} \in \mathcal{J}(\mathbf{x})$ such that $\|\mathbf{u}\| > \|\mathbf{x}\|$. Then, using Lemma 6.1, we get:

$$\begin{aligned} \mathcal{M}_2(\mathbf{x},\mathbf{u}) &\geq \|\mathbf{x}\|^2[\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)+\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)] \\ &\quad - \|\mathbf{u}\|\|\mathbf{x}\|\kappa\,\underbrace{|\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)-\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)|} \\ &\geq (1-\kappa)\|\mathbf{x}\|^2(\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)+\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)). \end{aligned} \tag{6.11}$$

Now, consider $\mathbf{u} \in \mathcal{J}(\mathbf{x})$ such that $\|\mathbf{u}\| \leq \|\mathbf{x}\|$. Then, there holds:

$$\begin{aligned} \mathcal{M}_2(\mathbf{x},\mathbf{u}) &\geq \|\mathbf{x}\|^2[\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)+\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)]- \\ &\qquad \underbrace{\|\mathbf{u}\|}_{\leq\|\mathbf{x}\|}\,\|\mathbf{x}\|\kappa|\underbrace{\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)}_{\geq 0}+\underbrace{\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)}_{\geq 0}| \\ &\geq (1-\kappa)\|\mathbf{x}\|^2(\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)+\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)), \end{aligned} \tag{6.12}$$

where the last inequality holds due to the fact that $|a-b| \leq |a|+|b|$, for any $a,b \in \mathbb{R}$. Now, we have:

$$\begin{aligned} \mathcal{M}_2(\mathbf{x},\mathbf{u}) &\geq (1-\kappa)\|\mathbf{x}\|^2(\underbrace{\mathcal{N}(\|\mathbf{x}+\mathbf{u}\|)}_{\geq\mathcal{N}(\|x\|+\|u\|)}+\underbrace{\mathcal{N}(\|\mathbf{x}-\mathbf{u}\|)}_{\geq\mathcal{N}(\|\mathbf{x}\|+\|\mathbf{u}\|)}) \\ &\geq 2(1-\kappa)\|\mathbf{x}\|^2\mathcal{N}(\|\mathbf{x}\|+\|\mathbf{u}\|), \text{ for any } \mathbf{u} \in \mathcal{J}(\mathbf{x}). \end{aligned} \tag{6.13}$$

From (6.13), we finally get:

$$\begin{aligned} \boldsymbol{\phi}(\mathbf{x})^\top\mathbf{x} &\geq \int\limits_{\mathcal{J}(\mathbf{x})} 2(1-\kappa)\|\mathbf{x}\|^2\mathcal{N}(\|\mathbf{x}\|+\|\mathbf{u}\|)p(\mathbf{u})d\mathbf{u} \\ &= 2(1-\kappa)\|\mathbf{x}\|^2 \int\limits_{\mathcal{J}(\mathbf{x})} \mathcal{N}(\|\mathbf{x}\|+\|\mathbf{u}\|)p(\mathbf{u})d\mathbf{u}. \end{aligned} \tag{6.14}$$ ∎

LEMMA 6.3. *Let Assumptions 1, 8, and Assumption 9 with condition 3. hold (the nonlinearity with bounded outputs case). Then, for each $t = 1, 2, ...,$ we have:*

$$\|\nabla f(\mathbf{x}^t)\| \leq G'_t := L\,(\,a\,C'_2\,\frac{t^{1-\delta}}{1-\delta} + \|\mathbf{x}^0 - \mathbf{x}^\star\|\,). \tag{6.15}$$

Proof. The proof is analogous to the proof of Lemma 5.4. ∎

6.2. Proofs of Theorems 4.1, 4.2, and 4.3: Joint nonlinearities. We are now ready to prove the results for the joint nonlinearities case.

Proof (Proof of Theorem 4.1) We carry out the proof again by verifying conditions B1-B5 in Theorem 5.1. Algorithm (2.3) admits again the representation in Theorem 5.1 with

$$\mathbf{r}(\mathbf{x}) = -\boldsymbol{\phi}(\nabla f(\mathbf{x})),\ \boldsymbol{\gamma}(t+1,\mathbf{x},\omega) = \boldsymbol{\phi}(\nabla f(\mathbf{x})) - \boldsymbol{\Psi}(\nabla f(\mathbf{x}) + \boldsymbol{\nu}^t). \tag{6.16}$$

Conditions B1 and B2 hold analogously to the proof of Theorem 3.1. Condition B3 follows from Lemma 6.2. Condition B4 holds analogously to the proof of Theorem 3.1. Finally, condition B5 follows from the definition of the step-size sequence in Theorem 4.1. Thus, the result. □ We next prove Theorem 4.3. *Proof* (Proof of Theorem 4.3) We carry out the proof again by verifying conditions C1–C5 in Equation (8.2). The conditions C2–C5 are verified analogously as in the proof of Theorem 3.3. For condition C1, first fix an arbitrary $\mathbf{u} \neq 0$, and consider points $\mathbf{x}$ in the vicinity of $\mathbf{x}^\star$. Then, using the differentiability of $\mathcal{N}(a)$ for $a \neq 0$ and the differentiability of ∇f, it can be shown that:

$$\begin{aligned}\boldsymbol{\Psi}(\mathbf{u}+\nabla f(\mathbf{x})) &= \mathbf{u}\mathcal{N}(\|\mathbf{u}\|) + \mathcal{N}(\|\mathbf{u}\|)\nabla^2 f(\mathbf{x}^\star)(\mathbf{x}-\mathbf{x}^\star)\\ &+ \mathcal{N}'(\|\mathbf{u}\|)\frac{\mathbf{u}\mathbf{u}^\top}{\|\mathbf{u}\|}\nabla^2 f(\mathbf{x}^\star)(\mathbf{x}-\mathbf{x}^\star) + o(\|\mathbf{x}-\mathbf{x}^\star\|).\end{aligned}$$

We next integrate the above equality with respect to the joint pdf $p(\mathbf{u})$. For the first term above, note that $\int \mathcal{N}(\|\mathbf{u}\|)\mathbf{u}p(\mathbf{u})d\mathbf{u} = 0$, because $p(\mathbf{u}) = p(-\mathbf{u})$, for all $\mathbf{u}$. The second term is integrable as $\sup_{a>0}\mathcal{N}(a) < \infty$ (Assumption 13). The third term is integrable as function $a \mapsto a\mathcal{N}(a)$ is by assumptions non-decreasing, then, by taking its derivative, it follows that $|\mathcal{N}'(a)| \leq \mathcal{N}(a)/a$, $a > 0$, and so $\|\mathbf{u}\mathbf{u}^\top\mathcal{N}'(\|\mathbf{u}\|)\|/\|\mathbf{u}\| \leq \mathcal{N}(\|\mathbf{u}\|)$. Now, using the definition of $\mathbf{r}(\mathbf{x})$, it follows that $\mathbf{r}(\mathbf{x})$ admits the representation (5.2), with:

$$\mathbf{B} = -\left(\int \mathcal{N}(\|\mathbf{u}\|)p(\mathbf{u})d\mathbf{u} + \int_{\mathbf{u}\neq 0}\frac{\mathbf{u}\mathbf{u}^\top}{\|\mathbf{u}\|}\mathcal{N}'(\|\mathbf{u}\|)p(\mathbf{u})d\mathbf{u}\right)\nabla^2 f(\mathbf{x}^\star).$$

The conditions C1–C5 hold; thus, the result. □

We are now ready to prove Theorem 4.2.

Proof (Proof of Theorem 4.2) We first consider the case when Assumptions 1, 8, 9, and 10 hold. Analogously to the proof of 3.2, it can be shown that, a.s.:

$$\mathbb{E}[f(\mathbf{x}^{t+1})|\mathcal{F}_t] \leq f(\mathbf{x}^t) - \alpha_t\,\boldsymbol{\phi}(\nabla f(\mathbf{x}^t))^\top\nabla f(\mathbf{x}^t) + \alpha_t^2\,C_{17}, \tag{6.17}$$

for some positive constant C_{17}. By Lemma 6.2, there holds, for $\mathbf{a} := \nabla f(\mathbf{x}^t)$, a.s.:

$$\left(\boldsymbol{\phi}(\mathbf{a})\right)^\top\mathbf{a} \geq 2(1-\kappa)\|\mathbf{a}\|^2\int_{\mathcal{J}}\mathcal{N}(\|\mathbf{a}\|+\|\mathbf{u}\|)p(\mathbf{u})d\mathbf{u}, \tag{6.18}$$

where we recall $\mathcal{J} = \{\mathbf{u} : \frac{\mathbf{u}^\top\mathbf{a}}{\|\mathbf{u}\|\|\mathbf{a}\|} \in [0,\kappa]\}$, and $\kappa \in (0,1)$ is a constant. Note that, as $a \mapsto a\mathcal{N}(a)$ is non-decreasing, $\mathcal{N}$ satisfies: $\mathcal{N}(b) \geq \min\left(\frac{\mathcal{N}(1)}{b}, \mathcal{N}(1)\right)$ for any $b > 0$. Consider constant B_0 in condition 2. of Assumption 8. Then, for all $\mathbf{u}$ such that $\|\mathbf{u}\| \leq B_0$, there holds $\mathcal{N}(\|\mathbf{a}\|+\|\mathbf{u}\|) \geq \min\left\{\frac{\mathcal{N}(1)}{\|\mathbf{a}\|+B_0}, \mathcal{N}(1)\right\}$. We now have, a.s.:

$$\|\nabla f(\mathbf{x}^t)\|^2\int_{\mathcal{J}}\mathcal{N}\left(\|\nabla f(\mathbf{x}^t)\|+\|\mathbf{u}\|\right)p(\mathbf{u})d\mathbf{u} \tag{6.19}$$

$$\geq \|\nabla f(\mathbf{x}^t)\|^2\int_{\mathcal{J}_4}\min\left\{\frac{\mathcal{N}(1)}{B_0+\|\nabla f(\mathbf{x}^t)\|}, \mathcal{N}(1)\right\}p(\mathbf{u})d\mathbf{u} \tag{6.20}$$

$$\geq \|\nabla f(\mathbf{x}^t)\|^2\frac{\mathcal{N}(1)}{B_0+G_t'}\int_{\mathcal{J}_4}p(\mathbf{u})d\mathbf{u}. \tag{6.21}$$

Here, $J_4 = \{u \in \mathbb{R}^d : \frac{\mathbf{u}^\top \nabla f(\mathbf{x}^t)}{\|\mathbf{u}\|\|\nabla f(\mathbf{x}^t)\|} \in [0, \kappa],\ \|\mathbf{u}\| \le B_0\}$. In (6.20), we used the fact that $\mathcal{N}(a)$ is non-negative for any $a \ge 0$, and in (6.21), we used Lemma 6.3.

Therefore, we have that, almost surely, for sufficiently large t:

$$\|\nabla f(\mathbf{x}^t)\|^2 \int_J \mathcal{N}(\|\nabla f(\mathbf{x}^t)\| + \|\mathbf{u}\|) p(\mathbf{u}) d\mathbf{u} \ge C_{18} \frac{\|\nabla f(\mathbf{x}^t)\|^2}{G_t' + B_0},$$

for some positive constant C_{18}.

Combining the last bound with Lemmas 6.2 and 6.3, in view of condition 2. in Assumption 8, we obtain that, for sufficiently large t, a.s.:

$$(\phi(\nabla f(\mathbf{x}^t)))^\top \nabla f(\mathbf{x}^t) \ge C_{19} \frac{\|\nabla f(\mathbf{x}^t)\|^2}{B_0 + G_t'}, \tag{6.22}$$

where the positive constant C_{19} can be taken as $C_{19} = 2(1-\kappa)\lambda(\kappa)\mathcal{N}(1)$. Applying the bound (6.22) to (6.17) we obtain an equivalent to (5.19). Therein, c' in (5.19) is replaced with a positive constant c'' that can be taken as $c'' = \frac{4\,a\,(1-\kappa)\lambda(\kappa)(1-\delta)\mathcal{N}(1)}{L\,(\,a\,C_2' + \|\mathbf{x}^0 - \mathbf{x}^\star\|\,) + B_0)}$. We now proceed analogously to the proof of Theorem 3.2, by applying claims (2) and (3) of Theorem 5.2. The desired MSE result now follows, with the rate ζ being any positive number less than

$$\min\left\{2\delta - 1,\ \frac{4\,a\,\mu\,(1-\kappa)\lambda(\kappa)(1-\delta)\mathcal{N}(1)}{L\,(\,a\,C_2' + \|\mathbf{x}^0\| + \|\mathbf{x}^\star\|\,) + B_0}\right\}. \tag{6.23}$$

We now consider the case when Assumptions [illegible] 8, [illegible] 11, and 12 hold. We have, by assumption, that $\inf_{\mathbf{x}\neq 0} \frac{\|\Psi(\mathbf{x})\|}{\|\mathbf{x}\|} > 0$. This is equivalent to saying that $\mathcal{N}$ is lower-bounded by a positive constant, i.e., $\mathcal{N}(a) \ge C_{20}$, for each a, for some constant $C_{20} > 0$. Then, it follows that, a.s.:

$$(\phi(\nabla f(\mathbf{x}^t)))^\top \nabla f(\mathbf{x}^t) \ge C_{21} \|\nabla f(\mathbf{x}^t)\|^2, \tag{6.24}$$

for some positive constant C_{21}. The proof then proceeds analogously to the proof of Theorem 3.2 by applying the appropriate variant of Theorem 5.2. ☐

7. Experiments. In order to benchmark the proposed nonlinear SGD framework, we consider `Heart`, `Diabetes` and `Australian` datasets from the LibSVM library [9]. We consider the logistic regression loss function for binary classification, see, e.g., [15], where function f in (2.1) is the empirical loss, i.e., the sum of the logistic losses across all data points in a given dataset.

As it has been studied in [15] (see Figure 2 in [15]), we have, near the solution $\mathbf{x}^\star$, the following behavior with respect to gradient noise. (See also [15] for details how the gradient noise is evaluated in Figure 2 therein.) With the HEART dataset, tails of stochastic gradients are not heavy. On the other hand, for DIABETES and AUSTRALIAN datasets, the gradient noise has outliers and exhibits a heavy-tail behavior.

We consider three different nonlinearities to demonstrate the effectiveness of our nonlinear framework, namely, `tanh` (hyperbolic tangent), `sign` and a bi-level customization of `sign` with $\Psi(x) = -1, -0.5, 0.5, 1$, for $x \in (-\infty, -0.5], (-0.5, 0], (0, 0.5], (0.5, \infty]$, respectively (`nonlinear-quantizer` in figures). Note that the `tanh` function may be considered a smooth approximation of `sign`. We benchmark the above methods against the linear SGD, clipped-SGD and SSTM along with a clipped version of SSTM from [15]. For each of the methods, we use batch sizes of 50, 100 and 20 for the `Australian`, `Diabetes` and `Heart` datasets, respectively. We also consider clipped-SGD with periodically decreasing clipping level (`d-clipped-SGD` in Figures) as a baseline as introduced in [15]. This method starts with some initial clipping level and after every l epochs the clipping level is multiplied by some constant $c \in (0,1)$. The step sizes α_t (learning rates) for each method from our framework were tuned after an experimentation. The learning rates for the baselines, i.e., SGD, clipped-SGD, SSTM and clipped-SSTM are also tuned and are selected to be as in [15]. In more detail, the learning rates for the proposed methods are of the form $a/(b\,(t+1)+L)$, where we recall that t is the iteration counter, L is the smoothness constant of ∇f, and parameters a, b are tuned via grid search. The value of a is chosen to be 1.0, 1.5 and 5.0, respectively, for `Heart`, `Diabetes` and `Australian` and for all the three non-linearities. The value of b is chosen to be 0.001, 7.0 and 7.0 respectively for `Australian`, `Heart` and `Diabetes` datasets for the `sign` nonlinearity. The value of b is chosen to be 0.0001, 2.0 and 3.0×10^{-6} respectively for `Australian`, `Heart` and `Diabetes` datasets for the `tanh` nonlinearity. The value of b is chosen to be 0.001, 5.0 and 5.0 respectively for `Australian`, `Heart` and `Diabetes` datasets for the `nonlinear-quantizer` nonlinearity.

We first note that (see Figure 4.1) `d-clipped-SGD` stabilizes the trajectory as compared to the linear SGD, even if the initial clipping level was high. At the same time, clipped-SGD with large clipping levels performs similarly as SGD. It is noteworthy, that SGD has the least oscillations for `Australian` and `Diabetes` datasets, despite the fact that these datasets have heavier or similar tails. This can be attributed to the fact that SGD does not get close to the solution in terms of functional value. SSTM in particular shows large oscillations, which can be attributed to it being a version of accelerated/momentum-based methods and the usage of small batch sizes. `Clipped-SSTM` on the other hand suffers less from oscillations and has a comparable convergence rate as SSTM. In comparison, all the three nonlinear schemes that have been proposed in this paper, have very little oscillations. While the `tanh` algorithm is outperformed by the algorithms with other nonlinearities from our framework, its performance is at par with the other baselines from [15]. In particular, the `sign` algorithm compares favorably to other baselines in terms of convergence for `Australian` and `Heart` datasets. The `nonlinear-quantizer` algorithm outperforms other baselines for the `Diabetes` dataset. The good behavior of `tanh` and `sign` on the heavy-tail data sets, specially relative to the linear SGD, also viewing `tanh` as a smooth approximation of `sign`, might also be related with the insights from Example 3.4. In summary, the three simple example nonlinearities from the proposed framework are comparable or favorable over the considered state-of-the-art benchmarks on the studied datasets.

8. Conclusion. We proposed a general framework for nonlinear stochastic gradient descent (SGD) under heavy-tail gradient noise. Unlike existing studies of SGD under heavy-tail noise that focus on specific nonlinear functions (e.g., adaptive clipping), our framework includes a broad class of component-wise (e.g., sign gradient) and joint (e.g., gradient clipping) nonlinearities. We establish for the considered methods almost sure convergence, MSE convergence rate, and also asymptotic covariance for component-wise nonlinearities. We carry out numerical experiments on several real datasets that exhibit heavy tail gradient noise effects. The experiments show that, while our framework is more general than existing studies of SGD under heavy-tail noise, several easy-to-implement nonlinearities from our framework are competitive with state-of-the-art alternatives.

REFERENCES

[1] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, *QSGD: Communication-efficient sgd via gradient quantization and encoding*, in Advances in Neural Information Processing Systems, 2017, pp. 1709–1720.

[2] L. Balles, F. Pedregosa, and N. L. Roux, *The geometry of sign gradient descent*, arXiv preprint arXiv:2002.08056, (2020).

[3] H. Bercovici and V. Pata, *Stable laws and domains of attraction in free probability theory*, Ann. of Math., 149 (1999), pp. 1023–1060.

[4] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, *signsgd: Compressed optimisation for non-convex problems*, in International Conference on Machine Learning, PMLR, 2018, pp. 560–569.

[5] L. Bottou, *Large-scale machine learning with stochastic gradient descent*, in Proceedings of COMPSTAT'2010, Springer, 2010, pp. 177–186.

[6] L. Bottou, F. E. Curtis, and J. Nocedal, *Optimization methods for large-scale machine learning*, Siam Review, 60 (2018), pp. 223–311.

[7] R. H. Byrd, G. M. Chin, W. Neveitt, and J. Nocedal, *On the use of stochastic hessian information in optimization methods for machine learning*, SIAM Journal on Optimization, 21 (2011), pp. 977–995.

[8] V. Cevher, S. Becker, and M. Schmidt, *Convex optimization for big data: Scalable, randomized, and parallel algorithms for big data analytics*, IEEE Signal Processing Magazine, 31 (2014), pp. 32–43.

[9] C.-C. Chang and C.-J. Lin, *Libsvm: a library for support vector machines*, ACM Transactions on intelligent systems and technology (TIST), 2 (2011), pp. 1–27.

[10] S. Dasarathan, C. Tepedelenlioğlu, M. K. Banavar, and A. Spanias, *Robust consensus in the presence of impulsive channel noise*, IEEE Transactions on Signal Processing, 63 (2015), pp. 2118–2129.

[11] D. Davis, D. Drusvyatskiy, L. Xiao, and J. Zhang, *From low probability to high confidence in stochastic convex optimization.*, J. Mach. Learn. Res., 22 (2021), pp. 49–1.

[12] S. Ghadimi and G. Lan, *Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization i: A generic algorithmic framework*, SIAM Journal on Optimization, 22 (2012), pp. 1469–1492.

[13] S. Ghadimi and G. Lan, *Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization I: A generic algorithmic framework*, SIAM J. Optim., 22 (2012), pp. 1469–1492.

[14] S. Ghadimi and G. Lan, *Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization, II: Shrinking procedures and optimal algorithms*, SIAM J. Optim., 23 (2013), pp. 2061–2089.

[15] E. Gorbunov, M. Danilova, and A. Gasnikov, *Stochastic optimization with heavy-tailed noise via accelerated gradient clipping*, arXiv preprint arXiv:2005.10785, (2020).

[16] E. Gorbunov, F. Hanzely, and P. Richtárik, *A unified theory of sgd: Variance reduction, sampling, quantization and coordinate descent*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 680–690.

[17] M. Gurbuzbalaban, U. Simsekli, and L. Zhu, *The heavy-tail phenomenon in sgd*, in International Conference on Machine Learning, PMLR, 2021, pp. 3964–3975.

[18] S. Horváth, D. Kovalev, K. Mishchenko, S. Stich, and P. Richtárik, *Stochastic distributed learning with gradient quantization and variance reduction*, arXiv preprint arXiv:1904.05115, (2019).

[19] A. Juditsky, A. Nazin, A. Nemirovsky, and A. Tsybakov, *Algorithms of robust stochastic optimization based on mirror descent method*, arXiv:1907.02707, (2019).

[20] S. Kar, J. M. Moura, and K. Ramanan, *Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication*, IEEE Transactions on Information Theory, 58 (2012), pp. 3575–3605.

[21] S. Kar and J. M. F. Moura, *Convergence rate analysis of distributed gossip (linear parameter) estimation: Fundamental limits and tradeoffs*, IEEE Jour. Sel. Top. Sig. Proc., 5 (2011), pp. 674–690.

[22] U. A. Khan, S. Kar, and J. M. Moura, *Distributed average consensus: Beyond the realm of linearity*, in 2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers, IEEE, 2009, pp. 1337–1342.

[23] L. Lei and M. I. Jordan, *On the adaptivity of stochastic gradient-based optimization*, SIAM Journal on Optimization, 30 (2020), pp. 1473–1500.

[24] H. Mania, X. Pan, D. Papailiopoulos, B. Recht, K. Ramchandran, and M. I. Jordan,

Perturbed iterate analysis for asynchronous stochastic optimization, SIAM Journal on Optimization, 27 (2017), pp. 2202–2229.
[25] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, *Robust stochastic approximation approach to stochastic programming*, SIAM Journal on optimization, 19 (2009), pp. 1574–1609.
[26] M. B. Nevelson and R. Z. Khasminskiĭ, *Stochastic approximation and recursive estimation*, vol. 47, American Mathematical Soc., 1976.
[27] F. Niu, B. Recht, C. Ré, and S. J. Wright, *Hogwild!: A lock-free approach to parallelizing stochastic gradient descent*, arXiv preprint arXiv:1106.5730, (2011).
[28] R. Pascanu, T. Mikolov, and Y. Bengio, *On the difficulty of training recurrent neural networks*, in International Conference on Machine Learning, PMLR, 2013, pp. 1310–1318.
[29] V. Pichapati, A. T. Suresh, F. X. Yu, S. J. Reddi, and S. Kumar, *Adaclip: Adaptive clipping for private sgd*, arXiv preprint arXiv:1908.07643, (2019).
[30] B. T. Polyak and Y. Z. Tsypkin, *Adaptive estimation algorithms: convergence, optimality, stability*, Avtomatika i Telemekhanika, (1979), pp. 71–84.
[31] A. Shapiro, D. Dentcheva, and A. Ruszczynski, *Lectures on stochastic programming: modeling and theory*, SIAM, 2021.
[32] U. Simsekli, M. Gürbüzbalaban, T. H. Nguyen, G. Richard, and L. Sagun, *On the heavy-tailed theory of stochastic gradient descent for deep neural networks*, arXiv preprint arXiv:1912.00018, (2019).
[33] S. S. Stanković, M. Beko, and M. S. Stanković, *A robust consensus seeking algorithm*, in IEEE EUROCON 2019-18th International Conference on Smart Technologies, IEEE, 2019, pp. 1–6.
[34] S. Sundaram and B. Gharesifard, *Consensus-based distributed optimization with malicious nodes*, in 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton), IEEE, 2015, pp. 244–249.
[35] F. Yousefian, A. Nedić, and U. V. Shanbhag, *On stochastic gradient and subgradient methods with adaptive steplength sequences*, Automatica, 48 (2012), pp. 56–67.
[36] J. Zhang, T. He, S. Sra, and A. Jadbabaie, *Why gradient clipping accelerates training: A theoretical justification for adaptivity*, arXiv preprint arXiv:1905.11881, (2019).
[37] J. Zhang, S. P. Karimireddy, A. Veit, S. Kim, S. J. Reddi, S. Kumar, and S. Sra, *Why are adaptive methods good for attention models?*, arXiv preprint arXiv:1912.03194, (2019).

Appendix.

A. Proof of Theorem 5.2. We first state and prove the following Lemma.

Lemma 8.1. *Consider a deterministic sequence*

$$v^{t+1} = \left(1 - \frac{a_3}{(t+1)^\delta}\right) v^t + \frac{a_4}{(t+1)^\delta},\ t \geq t_0,$$

with $a_3, a_4 > 0$ *and* $0 < \delta \leq 1$, $t_0 > 0$, *and* $v^{t_0} \geq 0$. *Further, assume that* t_0 *is such that* $\frac{a_3}{(t+1)^\delta} \leq 1$, *for all* $t \geq t_0$. *Then,* $\lim_{t\to\infty} v^t = \frac{a_4}{a_3}$.

Proof. Let $e^t = v^t - \frac{a_4}{a_3}$. It is easy to verify that:

$$e^{t+1} = \left(1 - \frac{a_3}{(t+1)^\delta}\right) e^t,\ t \geq t_0.$$

Then, for all $t \geq t_0$, there holds:

$$|e^{t+1}| = \left(1 - \frac{a_3}{(t+1)^\delta}\right) |e^t| \leq \exp\left(-a_3 \sum_{s=t_0}^{t} \frac{1}{(s+1)^\delta}\right) |e^{t_0}| \tag{8.1}$$

where in (8.1) we used the inequality $1 + a \leq \exp(a)$, $a > 0$. Letting $t \to \infty$ and the fact that $\delta \leq 1$ so that the sequence $\frac{1}{(s+1)^\delta}$, $s \geq t_0$, is non-summable, we obtain that $e^t \to 0$, which in turn implies the claim of the Lemma. ∎

We now continue with proving Theorem 5.2. First, let us prove claim (1). Note that:

$$z^{t+1} \leq \left(1 - \frac{a_1}{(t+1)^{\delta_1}}\right) z^t + \frac{a_2}{(t+1)^{\delta_2}},\ t \geq t'. \tag{8.2}$$

Multiplying the above inequality with $(t+1)^{\delta_2-\delta_1}$, defining $\widehat{z}^t = t^{\delta_2-\delta_1}\, z^t$, we get:

$$\widehat{z}^{t+1} \leq \left(1 - \frac{a_1}{(t+1)^{\delta_1}}\right)(1+1/t)^{\delta_2-\delta_1}\,\widehat{z}^t + \frac{a_2}{(t+1)^{\delta_1}}.$$

Next, using, e.g., a Taylor expansion of function $a \mapsto (1+a)^{\delta_2-\delta_1}$, it can be shown that $(1+1/t)^{\delta_2-\delta_1} \leq 1 + \frac{2(\delta_2-\delta_1)}{t}$, for any $t \geq t_\delta$, for appropriately chosen $t_\delta > 0$. Therefore,

$$\begin{aligned}&\left(1 - \frac{a_1}{(t+1)^{\delta_1}}\right)(1+1/t)^{\delta_2-\delta_1}\\ &\leq 1 - \frac{a_1}{(t+1)^{\delta_1}} + \frac{2(\delta_2-\delta_1)}{t} - \frac{2a_1(\delta_2-\delta_1)}{t(t+1)^{\delta_1}} \leq 1 - \frac{a_1}{2(t+1)^{\delta_1}},\end{aligned}$$

for any $t \geq t_1$, for appropriately taken $t_1 > 0$. Using the latter bound, we obtain: $\widehat{z}^{t+1} \leq \left(1 - \frac{a_1}{2(t+1)^{\delta_1}}\right)\widehat{z}^t + \frac{a_2}{(t+1)^{\delta_1}}$, $t \geq t_1$. Now, applying Lemma 8.1, we obtain that $\widehat{z}^t = O(1)$, and therefore $z^t = O(1/t^{\delta_2-\delta_1})$. This proves claim (1) in Theorem 5.2.

We now prove claim (2). Multiplying (8.2) by $(t+1)^{\delta_2-1}$ and defining $\widehat{z}^t = t^{\delta_2-1}\, z^t$, we obtain:

$$\begin{aligned}\widehat{z}^{t+1} &\leq \left(1 - \frac{a_1}{(t+1)}\right)(1+1/t)^{\delta_2-1}\widehat{z}^t + \frac{a_2}{t+1}\\ &\leq \left(1 - \frac{a_1-(\delta_2-1)}{(t+1)} + \frac{C_{22}}{t^2}\right)\widehat{z}^t + \frac{a_2}{t+1} \qquad (8.3)\\ &\leq \left(1 - \frac{a_1-(\delta_2-1)}{2\,(t+1)}\right)\widehat{z}^t + \frac{a_2}{t+1},\ t \geq t_2, \qquad (8.4)\end{aligned}$$

for appropriately chosen $t_2 > 0$ and $C_{22} > 0$. In (8.3), we used the fact that $(1+1/t)^{\delta_2-1} \leq 1 + \frac{\delta_2-1}{t} + \frac{C_{23}}{t^2}$, for all $t \geq 1$ and some $C_{23} > 0$ (the inequality can be obtained, e.g., via a Taylor approximation). The claim (2) of Theorem 5.2 now follows by applying Lemma 8.1 to (8.4).

We now prove claim (3). Let $a_1 < \delta_2 - 1$, and fix an arbitrary positive number ζ, $\zeta < a_1$. Then, we have, for $\widehat{z}^t = t^\zeta z^t$:

$$\begin{aligned}\widehat{z}^{t+1} &\leq \left(1 - \frac{a_1}{(t+1)}\right)(1+1/t)^\zeta\widehat{z}^t + \frac{a_2}{(t+1)^{\delta_2-\zeta}}\\ &\leq \left(1 - \frac{a_1-\zeta}{t} + \frac{C_{24}}{t^2}\right)\widehat{z}^t + \frac{a_2}{(t+1)^{\delta_2-\zeta}}\\ &\leq \left(1 - \frac{a_1-\zeta}{2\,(t+1)}\right)\widehat{z}^t + \frac{a_2}{t+1},\ t \geq t_3,\end{aligned}$$

for appropriately chosen $t_3 > 0$ and $C_{24} > 0$. In the last inequality, we used the fact that $\zeta < a_1 \leq \delta_2 - 1$, and so $\delta_2 - \zeta > 1$. Finally, applying Lemma 8.1, claim (3) follows. □

B. A demonstration that the linear SGD's iterate sequence has infinite variance. We provide here a simple demonstration that the linear SGD's iterate sequence has infinite variance under the setting of Assumption 1, Assumption 3, and Assumption 5, condition 3., holds.

More precisely, assume that the gradient noise ν^t has infinite variance. Consider algorithm (2.3) for solving problem (1) with $f : \mathbb{R} \mapsto \mathbb{R}$, $f(x) = \frac{x^2}{2}$, with Ψ being the identity function. Further, consider arbitrary sequence of positive step-sizes $\{\alpha_t\}$. Then, we have:

$$x^{t+1} = (1 - \alpha_t)\, x^t - \alpha_t\, \nu^t,\ t = 0, 1, ..., \tag{8.5}$$

with arbitrary deterministic initialization $x^0 \in \mathbb{R}$. Then, squaring (8.5), using the independence of x^t and ν^t, and the fact that ν^t has zero mean, we get: $\mathbb{E}\left[(x^{t+1})^2\right]$ $= (1-\alpha_t)^2\, \mathbb{E}\left[(x^t)^2\right] + \alpha_t^2\, \mathbb{E}[(\nu^t)^2] \geq \alpha_t^2\ \mathbb{E}[(\nu^t)^2]$, $t = 0, 1, ...$ Taking expectation and using the fact that $\mathbb{E}[(\nu^t)^2] = +\infty$, we see that $\mathbb{E}\left[(x^t)^2\right] = +\infty$, for any $t \geq 1$.

C. Extension of Theorem 3.2 for gradient noise vector with mutually dependent entries. We show that Theorem 3.2 continues to hold when we have an i.i.d. zero mean noise vector sequence $\{\boldsymbol{\nu}^t\}$ with a joint pdf $p : \mathbb{R}^d \mapsto \mathbb{R}$. In more detail, we provide an extension of Lemma 6.2 but for component-wise nonlinearities.

Namely, as in Lemma 6.2, consider, for a fixed $\mathbf{y} \neq 0$:

$$\int \psi(\mathbf{y} + \mathbf{u})^\top \mathbf{y}\, p(\mathbf{u})\, d\mathbf{u}. \tag{8.6}$$

As, for $\mathbf{a} \in \mathbb{R}^d$, we have $\Psi(\mathbf{a}) = (\Psi(a_1), ..., \Psi(a_d))$ (component-wise nonlinearity), we have:

$$\int \psi(\mathbf{y} + \mathbf{u})^\top \mathbf{y}\, p(\mathbf{u})\, d\mathbf{u} = \int \left(\sum_{i=1}^d \psi(y_i + u_i) y_i\right) p(\mathbf{u})\, d\mathbf{u}$$
$$= \sum_{i=1}^d \int (\psi(y_i + u_i) y_i)\, p(\mathbf{u})\, d\mathbf{u} = \sum_{i=1}^d \int (\psi(y_i + u_i) y_i)\ p_i(u_i)\, du_i,$$

where $p_i(u_i)$ is the marginal pdf of the i-th component of $\boldsymbol{\nu}^t$. It is easy to show, as $p(\mathbf{u}) = p(-\mathbf{u})$, $\mathbf{u} \in \mathbb{R}^d$, that, for any $i = 1, ..., d$, we have $p_i(u) = p_i(-u)$, $u \in \mathbb{R}$. Define $\phi_i(a) = \int \Psi(a + u) p_i(u) du$. Note that $\phi_i(a)$ now obeys Lemma 5.3. In particular, ϕ_i is also odd, and hence:

$$\int \psi(\mathbf{y} + \mathbf{u})^\top \mathbf{y}\, p(\mathbf{u})\, d\mathbf{u} = \sum_{i=1}^d \int (\psi(y_i + u_i) y_i)\ p_i(u_i)\, du_i$$
$$= \sum_{i=1}^d \phi_i(y_i) y_i = \sum_{i=1}^d |\phi_i(y_i)|\, |y_i|.$$

The last inequality holds because, for any $i = 1, ..., d$, quantities $\phi_i(y_i)$ and y_i have equal sign. The proof now proceeds analogously to that of Theorem 3.2.

D. Proof of Lemma 5.3. The proof can be found in [30]; we include similar arguments for completeness. For claim 1., note that

$$\phi(a) = \int_{-\infty}^{+\infty} \Psi(a + u) p(u) du = -\int_{-\infty}^{+\infty} \Psi(-a - u) p(u) du$$
$$= -\int_{-\infty}^{+\infty} \Psi(-a + w) p(w) dw = -\phi(-a),$$

for any $a \in \mathbb{R}$, where we use the fact that Ψ is odd. For claim 2., note that $|\phi(a)| \leq \int_{-\infty}^{+\infty} |\Psi(a+u)| p(u)du \leq C_1 \int_{-\infty}^{+\infty} p(u)du = C_1$, where we used Assumption 7. Proof of claim 3. is similar to that of claim 2. For claim 4., note that $\phi(a) = \int_0^{+\infty} (\Psi(u+a) - \Psi(u-a))\, p(u)du$, and so, for $a' > a$, we have

$$\phi(a') - \phi(a) = \int_0^{+\infty} \quad [(\Psi(u+a') - \Psi(u+a)) + \\ + (\Psi(u-a) - \Psi(u-a'))]\, p(u)du \geq 0,$$

because Ψ is non-decreasing. Finally, for claim 5., to show that $\phi'(0)$ is given by (5.4), see the proof of Lemma 6 in [30]. To verify that $\phi'(0)$ is strictly positive, consider first the case that Ψ has a discontinuity at zero. Then, because $p(0) > 0$ by Assumption 3, it follows from (5.4) that $\phi'(0) \geq (\Psi(0+) - \Psi(0-))p(0) > 0$. Otherwise, if Ψ is continuous at zero, we have: $\phi'(0) \geq \int_{-c}^{c} \Psi'(u)p(u)du > 0$, where $c > 0$ is taken such that $\Psi(u)$ is continuous and strictly increasing and $p(u)$ is strictly positive for $|u| < c$.[8] Such c exists in view of Assumptions 3 and 5.

E. Derivations for Example 3.3. We calculate the rate ζ in Theorem 3.2 for the component-wise clipping nonlinearity with saturation value m, $m > 1$. Here, it can be shown, by doing direct calculations, that

$$\phi(w) = 2\, w \int_0^{m-w} p(u)du + \int_{m-w}^{m+w} (m+w-u)p(u)du, \;\; w \in [0, m]. \tag{8.7}$$

Furthermore, it can be shown that (see Appendix F): $\phi'(0) = 2\int_0^m p(u)du$. Noting that the second integral in (8.7) is nonnegative, and using the form $p(u)$ in (3.2), we obtain:

$$\phi(w) \geq 2\, w \int_0^{m-w} p(u)du = w\left(1 - \frac{1}{(m-w+1)^{\alpha-1}}\right), \; w \in [0, m]. \tag{8.8}$$

Also, we have: $\phi'(0) = 1 - \frac{1}{(m+1)^{\alpha}}$. From the latter equation and (8.8), we estimate that ξ can be taken as: $\xi = m + 1 - \left(\frac{2}{1+(m+1)^{-\alpha}}\right)^{1/(\alpha-1)} \geq m-1$, for any $\alpha > 2$, for any $m > 1$. Hence, we can also take $\xi = m-1$. Substituting the obtained estimates for $\phi'(0)$ and ξ into the rate ζ, we obtain the rate estimate in (3.3).

F. Derivation of $\phi'(0)$ for Example 3.5. Consider the coordinate-wise clipping nonlinearity Ψ with floor level $m > 0$. The function Ψ here is piece-wise differentiable, with the derivative $\Psi'(a) = 1$, for $a \in (-m, m)$, and $\Psi'(a) = 0$, for $|a| > m$. We now apply claim 5. in Lemma 5.3 and use formula (5.4) for evaluating $\phi'(0)$. As the coordinate-wise clipping function does not have discontinuity points, (5.4) simplifies to the following:

$$\phi'(0) = \int_{u \in \mathbb{R},\, u \neq -m,\, u \neq m} \Psi'(u)p(u)\, du = \int_{-m}^{+m} p(u)du = 2\int_0^m p(u)du,$$

where the last equality uses symmetry of function $p(u)$.

[8]If there are some (at most countably many) points inside interval $(-c, c)$ where Ψ is continuous but not differentiable, these points are excluded from the integration set in $\int_{-c}^{c} \Psi'(u)p(u)du$ without change in the integration result.

G. Derivations for Example 3.6. We provide here details for the derivations in Example 3.6. We first calculate σ_Ψ^2; we have:

$$\sigma_\Psi^2 = \int_{-\infty}^{\infty} |\Psi(u)|^2 p(u) du = \int_{-\infty}^{\infty} p(u) du = 1.$$

Next, by direct integration, we have for $\alpha > 3$:

$$\begin{aligned}\sigma_\nu^2 &= 2\int_0^\infty p(u) u^2 du \\ &= -(\alpha-1)\frac{[\,(\alpha-1)u((\alpha-2)u+2)\,]+2}{(\alpha-3)(\alpha-2)(\alpha-1)(1+u)^{\alpha-2}}\,|_0^\infty = \frac{2}{(\alpha-3)(\alpha-2)}.\end{aligned}$$

On the other hand, for $\alpha \in (2,3]$, we clearly have $\sigma_\nu^2 = +\infty$. Finally, using claim 5. in Lemma 5.3, and using the fact that $\Psi'(u) = 0$, for all $u \neq 0$, we obtain:

$$\phi'(0) = p(0)\,(\Psi(0+) - \Psi(0-)) = 2\,p(0) = \alpha - 1.$$

H. Derivations for Example 4.1. We consider the (joint) gradient clipping nonlinearity $\mathbf{\Psi}$ with the clipping level $M > 0$, and we consider $p(\mathbf{u})$ in (4.1).

Consider rate ζ in Theorem 4.2 that, for a sufficiently large α, can be approximated as:

$$\min\left\{2\delta - 1,\, (1-\delta)\frac{4\,\mu\,(1-\kappa)\lambda(\kappa)\mathcal{N}(1)}{L\,C_2'}\right\}. \tag{8.9}$$

Here, κ is an arbitrary scalar in $(0,1)$, and for the gradient clipping, we have that $\mathcal{N}(1) = C_2' = M$. Note that, regarding Assumption 8, quantity B_0 can be taken here to be an arbitrary positive number. Moreover, for $p(\mathbf{u})$ in (4.1), due to the radial symmetry, we have that

$$\lambda(\kappa) = [illegible](\kappa, B_0) = \frac{[illegible]}{\pi}\text{arc co}[illegible](1-\kappa)\,\mathcal{P}(B_0),\ \ \kappa \in (0,1),$$

where $\mathcal{P}(B_0) = \int_{\mathbf{u}:\,\|\mathbf{u}\|\le B_0} p(\mathbf{u}) d\mathbf{u} = 1 - \frac{1+(\alpha-1)B_0}{(1-B_0)^{\alpha-1}}$. We next maximize (8.10), i.e., we maximize $(1-\kappa)\lambda(\kappa, B_0)$ with respect to $\kappa \in (0,1)$, to get the largest (tightest) estimate of ζ. It is easy to see that $\max_{\kappa\in(0,1)}(1-\kappa)\lambda(\kappa, B_0) > 0.17\,\mathcal{P}(B_0)$. Substituting all the above developments into (8.10), we obtain:

$$\begin{aligned}\zeta &\approx \min\left\{2\delta - 1,\, (1-\delta)\frac{0.68\,\mu\,\mathcal{P}(B_0)}{L}\right\} \\ &= \min\left\{2\delta - 1,\, (1-\delta)\frac{0.68\,\mu}{L}\left(1 - \frac{1+(\alpha-1)B_0}{(1+B_0)^{\alpha-1}}\right)\right\}\end{aligned} \tag{8.10}$$

As B_0 can be arbitrary positive number, letting $B_0 \to +\infty$, we obtain the following rate estimate: $\min\left\{2\delta - 1,\, (1-\delta)\frac{0.68\,\mu}{L}.\right\}$. It is easy to see that the same rate estimate can be obtained for the normalized gradient nonlinearity. The only difference in the rate derivation is that therein $\mathcal{N}(1) = C_2' = 1$.

F Large Deviations Rates for Stochastic Gradient Descent with Strongly Convex Functions

The appended preprint follows.

Large deviations rates for stochastic gradient descent with strongly convex functions

Dragana Bajovic
Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia
dbajovic@uns.ac.rs

Dusan Jakovetic
Faculty of Sciences, University of Novi Sad, Novi Sad, Serbia
dusan.jakovetic@dmi.uns.ac.rs

Soummya Kar
Carnegie Mellon University, Pittsburgh, PA, USA
soummyak@andrew.cmu.edu [*]

Abstract

Recent works have shown that high probability metrics with stochastic gradient descent (SGD) exhibit informativeness and in some cases advantage over the commonly adopted mean-square error-based ones. In this work we provide a formal framework for the study of general high probability bounds with SGD, based on the theory of large deviations. The framework allows for a generic (not-necessarily bounded) gradient noise satisfying mild technical assumptions, allowing for the dependence of the noise distribution on the current iterate. Under the preceding assumptions, we find an upper large deviations bound for SGD with strongly convex functions. The corresponding rate function captures analytical dependence on the noise distribution and other problem parameters. This is in contrast with conventional mean-square error analysis that captures only the noise dependence through the variance and does not capture the effect of higher order moments nor interplay between the noise geometry and the shape of the cost function. We also derive exact large deviation rates for the case when the objective function is quadratic and show that the obtained function matches the one from the general upper bound hence showing the tightness of the general upper bound. Numerical examples illustrate and corroborate theoretical findings.

[*]This work is supported by the European Union's Horizon 2020 Research and Innovation program under grant agreement No 957337. The paper reflects only the view of the authors and the Commission is not responsible for any use that may be made of the information it contains.

1 Introduction

The large deviations theory represents a well-established principled approach for studying *rare events* that occur with stochastic processes, e.g., (Dembo et al. 1993). Typically, we are concerned with a sequence of rare events E_k related with the stochastic process of interest, indexed by, e.g., time k. In this setting, the probability of event E_k, $k = 1, 2, ...$ typically decays exponentially in k; the large deviations theory then enables to quantify this exponential rate. Such an approach has found many applications in statistics (Bucklew 1990), mechanics (Touchette 2009), communications (Shwartz et al. 1995), and information theory (T. M. Cover et al. 1991).

To be more concrete, consider an example of a sequence of random vectors X_k taking values in $\mathbb{R}^d$ that converge, e.g., almost surely, to a (deterministic) limit point $x^\star \in \mathbb{R}^d$. The rare event of interest E_k can then be, for example, $E_k = \{\|X_k - x^\star\| \geq \delta\}$, for some positive quantity δ, with $\|\cdot\|$ denoting the Euclidean norm. Equivalently, E_k can be represented as $\{X_k \in C_\delta\}$, where C_δ is the complement of the l_2 ball of radius δ centered at $x^\star$. Large deviations analysis then aims at discovering the corresponding rate of decay, i.e., the inaccuracy rate $\mathbf{I}(C_\delta)$:

$$\mathbb{P}(X_k \in C_\delta) = e^{-k\,\mathbf{I}(C_\delta)+o(k)}, \tag{1}$$

where $o(k)$ denotes terms growing slower than linearly with k. The inaccuracy rate $\mathbf{I}(C_\delta)$ can usually be expressed via the so called *rate function* $I : \mathbb{R}^d \mapsto \mathbb{R}$ according to the following formula (Bahadur 1960):

$$\mathbf{I}(C_\delta) = \inf_{x \in C_\delta} I(x). \tag{2}$$

Differently from the set function $\mathbf{I}$, the rate function I does not depend on the region C_δ; that is, when C_δ changes, only the region over which we minimize in (2) changes, while the function remains unchanged. Furthermore, this is true for arbitrary set C_δ. This means that, once the rate function is computed, the corresponding inaccuracy rate can be obtained via (2) for a new given region of interest.

In this paper, we are interested in applying the large deviations theory to analyzing the stochastic gradient descent (SGD) method. SGD is a simple but widely used optimization method that finds numerous practical applications, such as training machine learning and deep learning models, e.g., (Niu et al. 2011; Gorbunov, Hanzely, et al. 2020; Lei et al. 2020). More precisely, we consider unconstrained optimization problems where the goal is to minimize a smooth, strongly convex function $f : \mathbb{R}^d \to \mathbb{R}$, via the SGD method of the form:

$$X_{k+1} = X_k - \alpha_k \left(\nabla f(X_k) - Z_k\right). \tag{3}$$

Here, $k = 1, 2, ...$ is the iteration counter, $\alpha_k = a/k$, $a > 0$ is the step-size, and Z_k is a zero-mean gradient noise that may depend on X_k. In this context, we are interested in solving for (1) and (2) for the SGD method (3), where now $x^\star$ is interpreted as the (deterministic) global minimizer of f. In other words, we are interested in finding (or approximating) the rate function $I(x)$ that quantifies the "tails" or "rare events" of how the SGD sequence iterates X_k deviate from the solution $x^\star$.

Clearly, evaluating (2) for SGD is of significant interest. It readily provides insights into the high-probability bounds for SGD that have been subject of much research effort recently, (Ghadimi et al. 2012; Ghadimi et al. 2013; Juditsky et al. 2019; Gorbunov, Danilova, et al. 2020; Davis et al. 2021). However, unlike the typical high probability bound studies, the large deviations approach here is fully flexible with respect to the choice of set C_δ; e.g., the l_2-ball complement may be replaced with an arbitrary open set, such as l_p norm complement of an arbitrary l_p norm. While large deviations theory is a well-established field, there has been a limited body of work that applies large deviations to the analysis of SGD. Reference (Woodroofe 1972) is concerned with large deviations analysis for a scalar stochastic process equivalent to SGD in one dimension. The authors of (W. Hu et al. 2019) study large deviations of SGD when the step-size converges to zero; however, they are not concerned with large deviations when the iteration counter k increases – the case of our interest here.

Contributions. In this paper, we are interested in evaluating the large deviations rates in (1) and (2) for the SGD method, when the objective function f is smooth and strongly convex. Our main contributions are as follows. When f is a (strongly convex) quadratic function, we establish the so-called full large deviations principle for the sequence X_k. This means that we evaluate rate function $I(x)$ exactly, i.e., the corresponding rare event probability is computed exactly, with upper and lower bounds matched, up to exponentially decaying factors. We further explicitly quantify the rate function $I(x)$ as a function of the distribution of the gradient noise. This reveals a significant influence of higher order moments on the performance (in the sense of rare event probabilities) of SGD. This is in contrast with conventional SGD analyses, that typically capture only the dependence on the gradient noise variance. The large deviations principle for quadratic functions is established under a very general class of gradient noise distributions that are essentially only required to have a finite moment generating function. Next, for generic smooth and strongly convex costs f, we establish a large deviations upper bound (a lower bound on function $I(x)$) that certifies an exponential decay of the rare event probabilities in (1) with SGD. This is achieved when the distribution of the gradient noise is sub-Gaussian. We further show that the obtained large deviations upper bound is tight, as the corresponding rate function actually matches, up to higher order factors, the exact rate function that we formerly establish for the quadratic costs.

Our results are related with high probability bounds-type studies of SGD and related stochastic methods (Harvey et al. 2019; Ghadimi et al. 2012; Ghadimi et al. 2013; Juditsky et al. 2019; Gorbunov, Danilova, et al. 2020). Therein, for a given $\delta > 0$ and a confidence level $1-\beta$, $\beta \in (0,1)$, the goal is to find $K(\delta, \beta)$ such that $f(X_k) - f(x^\star) \leq \delta$ with probability at least $1-\beta$, for all $k \geq K(\delta, \beta)$. The works (Ghadimi et al. 2012; Ghadimi et al. 2013; Juditsky et al. 2019; Gorbunov, Danilova, et al. 2020) provide estimates of $K(\delta, \beta)$ that depend *logarithmically* on β. In more detail, (Ghadimi et al. 2012; Ghadimi et al. 2013) establish high probability bounds for the stochastic gradient methods therein assuming sub-Gaussian gradient noises. The work (Juditsky et al. 2019) calculates the corresponding bounds for the basic SGD and the mirror descent that utilize a gradient truncation technique, while relaxing the noise sub-Gaussianity. The work (Gorbunov, Danilova, et al. 2020) establishes high probability bounds for an accelerated SGD that also utilizes a clipping nonlinearity. The large deviations rates in (1) and (2) - give estimates of $K(\delta, \beta)$ that also depend logarithmically on β, when β is small (going to zero).[1]

Compared with existing high probability bound works, our results give the *exact* (tight) exponential decay rate in (2), and for an *arbitrary set* that does not contain $x^\star$, not only the Euclidean ball complements. To be concrete, the closest results to ours are obtained in (Harvey et al. 2019). While they are not directly concerned with obtaining large deviations rates, their results (with some additional work) lead to an exponential decay rates for Euclidean ball complements. In contrast, our results work for arbitrary open sets. Furthermore, focusing only on Euclidean ball complements, our results provide much tighter exponential rate bounds. Specifically, as we show in the paper, the exponential rate that we provide captures the interplay be-

[1]It is easy to see this by noting that, for μ-strongly convex costs, we have $f(x) - f(x^\star) \geq \frac{\mu}{2}\|x - x^\star\|^2$, for all $x \in \mathbb{R}^d$, requiring that the the right hand side of (1) be less than β, and reverse-engineering the smallest iterate k for which the latter holds.

tween the noise geometry and the cost function curvature, see Section 4.2 for details. From the technical perspective, this is achieved by working directly with the SGD iterates, as opposed to working with the distance of the iterates from the solution. To do so, we derive a novel set of techniques that build upon the large deviations theory rather than on martingale concentration inequalities.

The current paper is also related with large deviations analyses of stochastic processes that arise with distributed inference, such as estimation and detection. Distributed detection has been studied in (D. Bajović et al. 2011), for Gaussian observations, and in (Bajović et al. 2012), for generic observations. The work (Matta et al. 2016a) evaluates large deviations of the local states with a distributed detection method when the step size parameter decreases. Reference (Matta et al. 2016b) further analyzes the non-exponential terms and consider directed networks for a similar problem. The paper (Marano et al. 2019) considers distributed detection with 1-bit messages. (P. Hu et al. 2022) considers social learning problems. Reference (Bajovic 2022) analyzes large deviations for distributed estimation and social learning. Unlike these works on distributed inference, we are not directly concerned with distributed systems; also, the cost functions that we consider are more general and, unlike the works above, do not result in linear (distributed averaging) dynamics; hence, novel tools for large deviations analysis are required here.

The rest of the paper is organized as follows. Section 2 explains the problem that we consider and gives the required preliminaries. Section 3 provides the main results of the paper – a large deviations upper bound for generic costs and the full (exact) large deviations rates for quadratic costs. Specializing to the Gaussian noise, Section 4 provides analytical, closed-form expressions for the large deviations rate function. Section 5 gives the proof of the main lemma underlying the upper bound for the general functions. Finally, we conclude in Section 6. Appendix contains additional insights and examples, numerical results, and missing proofs.

2 Setup and preliminaries

We consider unconstrained optimization problems of the form

$$\min_{x \in \mathbb{R}^d} f(x). \tag{4}$$

We assume that f is L-smooth and μ-strongly convex, and that the stepsize in algorithm (3) is of the form $\alpha_k = a/(k+b)$, where a, $b > 0$.

Assumption 1. *We assume that f is twice differentiable, L-smooth and μ-strongly convex, where $0 < \mu \leq L$.*

Strong convexity implies uniqueness of the solution of (4), which is denoted by $x^\star$. We make the following assumption regarding the stepsize parameter a.

Assumption 2. *The stepsize parameter a satisfies $a\mu > 1$.*

Assumption 1 is standard in the analysis of optimization methods, i.e., it corresponds to a standard class of functions over which an optimization method analysis is carried out. Assumption 2 is required for some asymptotic arguments ahead, as $k \to \infty$. In practice, it may be restrictive that the constant a is too large in the step-size choice a/k, as at the initial iterations (small k's), we would have very large step-sizes. This is alleviated by having an appropriately chosen constant $b > 1$.

We denote by $\tilde{g}(X_k)$ the stochastic gradient of f returned by the gradient oracle at the current iterate X_k, and by $g(X_k)$ the (exact) gradient of f at the current iterate X_k. The difference between $\tilde{g}(X_k)$ and $g(X_k)$ (the gradient "noise") is denoted by $Z_k = \tilde{g}(X_k) - g(X_k)$. We make the following assumptions on Z_k.

Assumption 3.
1. *For each k, Z_k depends on the past iterates only through X_k.*
2. *For each k, the distribution of Z_k given X_k depends on X_k only through its realization and does not depend on the current iterate index, k.*
3. *For any given x, $\mathbb{E}[Z_k | X_k = x] = 0$, i.e., conditioned on the current iterate, the noise is zero-mean.*

Assumption 3 allows for a general gradient noise that may actually depend on the current iterate X_k. This is a more general setting than the frequently studied case when Z_k is i.i.d. and independent of X_k. Item 3. of Assumption 3 says that, conditioned on the current iterate, the noise is zero-mean on average. This is also a standard bias-free noise assumption. Finally, note that items 1. and 2. in Assumption 3 typically hold in machine learning settings. Therein, the goal is typically to minimize a population loss $f(x) = \mathbb{E}[\phi(x, v)]$ where the expectation is taken over the distribution of the data v, and ϕ is an instantaneous loss function. Given that, at some iteration k, X_k takes a value x, the gradient noise equals $\nabla_x \phi(x, v_k) - \mathbb{E}[\nabla_x \phi(x, v)]$, where v_k is the data point sampled at iteration k. Then, items 1. and 2. are clearly satisfied, provided that the data sampling process is independent of the evolution of X_k.

For $x \in \mathbb{R}^d$, we denote by $H(x)$ the Hessian matrix of f computed at x. For compactness, we denote $H^\star = H(x^\star)$, i.e., $H^\star$ is the Hessian matrix of f computed at $x^\star$. For any $x \in \mathbb{R}^d$, define $h : \mathbb{R}^d \mapsto \mathbb{R}^d$ as the residual of the first order Taylor's approximation of the gradient g at $x^\star$,

$$h(x) = g(x) - H^\star(x - x^\star), \tag{5}$$

for $x \in \mathbb{R}^d$. For each $\delta > 0$, define also

$$\overline{h}(\delta) = \sup_{x \in \mathbb{B}_{x^\star}(\delta)} \|h(x)\|, \tag{6}$$

where $B_x(\delta)$ denotes the closed Euclidean ball in $\mathbb{R}^d$ of radius $\delta \geq 0$, centered at x. The following result holds by a well-known corollary of Taylor's remainder theorem.

Lemma 1. *There holds $\overline{h}(\delta) = o(\delta)$, i.e., $\lim_{\delta \to 0} \frac{\overline{h}(\delta)}{\delta} = 0$.*

Remark 1. *Clearly, when f is quadratic, $H(x)$ is constant for all $x \in \mathbb{R}^d$ and equal to $H^\star$, implying $h(x) \equiv 0$ and also $\overline{h}(\delta) \equiv 0$.*

Remark 2. *Lemma 1 holds by the twice continuous differentiability of f. The quantity $\overline{h}(\delta)$ can be explicitly characterized if, in addition, it is assumed that the Hessian of function f is Lipschitz continuous, i.e., if $\|H(x) - H(y)\| \leq L_H \|x - y\|$, for all $x, y \in \mathbb{R}^d$, for some nonnegative constant L_H. It is easy to show that, in this case, we have $\|h(x)\| \leq L_H \|x - x^\star\|^2$, for any $x \in \mathbb{R}^d$. The latter implies a quadratic upper bound in δ on $\overline{h}(\delta)$, i.e., $\overline{h}(\delta) \leq L_H \delta^2$, for each $\delta \geq 0$.*

2.1 Distance to solution recursion

For analytical purposes, it is of interest to study the squared distance to solution of the current iterates $\|X_k - x^\star\|^2$. To characterize the evolution of this quantity, we use standard arguments that follow from strong convexity and Lipschitz smoothness:

$$\begin{aligned} \|X_{k+1} - x^\star\|^2 \leq \left(1 - 2\alpha_k\mu + 2\alpha_k^2 L^2\right) \|X_k - x^\star\|^2 + 2\alpha_k (X_k - x^\star)^\top Z_k \\ + 2\alpha_k^2 \|Z_k\|^2; \end{aligned} \tag{7}$$

details of the derivations can be found in Appendix A.

We introduce the function $\beta_k : \mathbb{R}^2 \mapsto \mathbb{R}$, defined by $\beta_k(u,v) = 1 - \alpha_k u + \alpha_k^2 v$. Similarly, for any two iteration indices $l \leq k$, we define $\beta_{k,l} : \mathbb{R}^2 \mapsto \mathbb{R}$ by $\beta_{k,l}(u,v) = \beta_k(u,v) \cdots \beta_l(u,v)$. The following technical lemma providing bounds on the product functions $\beta_{k,l}$ will be useful for the study of recursion (7) as well as other similar recursions that will emerge from the analysis.

Lemma 2. *Let l and k be two iteration indices such that $l < k$. For any nonnegative u, $v \in \mathbb{R}$, and $\alpha_k = a/(k+b)$, where $b \geq 1$, there holds:*

1. $\beta_{k,l}(u,v) \leq \left(\frac{l+b}{k+b+1}\right)^{au} e^{\frac{a^2 v}{l+b-1}}$;
2. *for each l such that $l+b \geq \frac{5au}{2}$, there holds* $\beta_{k,l}(u,v) \geq \left(\frac{l+b-1}{k+b}\right)^{au} e^{-\frac{a^2u^2}{l+b-1}}$;

The proof of Lemma 2 is given in Appendix A.

Finally, for each iteration index k, we denote by μ_k the Borel measure on $\mathbb{R}^d$ induced by X_k. Similarly, we denote by ν_k the Borel measure induced by $\|X_k - x^\star\|$.

2.2 Large deviations preliminaries

We next give a definition of the rate function and the large deviations principle.

Rate function I and the large deviations principle.

Definition 1 (Rate function I (Dembo et al. 1993))**.** *Function $I : \mathbb{R}^d \mapsto [0, +\infty]$ is called a rate* function *if it is lower semicontinuous, or, equivalently, if its level sets are closed. If, in addition, the level sets of I are compact (i.e., closed and bounded), then I is called a good rate function.*

Definition 2 (The large deviations principle (Dembo et al. 1993))**.** *Suppose that $I : \mathbb{R}^d \mapsto [0, +\infty]$ is lower semicontinuous. A sequence of measures μ_k on $\left(\mathbb{R}^d, \mathcal{B}\left(\mathbb{R}^d\right)\right)$, $k \geq 1$, is said to satisfy the large deviations principle (LDP) with rate function I if, for any measurable set $D \subseteq \mathbb{R}^d$, the following two conditions hold:*

1. $\limsup_{k\to+\infty} \dfrac{1}{k} \log \mu_k(D) \leq - \inf_{x \in \overline{D}} I(x)$;
2. $\liminf_{k\to+\infty} \dfrac{1}{k} \log \mu_k(D) \geq - \inf_{x \in D^{\circ}} I(x)$.

Log-moment generating functions of the noise Z_k and the iterates X_k. Following Assumption 3, we define the conditional LMGF of Z_k given the last iterate X_k.

Definition 3 (Conditional LMGF of Z_k)**.** *We denote by $\Lambda(\cdot;x)$ the log-moment generating function (LMGF) of Z_k given $X_k = x$,*

$$\Lambda(\lambda;x) := \log \mathbb{E}\left[e^{\lambda^\top Z_k}\middle| X_k = x\right], \text{ for } \lambda, x \in \mathbb{R}^d. \tag{8}$$

It will also be useful to define the conditional moment-generating function of $\|Z_k\|^2$, which we denote by $M(\cdot;x)$:

$$M(\nu;x) := \mathbb{E}\left[e^{\nu\|Z_k\|^2}\middle| X_k = x\right] \tag{9}$$

for $\nu \in \mathbb{R}$, $x \in \mathbb{R}^d$. By the inequality $e^x \leq x + e^{x^2}$, which holds for all $x \in \mathbb{R}$, we have $\mathbb{E}\left[e^{\lambda^\top Z_k}\middle| X_k\right] \leq \mathbb{E}[\lambda^\top Z_k|X_k] + \mathbb{E}\left[e^{(\lambda^\top Z_k)2}|X_k\right] \leq \mathbb{E}\left[e^{\|\lambda^2\|\|Z_k\|^2}\middle| X_k\right]$, where we used the Cauchy-Schwartz inequality, for the second term, and the fact that Z_k is zero-mean, for the first term. Thus,

$$\Lambda(\lambda;x) \leq \log M(\|\lambda^2\|, x) \tag{10}$$

for any realization x of X_k.

Lemma 3 lists properties of Λ that will be used in the paper.

Lemma 3 (Properties of Λ)**.** *For any given $x \in \mathbb{R}^d$ the following properties hold:*

1. *$\Lambda(\cdot;x)$ is convex and differentiable in the interior of its domain;*
2. *$\Lambda(0;x) = 0$ and $\nabla\Lambda(0;x) = \mathbb{E}[Z_k|X_k = x] = 0$;*
3. *$\Lambda(\lambda;x) \geq 0$, for each λ.*

Proof. Convexity and differentiability are general properties of log-moment generating functions (Dembo et al. 1993), as well as the zero value at the origin property and also that the gradient at the origin equals the mean vector; $\nabla\Lambda(0;x) = 0$ follows by the assumption that the noise is zero-mean, Assumption 3. The non-negativity from Part 3 follows by invoking convexity and exploiting the two properties from part 2, i.e., for any $x \in \mathbb{R}^d$: $\Lambda(\lambda;x) \geq \Lambda(0;x) + \nabla\Lambda(0;x)^\top\lambda = 0$. □

Example 1. *To illustrate the LMGF function Λ, we consider the case when, conditioned on an arbitrary realization $X_k = x$, the gradient noise Z_k is Gaussian, with mean vector equal to zero vector and covariance matrix $\Sigma(x)$. Using standard formula for the LMGF of a Gaussian multivariate, we have*

$$\Lambda(\lambda; x) = \frac{1}{2}\lambda^\top S(x)\lambda, \tag{11}$$

for $\lambda \in \mathbb{R}^d$. We note that when the gradient noise Z_k is independent of the current iterate X_k, the indices X_k in the preceding formula can be omitted, i.e., the expression for Λ simplifies to $\Lambda(\lambda; X_k) = \frac{1}{2}\lambda^\top S\lambda$, for all realizations X_k.

It will also be of interest to define the (unconditional) log-moment generating function of the iterates X_k.

Definition 4 (LMGF of $X_k - x^\star$). *We let Γ_k denote the (unconditional) moment generating function of X_k,*

$$\Gamma_k(\lambda) := \mathbb{E}\left[e^{\lambda^\top (X_k - x^\star)}\right], \tag{12}$$

for $\lambda \in \mathbb{R}^d$. The (unconditional) log-moment generating function of X_k is then given by $\log \Gamma_k$.

We assume that the initial iterate X_1 is deterministic[2]. Hence, Γ_1 is finite for all $\lambda \in \mathbb{R}^d$.

We assume that the family of functions $\Lambda(\cdot; x)$ satisfy the following regularity conditions.

Assumption 4 (Lipschitz continuity in x). *There exists a constant L_Λ such that for every $\lambda, x, y \in \mathbb{R}^d$, there holds:*

$$|\Lambda(\lambda; x) - \Lambda(\lambda; y)| \leq L_\Lambda \|\lambda\|^2 \|x - y\|. \tag{13}$$

Remark 3. *We note that Assumption 4 is trivially satisfied when the noise distribution does not depend on the current iterate. For another illustration, consider Gaussian random noise distribution from Example 1, for which we have:*

$$\Lambda(\lambda; x) - \Lambda(\lambda; y) = \frac{1}{2}\lambda^\top (S(x) - S(y))\lambda \tag{14}$$

$$\leq \frac{1}{2}\|\lambda\|^2 \|S(x) - S(y)\|. \tag{15}$$

[2]We note that this assumption can be relaxed to allow for random initial iterate; see Appendix D for details.

Comparing with the condition in (13)*, we see that* (13) *is satisfied when entries of the covariance matrix S, as functions of x, are Lipschitz continuous.*

The assumption below will be used for the proof of the main result of the paper, when the case of general convex functions is considered.

Assumption 5 (Sub-Gaussian noise)**.** *There exists a constant $C_1 > 0$ such that, for each λ, $x \in \mathbb{R}^d$*

$$\Lambda(\lambda; x) \le C_1 \frac{\|\lambda\|^2}{2}. \tag{16}$$

Remark 4. *Assumption 5 means that the gradient noise has "light tails," i.e., there exist positive constants c_1, c_2, such that the probability that the magnitude of the norm of the noise vector is above ϵ is upper bounded by $c_1\, e^{-c_2 \epsilon^2}$, for any $\epsilon > 0$. Clearly, a Gaussian zero-mean multivariate distribution satisfies this property, and also any noise distribution with compact support.*

This assumption also ensures that, for each given λ, the value of the variance "proxy" C_1 cannot grow without bound as the domain of iterates x enlarges. For a Gaussian distribution, this means that the variance, as a function of the current iterate should be uniformly bounded over the domain of the iterates, which is a typical assumption in related works.

We also use the following implications of Assumption 5.

Proposition 1. *1. There exists $C_2 > 0$ such that*

$$\mathbb{E}\left[\exp\left(\frac{\|Z_k^2\|}{C_2}\right) \middle| X_k\right] \le e. \tag{17}$$

2. For any $\nu \in [0, 1/C_2]$ there holds

$$M(\nu; X_k) \le \exp(\nu C_2). \tag{18}$$

Proof. The proof of part 1 can be derived by applying properties of sub-Gaussian random variables to $\|Z_k\|$; see, e.g., Proposition 2.5.2 in Vershynin 2018 and also Jin et al. 2019 for a treatment of sub-Gaussian random vectors.

To show part 2, fix $\nu \in [0, 1/C_2]$. By Hölder's inequality (applied for "p" $= 1/(\nu C_2) \ge 1$)

$$M(\nu; X_k) \le \left(\mathbb{E}\left[\exp\left(1/C_2 \|Z_k\|^2\right) \middle| X_k\right]\right)^{\nu C_2} \tag{19}$$
$$\le \exp(\nu C_2) \tag{20}$$

where in the second inequality we used part 1. $\square$

Remark 5. *When the distribution of Z_k is Gaussian, zero mean and with covariance matrix Σ, and independent of the current iterate, we have*

$$\Lambda(\lambda) = \frac{1}{2}\lambda^\top \Sigma \lambda \leq \frac{1}{2}\sigma_{\max}^2 \|\lambda\|^2, \tag{21}$$

where $\sigma_{\max}^2$ is the maximal eigenvalue of Σ. Comparing with Assumption 5, we see that condition (16) *holds with $C_1 = \sigma_{\max}^2$. It can also be shown that part 1. of Proposition 1 holds for $C_2 \geq 2\sigma_{\max}^2$.*

2.3 Key technical lemma

Definition 5. *The Fenchel-Legendre transform, or the conjugate, of a given function $\Psi : \mathbb{R}^d \mapsto \mathbb{R}$ is defined by*

$$I(x) = \sup_{\lambda \in \mathbb{R}^d} x^\top \lambda - \Psi(\lambda), \quad \text{for } x \in \mathbb{R}^d. \tag{22}$$

Lemma 4. *Let Ψ_k be a sequence of log-moment generating functions associated to a given sequence of measures $\mu_k : \mathcal{B}(\mathbb{R}^d) \mapsto [0, 1]$. Suppose that, for each $\lambda \in \mathbb{R}^d$, the following limit exists:*

$$\limsup_{k \to +\infty} \frac{1}{k} \Psi_k(k\lambda) \leq \Psi(\lambda). \tag{23}$$

If $\Psi(\lambda) < \infty$ for each $\lambda \in \mathbb{R}^d$, then the sequence μ_k satisfies the LDP upper bound with the rate function I equal to the Fenchel-Legendre transform of Ψ. If, in addition, (23) *holds as a limit and with equality, then the sequence of measures satisfies the LDP with rate function I.*

The second part of the lemma follows by the Gärtner-Ellis theorem. The first part can be proven by similar arguments as in the proof of the upper bound of the Gärtner-Ellis theorem; for details, see also the proof of Lemma 35 in (Bajovic 2022).

3 Large deviations rates for SGD iterates X_k

3.1 Large deviations rates for $\|X_k - x^\star\|$

To derive the main result – the large deviations rate function for the SGD sequence X_k, we first study large deviations properties of the sequence $\|X_k - x^\star\|$, $k = 1, 2, ...$ For the latter, we first exploit the idea from Harvey et al. 2019 to obtain a high probability bound for the (scaled) quantity $\|X_k - x^\star\|^2$, via its moment generating function. We then use this bound to derive a rate function (bound) for $\|X_k - x^\star\|$. Since our assumptions are distinct than those in Harvey et al. 2019 (e.g., the recursive form that we work with here contains factors that require special treatment than the one in Harvey et al. 2019, also we do not assume bounded noisy gradient, as is the case with the proof available in Harvey et al. 2019), we provide full proof details, see appendix.

Lemma 5. *For any k, there holds*

$$\mathbb{P}\left(\|X_k - x^\star\| \geq \delta\right) \leq e e^{-(k+k_0)B\delta^2}, \tag{24}$$

where $B = \min\{\frac{1}{k_0\|X_1 - x^\star\|}, \frac{2a\mu - 1}{4\max\{C_1, 2C_2\}a^2}\}$ *and* $k_0 = 4a^2L^2/(2a\mu - 1)$.

Remark 6. *The preceding theorem establishes a large deviations upper bound for the sequence of squared distance to solution iterates ξ_k, by exploiting noise sub-Gaussianity. By its nature, this result is a rough characterization of the large deviations rate function for the sequence X_k. In addition to being a result of independent interest, the utility consists in bounding the tails of distribution μ_k, as an enabling step towards deriving a fine, close to exact rate function for the SGD iterates X_k as the main contribution of this paper. The latter is the subject of the next section.*

3.2 Main result: large deviations rates for X_k

We now present our result for general convex functions satisfying assumptions from Section 2. The pillar of the analysis is the limit of the sequence of log-moment generating functions $\log \Gamma_k$ of the SGD iterates.

Lemma 6. *Suppose that Assumptions 1-5 hold and that the stepsize is given by $\alpha_k = a/(k + k_0)$. For any $\lambda \in \mathbb{R}^d$,*

$$\limsup_{k \to +\infty} \frac{1}{k} \log \Gamma_k(k\lambda) \leq \overline{\Psi}(\lambda) := \Psi^\star(\lambda) + r(\lambda), \tag{25}$$

where $\Psi^\star$ is defined by

$$\Psi^\star(\lambda) = \int_0^1 \Lambda(aQD(\theta)Q^\top \lambda; x^\star) d\theta, \tag{26}$$

where $H^\star = QDQ^\top$, $QQ^\top = I$, $D = \mathrm{diag}\{\rho_1, ..., d_n\}$, $D(\theta) = \mathrm{diag}\{\theta^{a\rho_1 - 1}, ..., \theta^{ad_n - 1}\}$, $r(\lambda) = \frac{4a^2\overline{\gamma}^2 L_\Lambda}{B^2}\|\lambda\|^4 + a\|\lambda\|\overline{h}\left(\frac{2\overline{\gamma}\|\lambda\|}{B}\right)$, *and* $\overline{\gamma} = \max\{1, \sqrt{(1 - a\mu)^2 + a^2(L^2 - \mu^2)}\}$.

The proof of Lemma 6 is given in section 5. Having the limit in (25), LDP upper bound follows by Lemma 4.

Theorem 1. *Suppose that Assumptions 1-5 hold and that the stepsize is given by* $\alpha_k = a/(k+k_0)$*. Then, the sequence of iterates* X_k *satisfies the LDP upper bound with rate function* $\overline{I}$ *given as the Fenchel-Legendre transform of* $\overline{\Psi}$ *from Lemma 6, i.e., for any closed set* F*:*

$$\limsup_{k\to+\infty} \frac{1}{k} \log \mathbb{P}\left(X_k \in F\right) \leq - \inf_{x+x^\star \in F} \overline{I}(x). \tag{27}$$

Remark 7. *The rate function* $\overline{I}$ *depends on the Hessian matrix at the solution,* $H(x^\star)$*. However, coarser exponential rate bounds can be obtained by uniformly bounding the eigenvalues of* $H(x^\star)$*, as by our assumptions they are all confined in the interval* $[\mu, L]$*. See Appendix [illegible] for details.*

3.3 Discussions and interpretations

3.3.1 Positivity of $\overline{I}$ and exponential decay

From the fact that $\Psi^\star$, $r \geq 0$, and that both $\Psi^\star$ and r are finite on $\mathbb{R}^d$, it can be shown that $\overline{I} \geq 0$ and that $\overline{I}$ is a good rate function. Specifically, $\overline{I}(0) = 0$ and $\overline{I}(x) > 0$ for any $x \neq 0$. Therefore, for any closed set F such that $x^\star \notin F$, we have

$$\inf_{x+x^\star \in F} \overline{I}(x) > 0, \tag{28}$$

that is, the exponent in (27) is strictly positive ensuring the exponential decay of the probabilities $\mathbb{P}\left(X_k \in F\right)$. To illustrate this in intuitive terms, we take as a special case the set $F = B^{\mathrm{c}}_{x^\star}(\delta)$, for some $\delta > 0$. Then, the event of interest becomes $\{X_k \in F\} = \{\|X_k - x^\star\| \geq \delta\}$. Thus, for any $\delta > 0$, Theorem 1 implies that

$$\limsup_{k\to+\infty} \frac{1}{k} \log \mathbb{P}\left(\|X_k - x^\star\| \geq \delta\right) \leq -R(\delta), \tag{29}$$

where $R(\delta) = \inf_{\|x\|\geq\delta} I(x) > 0$.

3.3.2 Remainder term r

Recalling Lemma 1, it is easy to see that $r(\lambda) = o(\|\lambda\|^2)$, i.e., $\lim_{\|\lambda\|\to 0} \frac{r(\lambda)}{\|\lambda\|^2} = 0$. Also, for a function f that has Lipschitz Hessian, see Remark 2, the residual function r behaves roughly as $\sim \|\lambda\|^3$.

Further, for the special case when f is quadratic, $\overline{h}(\delta) = 0$, and hence r contains only the first term, and thus $r(\lambda) \sim \|\lambda\|^4$. Similarly, when the noise distribution does not depend on the current iterate, we have that $L_\Lambda = 0$, and hence $r(\lambda) = o(\|\lambda\|^2)$. Finally, for the case when both of the preceding conditions hold, the residual term is zero at all points: $r \equiv 0$, and hence the rate function $\overline{I} = I^\star$, where $I^\star$ is the Fenchel-Legendre transform of $\Psi^\star$.

3.3.3 Small deviations regime

When high precision estimates are sought, or equivalently, for small δ in (29), the candidate values of $\overline{I}$ in the minimization are very close to 0. By the fact that the remainder term $r(\lambda) = o(\|\lambda^2\|)$, it can be shown that, in the small deviations regime, $\overline{I}$ is determined by $\Psi^\star$ only, i.e., $\overline{I} \approx I^\star$, and, also, its behaviour is dominantly characterized by the noise variance.

3.4 LDP for quadratic functions

In this section we provide the full LDP for the case when f is a quadratic function. The proof of Theorem 2 is given in Appendix E.

Theorem 2. *Suppose that the objective function f is quadratic, that Assumptions 2-3 hold, with the step size given by $\alpha_k = a/k$. Suppose also that the noise distribution does not depend on the current iterate and that it has a finite log-moment generating function Λ. Then, the sequence X_k satisfies the large deviations principle with the rate function $I^\star$ given as the conjugate of $\Psi^\star$ defined in (26), with $\Lambda(\cdot; x^\star)$ replaced by Λ.*

The rate function $I^\star$ depends on the distribution of Z_k and fully captures all moments of this distribution. In particular, for non-Gaussian distributions, it captures exactly the dependence not only on the variance, but also on higher order moments.

Remark 8. *We note that, in contrast with Theorem 1, for Theorem 2 the conditional distribution of Z_k can be arbitrary, as long as Λ is finite. In particular, it allows for distributions that are not light-tailed, such as Laplacian.*

Remark 9. *Recalling the discussion from subsection 3.3.2, we see that the upper bound rate function from Theorem 1 and the rate function from Theorem 2 match, hence showing that the bound in Theorem 1 is tight.*

4 Gaussian noise: analytical characterization of the rate function

If the noise Z_k has a Gaussian distribution with mean value zero and covariance matrix Σ, then $\Psi^\star$ is computed by

$$\Psi^\star(\lambda) = \frac{a^2}{2}\int_0^1 \lambda^\top Q D(\theta) Q^\top \Sigma Q D(\theta) Q^\top \lambda d\theta. \tag{30}$$

To simplify the notation, let $S = Q^\top \Sigma Q$, and $M(\theta) = D(\theta) S D(\theta)$. It is easy to verify that $M_{ij}(\theta) = S_{ij}\theta^{a(\rho_i+\rho_j)-2}$, for any $i, j = 1, ..., d$, and thus $\int_0^1 M_{ij}(\theta) d\theta = S_{ij}/(a(\rho_i + \rho_j) - 1)$. Hence, we obtain the following closed-form expression for $\Psi^\star$:

$$\Psi^\star(\lambda) = \frac{a^2}{2}\lambda^\top Q S^\star Q \lambda, \tag{31}$$

where $S^\star_{ij} = S_{ij}/(a(\rho_i + \rho_j) - 1)$, for $i, j = 1, ..., d$.

Recalling the Definition 5, it can be shown that the Fenchel-Legendre transform $I^\star$ of $\Psi^\star$ is given by

$$I^\star(z) = \frac{1}{2a^2} z^\top Q^\top S^{\star -1} Q z. \tag{32}$$

To obtain further information about the rate function $I^\star$, we consider the special case when the Hessian matrix $H^\star$ and the covariance matrix Σ share the same eigenspace (given by the columns of the matrix Q). Intuitively, the latter means that the orientation of the quadratic approximation of f at the origin is aligned with the gradient noise distribution in each of the axes. In this case, it follows that $S = Q^\top \Sigma Q$ is diagonal with $S_{ii} = \sigma^2_{ii}$, where σ^2_{ii} is the i-th eigenvalue of Σ (i.e., the eigenvalue of Σ corresponding to its eigenvector given by the i-th column of matrix Q). It follows that $S^\star$ is also diagonal with $S^\star_{ii} = \sigma^2_{ii}/(2a\rho_i - 1)$. Thus, the following neat expression for the rate function $I^\star$ emerges:

$$I(z) = \frac{1}{2a^2} z^\top Q^\top \mathrm{diag}\left(\frac{2a\rho_1 - 1}{\sigma^2_{11}}, ..., \frac{2a\rho_d - 1}{\sigma^2_{dd}}\right) Q z. \tag{33}$$

4.1 Decay rates with l_2 balls

We consider the case when in the large deviations event of interest $\{X_k \in F\}$ the set F is given as the complement of an l_2 ball around the solution $x^\star$:

$F = B^{\mathrm{c}}_{x^\star}(\delta)$, i.e., $\{X_k \in F\} = \{\|X_k - x^\star\| \geq \delta\|$. Assuming that the residual is zero (see the result for quadratic functions in Section 3.4), by Theorem 1, we have

$$\limsup_{k\to+\infty} \frac{1}{k} \log \mathbb{P}\left(\|X_k - x^\star\| \geq \delta\right) \leq \inf_{\|z\|\geq\delta} I(z) =: \mathbf{I}(B^{\mathrm{c}}_{x^\star}(\delta)). \tag{34}$$

For the Gaussian noise assumed in this section, we have:

$$\begin{aligned}\mathbf{I}(B^{\mathrm{c}}_{x^\star}(\delta)) &= \inf_{\|z\|\geq\delta} \frac{1}{2a^2} z^\top Q^\top S^{\star-1} Q z \\ &= \frac{\delta^2}{2a^2} \inf_{\|w\|\geq 1} w^\top Q^\top S^{\star-1} Q w \\ &= \frac{\delta^2}{2a^2} \frac{1}{\lambda_{\max}(S^\star)},\end{aligned} \tag{35}$$

where $\lambda_{\max}(S^\star)$ is the largest eigenvalue of the matrix $S^\star$. Hence, to find the value of the exponent $\mathbf{I}$ for any given ball-shaped set, it suffices to find (once) the maximal eigenvalue of $S^\star$, and the exponent $\mathbf{I}$ would be easily computed by the quadratic function (35).

We close the analysis with a particularly elegant solution for the special case when $H^\star$ and Σ are axes aligned. As detailed at the beginning of the section, in the latter case, $S^\star$ is diagonal, with $S^\star_{ii} = \sigma^2_{ii}/(2a\rho_i - 1)$, and the rate function is given by (33). Thus, to find the maximal eigenvalue of $S^\star$ reduces to finding the index i for which $\frac{\sigma^2_{ii}}{2a\rho_i - 1}$ is highest, or, equivalently, $\frac{2a\rho_i-1}{\sigma^2_{ii}}$ the lowest, which then yields:

$$\mathbf{I}(B^{\mathrm{c}}_{x^\star}(\delta)) = \frac{\delta^2}{2a^2} \min\{\frac{2a\rho_i - 1}{\sigma^2_{ii}} : i = 1, ...d\}, \tag{36}$$

where, we recall, ρ_i is the i-th eigenvalue of $H^\star$. What the expression above is saying is that, in order to find the exponential decay rate for an l_2 ball, we should search for the direction i in which the value $\frac{\sigma^2_{ii}}{2\rho_i - 1}$ is highest. In a sense, the latter quantity can be thought of as the effective noise variance, capturing the interplay between the noise distribution and the shape of the function at the solution. Specifically, if along the direction where the noise variance is highest, say $i^\star$, the function has a high curvature (i.e., large $\rho_{i^\star}$), this will effectively alleviate the effects of noise and increase the rate function, in comparison to the case when the curvature along i is lower, and therefore result in faster convergence.

Finally, when the noise is isotropic, i.e., such that $\sigma_{ii}^2 = \sigma^2$, for all i, exploiting the fact that the spectrum of $H^\star$ lies inside the interval $[\mu, L]$, the rate function is found by:

$$\mathbf{I}(B^{\mathrm{c}}_{x^\star}(\delta)) = \frac{\delta^2}{2a^2}\frac{2a\mu - 1}{\sigma^2}. \tag{37}$$

4.2 Comparison with the rate from Lemma 5

We now compare the rate function bounds obtained from Lemma 5 and Theorem 1. To gain deeper insights, we will assume that the residual term r equals zero (compare with Section 3.4). We also assume that the noise is Gaussian and axes-aligned with the matrix $H^\star$ (see the preceding subsection). The exponent B from 5 can be upper bounded by[3]

$$B \leq \frac{2a\mu - 1}{4{\sigma_{\max}}^2 a^2},$$

where we exploited the fact that, for Gaussian noise, $C_1 = \sigma^2_{\max}$, see Remark 5. Hence, for an l_2 ball of radius δ, the exponent that Lemma 5 provides is bounded by

$$B\delta^2 \leq \frac{\delta^2}{4a^2}\frac{2a\mu - 1}{\sigma^2_{\max}}. \tag{38}$$

The counterpart obtained from Theorem 1 is given by expression (36). To show direct comparison with (38), we further upper bound this value by decoupling the minimization over i :

$$\begin{aligned}
\mathbf{I}(B^{\mathrm{c}}_{x^\star}(\delta)) &= \frac{\delta^2}{2a^2}\min\{\frac{2a\rho_i - 1}{\sigma_{ii}^2} : i = 1, ...d\} \\
&\geq \frac{\delta^2}{2a^2}\frac{\min\{2a\rho_i - 1 : i = 1, .., d\}}{\max\{\sigma_{ii}^2 : i = 1, .., d\}} \\
&= \frac{\delta^2}{2a^2}\frac{2a\mu - 1}{\sigma^2_{\max}}.
\end{aligned} \tag{39}$$

Comparing with (38) (and ignoring the scaling constant 2), the following important point can be noted: on intuitive level, the derivation of the rate B is equivalent to that of decoupling the effects of the noise distribution and

[3]The dependence in B on X_1 in Lemma 5 seems to be an artifact of the conducted proof method, rather than an essential property of the exponential rate that Lemma 5 pursues. Hence, for unbiased comparison, we omit this factor in the analysis of the rate B.

the shape of the function f at the origin with the rate $I^\star$. Hence, in contrast with $I^\star$, the rate B is oblivious to the interplay between these two quantities – from a purely technical perspective, this distinction is a consequence of relying on recursions on the iterates' distance to solution, $\|X_k - x^\star\|$, as opposed to working directly with the iterates X_k, as is the case in the proof of Theorem 1.

5 Proof of Lemma 6

This section provides the main elements of the proof of the limit in (25); the proofs of omitted results can be found in Appendix C. Fix $\lambda \in \mathbb{R}^d$. Fix $k \geq 1$. Define $\eta_l = B_{k,l}\eta_k$, $B_{k,l} = (I - \alpha_l H^\star)\cdots(I - \alpha_k H^\star)$, $\eta_k = k\lambda$. By Lemma 2,

$$\|\eta_l\| \leq k\left(\frac{l+k_0}{k+k_0+1}\right)^{a\mu}\|\lambda\| \leq (l+k_0)\|\lambda\|. \tag{40}$$

For an arbitrary $l \leq k$, there holds

$$\begin{aligned}\Gamma_{l+1}(\eta_l) &= \mathbb{E}\left[\exp\left(\eta_l^\top(X_{l+1} - x^\star)\right)\right]\\ &= \mathbb{E}\left[\mathbb{E}\left[\exp\left(\eta_l^\top(X_l - \alpha_l g(X_l) + \alpha_l Z_l - x^\star)\right)|X_l\right]\right]\\ &= \mathbb{E}\left[\exp\left(\Lambda(\alpha_l\eta_l; X_l) + \eta_l^\top(X_l - \alpha_l g(X_l) - x^\star)\right)\right]\\ &= \int_{x\in\mathbb{R}^d}\Gamma_{l+1|l}(\eta_l; x)\mu_l(dx),\end{aligned} \tag{41}$$

where $\Gamma_{l+1|l}(\cdot; x)$ denotes the conditional moment generating function of X_{l+1}, given $X_l = x$. We now fix $\delta > 0$ (the exact value to be chosen later) and split the analysis in two cases: 1) $\mathcal{A}_{l,\delta} = \{X_l \in B_{x^\star}(\delta)\}$; and 2) $\mathcal{A}_{l,\delta}^{\mathrm{c}} = \{X_l \in B_{x^\star}^{\mathrm{c}}(\delta)\}$.

Introduce

$$\begin{aligned}\Gamma_{l+1|\mathcal{A}_{l,\delta}}(\eta_l) &:= \mathbb{E}\left[1_{\|X_l - x^\star\|\leq\delta}\Gamma_{l+1|l}(\eta_l; X_l)\right]\\ &= \int_{\|x-x^\star\|\leq\delta}\Gamma_{l+1|l}(\eta_l; x)\mu_l(dx)\end{aligned} \tag{42}$$

$$\begin{aligned}\Gamma_{l+1|\mathcal{A}_{l,\delta}^{\mathrm{c}}}(\eta_l) &:= \mathbb{E}\left[1_{\|X_l - x^\star\|>\delta}\Gamma_{l+1|l}(\eta_l; X_l)\right]\\ &= \int_{\|x-x^\star\|>\delta}\Gamma_{l+1|l}(\eta_l; x)\mu_l(dx);\end{aligned} \tag{43}$$

note that

$$\Gamma_{l+1}(\eta_l) = \Gamma_{l+1|A_{l,\delta}}(\eta_l) + \Gamma_{l+1|A_{l,\delta}^{\mathrm{c}}}(\eta_l). \tag{44}$$

Case 1: $x \in \mathcal{A}_{l,\delta}$. Fix $x \in \mathbb{R}^d$ such that $\|x\| \leq \delta$. We have:

$$\begin{aligned}
\Gamma_{l+1|l}(\eta_l; x) &= \exp\Big(\Lambda(\alpha_l \eta_l; x) + \eta_l^\top (x - \alpha_l g(x) - x^\star)\Big) \\
&= \exp\Big(\Lambda(\alpha_l \eta_l; x) + \eta_l^\top ((I - \alpha_l H^\star)(x - x^\star) - \alpha_l h(x)\Big) \\
&\leq \exp\left(\Lambda(\alpha_l \eta_l; x^\star) + L_\Lambda \|\eta_l^2\| \|x - x^\star\| + \alpha_l \|\eta_l\| \|h(x)\|\right) \\
&\quad \times \exp\Big(\eta_l^\top ((I - \alpha_l H^\star)(x - x^\star)\Big) \qquad (45) \\
&\leq \exp\Big(\Lambda(\alpha_l \eta_l; x^\star) + L_\Lambda \alpha_l^2 \|\eta_l^2\| \delta^2 + \alpha_l \|\eta_l\| \overline{h}(\delta) + \eta_{l-1}^\top (x - x^\star)\Big), \\
&\qquad (46)
\end{aligned}$$

where in (45) we used Lipschitz continuity of Λ in [illegible] Assumption 4, and in (46) we used the fact that $\|x - x^\star\| \leq \delta$. It foll[illegible]s th[illegible]

$$\Gamma_{l+1|\mathcal{A}_\delta}(\eta_l) \leq \exp\left(\Lambda(\alpha_l \eta_l; x^\star) \text{[illegible]}_0(\lambda, \delta)\right) \Gamma_l(\eta_{l-\text{[illegible]}} \qquad (47)$$

where $r_0(\lambda, \delta) = L_\Lambda a^2 \|\lambda\|^2 \delta^2 + a\|\lambda\| \overline{h}(\delta)$.

Case 2: $x \in \mathcal{A}_{l,\delta}^c$. By strong co[illegible]ity and [illegible]ipschitz smoothness of f in Assumption 1, for each $l \geq 1$, the f[illegible]owin[illegible] holds[illegible]

$$\begin{aligned}
\|X\text{[illegible]} g(\text{[illegible]}) - x\text{[illegible]} &\leq \gamma_l \|X_l - x^\star\|, \qquad (48) \\
\text{[illegible]} &\leq \overline{\gamma} \|X_l - x^\star\| \qquad (49)
\end{aligned}$$

where $\gamma_l = ($[illegible]$ + $[illegible]$L^2)^{1/2}$, see Appendix A for the proof, and $\overline{\gamma} = \sup\{\gamma_l : l$ [illegible]$1, 2, ...\}$; [illegible]s ea[illegible] to verify that $\overline{\gamma} = \max\{1, \sqrt{(1 - a\mu)^2 + a^2(L^2 - a^2)}\}$.

For an [illegible]bitrary x [illegible] $\mathbb{R}^d$, we have:

$$\begin{aligned}
\Gamma_{l+1|l}(\eta_l; \text{[illegible]} &\text{[illegible]}\text{xp}\Big(\Lambda(\alpha_l \eta_l; x) + \eta_l^\top (x - \alpha_l g(x) - x^\star)\Big) \\
&\leq \exp\left(\frac{C_1 \alpha_l^2 \|\eta_l\|^2}{2}\right) \exp\left(\overline{\gamma} \|\eta_l\| \|x - x^\star\|\right), \qquad (50) \\
&\leq \exp\left(\frac{C_1 a^2 \|\lambda\|^2}{2}\right) \exp\left(\overline{\gamma}(l + k_0) \|\lambda\| \|x - x^\star\|\right), \qquad (51)
\end{aligned}$$

where in (50) we used the assumption that Z_k is sub-Gaussian, Assumption 5, for the first term, together with (48) and Cauchy-Schwartz, for the second term, while in (51) we exploited (40). Recalling the induced measure ν_l, we now have

$$\Gamma_{l+1|\mathcal{A}_{l,\delta}^c}(\eta_l) \leq \exp\left(\frac{C_1 a^2 \|\lambda\|^2}{2}\right) \int_{z \geq \delta} e^{(l+k_0)\overline{\gamma}\|\lambda\| z} \nu_l(dz). \qquad (52)$$

The idea of analysing the "tail" term $\Gamma_{l+1|\mathcal{A}_{l,\delta}^{c}}(\eta_l)$ is the following: by Theorem 5, we know that the probability density ν_l at a given point z behaves roughly as $e^{-(l+k_0)Bz^2}$. If δ is sufficiently large, then, for all $z \geq \delta$, the negative exponential rate of the measure $\nu_l(z)$ is in absolute terms higher than the exponent $(l+k_0)\overline{\gamma}\|\lambda\|z$. Integrating by parts, we obtain that for $\delta = \frac{2\overline{\gamma}\|\lambda\|}{B}$, the integral on the right hand-side of (52) is upper bounded by a constant K. Thus:

$$\Gamma_{l+1|\mathcal{A}_{l,\delta}^{c}}(\eta_l) \leq K \exp\left(\frac{C_1 a^2 \|\lambda\|^2}{2}\right). \tag{53}$$

Combining with (47) and recalling (44),

$$\Gamma_{l+1}(\eta_l) \leq \exp\left(\Lambda(\alpha_l \eta_l; x^\star) + r(\lambda)\right) \Gamma_l(\eta_{l-1}) + K \exp\left(\frac{C_1 a^2 \|\lambda\|^2}{2}\right),$$

where $r(\lambda) = r_0\left(\lambda, \frac{2\overline{\gamma}\|\lambda\|}{B}\right)$. Iterating the preceding recursion, where we exploit the nonnegativity of Λ, property 3. from Lemma 3, we obtain:

$$\begin{aligned}\Gamma_{k+1}(k\lambda) &\leq \exp\left(\sum_{l=1}^{k}\left(\Lambda(\alpha_l \eta_l; x^\star) + r(\lambda)\right)\right)\Gamma_1(\alpha_1 \eta_1)\\ &+ K \exp\left(\frac{C_1 a^2 \|\lambda\|^2}{2}\right) \sum_{l=1}^{k} e^{\sum_{j=l}^{k}(\Lambda(\alpha_j \eta_j; x^\star) + r(\lambda))}\\ &\leq (k+1) K \exp\left(\frac{C_1 a^2 \|\lambda\|^2}{2} + a\|\lambda\|\|X_1 - x^\star\|\right)\\ &\quad \times e^{\sum_{l=1}^{k}(\Lambda(\alpha_l \eta_l; x^\star) + r(\lambda))}.\end{aligned} \tag{54}$$

Taking the limit, dividing by k, and taking the lim sup

$$\begin{aligned}&\limsup_{k \to +\infty} \frac{1}{k} \log \Gamma_{k+1}(k\lambda) \leq\\ &\qquad r(\lambda) + \limsup_{k \to +\infty} \frac{1}{k} \sum_{l=1}^{k} \Lambda(\alpha_l \eta_l; x^\star).\end{aligned} \tag{55}$$

Finally, it can be shown that

$$\lim_{k \to +\infty} \frac{1}{k} \sum_{l=1}^{k} \Lambda(\alpha_l \eta_l; x^\star) = \int_0^1 \Lambda(aQD(\theta)Q^\top \lambda; x^\star) d\theta. \tag{56}$$

The proof of (56) is provided in Appendix C. This completes the proof of Lemma 6.

6 Conclusions

We developed large deviations analysis for the stochastic gradient descent (SGD) method, when the objective function is smooth and strongly convex. For (strongly convex) quadratic costs, we establish the full large deviations principle. That is, we derive the exact exponential rate of decay of the probability that the iterate sequence generated by SGD stays within an arbitrary set that is away from the problem solution. This is achieved for a very general class of gradient noises, that may be iteration-dependent and are required to have a finite log-moment generating function. For generic costs, we derive a tight large deviations upper bound that, up to higher order terms, matches the exact rate derived for the quadratics.

References

Bahadur, R. R. (1960). "On the Asymptotic Efficiency of Tests and Estimates". In: *Sankhya: The Indian Journal of Statistics, 1933-1960* 22.3/4, pp. 229–252. ISSN: 00364452. URL: `http://www.jstor.org/stable/25048458`.

Bajovic, Dragana (2022). *Inaccuracy rates for distributed inference over random networks with applications to social learning.* DOI: `10.48550/ARXIV.2208.05236`. URL: `https://arxiv.org/abs/2208.05236`.

Bajović, D., D. Jakovetić, J. M. F. Moura, J. Xavier, and B. Sinopoli (Nov. 2012). "Large Deviations Performance of Consensus+Innovations Distributed Detection with Non-Gaussian Observations". In: *IEEE Transactions on Signal Processing* 60.11, pp. 5987–6002.

Bucklew, J. A. (1990). *Large Deviations Techniques in Decision, Simulation and Estimation.* New York: Wiley.

D. Bajović, D. Jakovetić, J. Xavier, B. Sinopoli, and J. M. F. Moura (Sept. 2011). "Distributed Detection via Gaussian Running Consensus: Large Deviations Asymptotic Analysis". In: *IEEE Transactions on Signal Processing* 59.9, pp. 4381–4396.

Davis, Damek, Dmitriy Drusvyatskiy, Lin Xiao, and Junyu Zhang (2021). "From Low Probability to High Confidence in Stochastic Convex Optimization." In: *J. Mach. Learn. Res.* 22, pp. 49–1.

Dembo, A. and O. Zeitouni (1993). *Large Deviations Techniques and Applications.* Boston, MA: Jones and Barlett.

Ghadimi, S. and G. Lan (2012). “Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization I: A generic algorithmic framework”. In: *SIAM J. Optim.* 22.4, pp. 1469–1492.

— (2013). “Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization, II: Shrinking procedures and optimal algorithms”. In: *SIAM J. Optim.* 23.4, pp. 2061–2089.

Gorbunov, Eduard, Marina Danilova, and Alexander Gasnikov (2020). “Stochastic optimization with heavy-tailed noise via accelerated gradient clipping”. In: *arXiv preprint arXiv:2005.10785.*

Gorbunov, Eduard, Filip Hanzely, and Peter Richtárik (2020). “A unified theory of SGD: Variance reduction, sampling, quantization and coordinate descent”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 680–690.

Harvey, Nicholas J. A., Christopher Liaw, Yaniv Plan, and Sikander Randhawa (2019). “Tight analyses for non-smooth stochastic gradient descent”. In: *32nd Annual Conference on Learning Theory*. Vol. 99. PMLR, pp. 1–35.

Hu, Ping, Virginia Bordignon, Stefan Vlaski, and Ali H. Sayed (2022). *Optimal Aggregation Strategies for Social Learning over Graphs*. DOI: 10.48550/ARXIV.2203.07065. URL: https://arxiv.org/abs/2203.07065.

Hu, Wenqing, Chris Junchi Li, Lei Li, and Jian-Guo Liu (2019). “On the diffusion approximation of nonconvex stochastic gradient descent”. In: *Annals of Mathematical Sciences and Applications* 4.1.

Jin, Chi, Praneeth Netrapalli, Rong Ge, Sham M. Kakade, and Michael I. Jordan (2019). *A Short Note on Concentration Inequalities for Random Vectors with SubGaussian Norm*. DOI: 10.48550/ARXIV.1902.03736. URL: https://arxiv.org/abs/1902.03736.

Juditsky, A., A. Nazin, A. Nemirovsky, and A. Tsybakov (2019). “Algorithms of robust stochastic optimization based on mirror descent method”. In: *arXiv:1907.02707.*

Lei, Lihua and Michael I Jordan (2020). “On the adaptivity of stochastic gradient-based optimization”. In: *SIAM Journal on Optimization* 30.2, pp. 1473–1500.

Marano, S. and A. H. Sayed (Oct. 2019). “Detection under one-bit messaging over adaptive networks”. In: *IEEE Trans. Information Theory* 65.10, pp. 6519–6538.

Matta, V., P. Braca, S. Marano, and A. H. Sayed (Aug. 2016a). “Diffusion-based adaptive distributed detection: Steady-state performance in the slow adaptation regime”. In: *IEEE Trans. Information Theory* 62.8, pp. 4710–4732.

Matta, V., P. Braca, S. Marano, and A. H. Sayed (Dec. 2016b). “Distributed detection over adaptive networks: Refined asymptotics and the role of connectivity”. In: *IEEE Trans. Signal and Information Processing over Networks* 2.4, pp. 442–460.

Niu, Feng, Benjamin Recht, Christopher Ré, and Stephen J Wright (2011). “Hogwild!: A lock-free approach to parallelizing stochastic gradient descent”. In: *arXiv preprint arXiv:1106.5730.*

Shwartz, Adam and Alan Weiss (1995). *Large Deviations for Performance Analysis: Queues, Communications, and Computing.* New York: Chapman and Hall.

T. M. Cover and J. A. Thomas (1991). *Elements of Information Theory.* New York: John Wiley and Sons.

Touchette, Hugo (2009). “The large deviation approach to statistical mechanics”. In: *Physics Reports* 478.1, pp. 1–69. ISSN: 0370-1573.

Vershynin, Roman (2018). *High-Dimensional Probability: An Introduction with Applications in Data Science.* Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press. DOI: `10.1017/9781108231596`.

Woodroofe, Michael (1972). “Normal Approximation and Large Deviations for the Robbins-Monro Process”. In: *Z. Wahrscheinlichkeitstheorie verw. Geb.* 21, pp. 329–33[illegible].

Appendix A.

Proof of recursion 7. For any $k \geq 1$, we have:

$$\begin{aligned}
\|X_{k+1} - x^\star\| &= \|X_k - \alpha_k g(X_k) + \alpha_k Z_k - x^\star\| \\
&= \|X_k - x^\star\|^2 - 2\alpha_k (X_k - x^\star)^\top (g(X_k) - Z_k) \\
&+ \alpha_k^2 \|g(X_k) - Z_k\|^2 \\
&\leq (1 - 2\alpha_k \mu)\xi_k + 2\alpha_k (X_k - x^\star)^\top Z_k \\
&+ 2\alpha_k^2 \|g(X_k)\| + 2\alpha_k^2 \|Z_k\|^2 \\
&\leq \left(1 - 2\alpha_k \mu + 2\alpha_k^2 L^2\right) \xi_k + 2\alpha_k (X_k - x^\star)^\top Z_k \\
&+ 2\alpha_k^2 \|Z_k\|^2, \qquad (57)
\end{aligned}$$

where the first inequality follows from the strong convexity of f, Assumption 1, and the fact that, for $a, b \in \mathbb{R}^d$, $\|a - b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$, and the second inequality follows from the Lipschitz smoothness of f, Assumption 1. □

Proof of Lemma 2. Fix l and k where $1 \leq l \leq k$. Fix u, $v \geq 0$. From the upper and the lower Darboux sum for the logarithmic function applied to the interval $[l, k]$, we obtain:

$$\log \frac{k+1}{l} \leq \frac{1}{l} + \ldots + \frac{1}{k} \leq \log \frac{k}{l-1}. \qquad (58)$$

For the 2-sum we use the following simple bound $1/l^2 \leq 1/(l(l-1)) = 1/(l-1) - 1/l$ to obtain:

$$\frac{1}{l^2} + \ldots + \frac{1}{k^2} \leq \frac{1}{l-1} - \frac{1}{l} + \ldots + \frac{1}{k-1} - \frac{1}{k} \leq \frac{1}{l-1}. \qquad (59)$$

To prove part 1, we use that $1 + x \leq e^x$ applied to each of the terms in the product $\beta_{k,l}$, together with the left hand-side inequality of (58) and the right hand-side inequality of (59):

$$\begin{aligned}
\beta_{k,l}(u, v) &\leq e^{-au \sum_{j=l}^{k} \frac{1}{j+b} + a^2 v \sum_{j=l}^{k} \frac{1}{(j+b)^2}} \\
&\leq e^{-au \log\left(\frac{k+b+1}{l+b}\right) + \frac{a^2 v}{l+b-1}} \\
&= \left(\frac{l+b}{k+b+1}\right)^{au} e^{\frac{a^2 v}{l+b-1}}. \qquad (60)
\end{aligned}$$

To prove part 2, we first note that, since $v \geq 0$, there holds $\beta_{k,l}(u,v) \geq \beta_{k,l}(u,0)$, i.e., $\beta_{k,l}(u,v) \geq (1-\alpha_k u) \cdots (1-\alpha_l u)$. We now use that, for $x \leq \frac{2}{5}$, $1 - x \geq e^{-x-x^2}$:

$$\begin{aligned}\beta_{k,l}(u,v) &\geq e^{-au\sum_{j=l}^{k}\frac{1}{j+b} - a^2u^2\sum_{j=l}^{k}\frac{1}{(j+b)^2}}\\ &\geq \left(\frac{l+b-1}{k+b}\right)^{au} e^{-\frac{a^2u^2}{l+b-1}}.\end{aligned} \tag{61}$$

This completes the proof of the lemma. □

Proof of (48). Here we prove an alternative recursion on $\|X_k - x^\star\|$, used within the proof of Lemma 6. Specifically, we show that, for any k,

$$\|X_{k+1} - x^\star\| \leq \gamma_k \|X_k - x^\star\| + \alpha_k \|Z_k\|, \tag{62}$$

where, we recall, $\gamma_k = \left(1 - 2\alpha_k\mu + \alpha_k^2 L^2\right)^{1/2}$.

From the triangle inequality applied to the Euclidean norm,

$$\begin{aligned}\|X_{k+1} - x^\star\| &= \|X_k - \alpha_k g(X_k) + \alpha_k Z_k - x^\star\|\\ &\leq \|X_k - \alpha_k g(X_k) - x^\star\| + \alpha_k\|Z_k\|\end{aligned} \tag{63}$$

Exploiting L-smoothness and μ- convexity of f for the second term:

$$\begin{aligned}&\|X_k - \alpha_k g(X_k) - x^\star\|^2 \leq\\ &\|X_k - x^\star\|^2 - 2\alpha_k(X_k - x^\star)^\top g(X_k) + \alpha_k^2\|g(X_k)\|\\ &\leq \|X_k - x^\star\|^2 - 2\alpha_k\mu\|X_k - x^\star\|^2 + \alpha_k^2L^2\|X_k - x^\star\|^2\\ &= \gamma_k^2\|X_k - x^\star\|^2.\end{aligned} \tag{64}$$

Taking the square root and replacing in (63) yields (62). □

Appendix B

Proof of Lemma 5. First, we transform the recursion in (7) by defining $Y_{k+1} = (k + k_0)\|X_{k+1} - x^\star\|^2$, to obtain:

$$Y_{k+1} \leq a_k Y_k - b_k\sqrt{k+k_0-1}(X_k - x^\star)^\top Z_k + c_k\|Z_k\|^2, \tag{65}$$

where

$$a_k = \frac{k+k_0}{k+k_0-1}(1 - 2\alpha_k\mu + 2\alpha_k^2L^2) \tag{66}$$

$$b_k = \frac{a}{\sqrt{k+k_0-1}} \tag{67}$$

$$c_k = \frac{a^2}{k+k_0}. \tag{68}$$

The key technical result behind Lemma 5 is the following upper bound on the tail probability of the Y_k iterates:

$$\mathbb{P}\left(Y_k \geq \epsilon\right) \leq e e^{-B\epsilon}, \tag{69}$$

which holds for each $k \geq 1$, and $\epsilon \geq 0$. The result of Lemma 5 directly follows from (69) by taking $\epsilon_k = k\delta^2$, for each k.

Thus, in the remainder of the proof we focus on proving (24). It can be easily verified that, for each k,

$$a_k = 1 - \frac{2a\mu - 1}{k + k_0 - 1}\left(1 - \frac{2a^2L^2}{(2a\mu - 1)(k + k_0 - 1)}\right). \tag{70}$$

Recalling Assumption 2 and the value of k_0, we see that the above quantity is smaller than 1 for each k.

Denote by Φ_k the moment generating function of Y_k, and by $\Phi_{k+1|k}(\cdot; X_k)$ the moment generating function of Y_k conditioned on X_k :

$$\Phi_k(\nu) := \mathbb{E}\left[\exp(\nu Y_k)\right] \tag{71}$$

$$\Phi_{k+1|k}(\nu; X_k) := \mathbb{E}\left[\exp(\nu Y_k) | X_k\right], \tag{72}$$

for $\nu \in \mathbb{R}$; note that $\Phi_{k+1}(\nu) = \mathbb{E}\left[\Phi_{k+1|k}(\nu; X_k)\right]$, for each $\nu \in \mathbb{R}$. From the recursion (7), we have:

$$\begin{aligned}
&\Phi_{k+1|k}(\nu; X_k) = \\
&\exp(a_k\nu Y_k)\mathbb{E}\left[\exp(-b_k\nu\sqrt{k + k_0 - 1}(X_k - x^\star)^\top Z_k + c_k\|Z_k\|^2)\Big| X_k\right] \\
&\leq \exp(a_k\nu Y_k)\left(\mathbb{E}\left[\exp(-2b_k\nu\sqrt{k + k_0 - 1}(X_k - x^\star)^\top Z_k)\Big| X_k\right]\right)^{1/2} \times \\
&\quad \left(\mathbb{E}\left[\exp(2c_k\nu\|Z_k\|^2)\big| X_k\right]\right)^{1/2} \\
&\leq \exp(a_k\nu Y_k)\, \exp(2b_k^2\nu^2 Y_k)\, \left(\mathbb{E}\left[\exp(2c_k\nu\|Z_k\|^2)\big| X_k\right]\right)^{1/2}
\end{aligned} \tag{73}$$

Recalling (2), the last term is finite for $\nu \leq 1/(2a^2C_2) =: B_0$, and for such ν, the corresponding value is equal to $\exp(C_2c_k\nu)$. Thus, for each $\nu \leq B_0$,

$$\Phi_{k+1|k}(\nu; X_k) \leq \exp(\nu(a_k + 2b_k^2\nu)Y_k + C_2c_k\nu). \tag{74}$$

It is easy to see that $B \leq B_0$. Consider $\nu \leq B$. Taking the expectation on both sides of (74), the following recursive inequality on Φ_k is obtained for any $\nu \leq B$ and any $k \geq 1$:

$$\Phi_{k+1}(\nu) \leq \Phi((a_k + 2b_k^2B)\nu)\exp(C_2c_k\nu). \tag{75}$$

From this point, the proof proceeds similarly as in Harvey et al. 2019, i.e., by induction, and using $k = 1$ as the base, it can be shown that, for each $\nu \leq B$,

$$\Phi_k(\nu) \leq e^{\frac{\nu}{B}}. \tag{76}$$

By exponential Markov, from (76), for each $\nu \leq B$,

$$\mathbb{P}\left(Y_k \geq \epsilon\right) \leq \mathbb{E}\left[\exp \nu Y_k e^{-\nu\epsilon}\right]. \tag{77}$$

Taking $\nu = B$ yields the desired result. □

Appendix C.

Proof of (56) . Introduce step-wise constant functions $s_k : [0,1] \mapsto \mathbb{R}$, defined by

$$s_k(\theta) = \left\{ \begin{array}{cc} \Lambda(\alpha_l \eta_l; x^\star), & \text{for } \frac{l-1}{k} < \theta \leq \frac{l}{k} \\ 0, & \text{for } \theta = 0 \end{array} \right. . \tag{78}$$

It is easy to verify that the integral of s_k over $[0,1]$ equals the desired sum in the right hand-side of (55), i.e.,

$$\int_0^1 s_k(\theta) d\theta = \frac{1}{k}\sum_{l=1}^{k} \Lambda(\alpha_l \eta_l; x^\star). \tag{79}$$

We next show that

$$\lim_{k\to+\infty} s_k(\theta) = \Lambda(aQD(\theta)Q^\top \lambda; x^\star), \tag{80}$$

where $D(\theta)$ is as defined in the claim of the theorem. To show the preceding limit, note that, for each $\theta \in (0,1]$,

$$s_k(\theta) = \Lambda(k\alpha_{l_k} QD_{k,l_k}Q^\top \lambda; x^\star) \tag{81}$$

where $[D_{k,l_k}]_{ii} = \beta_{k,l_k}(\rho_i, 0)$, l_k is the index of the interval in the definition of s_k to which θ belongs, $l_k = \min\{l = 1,..,k : \theta \leq \frac{l}{k}\}$, and ρ_i is, we recall, the i-th eigenvalue of $H^\star$.

Using the bounds from Lemma 2, it is easy to establish the by sandwiching argument that

$$\lim_{k\to\infty} k\alpha_{l_k}\beta_{k,l_k}(\rho_i, 0) = a\theta^{a\rho_i - 1}. \tag{82}$$

The limit in (80) now follows by the continuity of $\Lambda(\cdot; x^\star)$, which follows by convexity of $\Lambda(\cdot; x^\star)$, Lemma 3.

Using the fact that s_k can be uniformly bounded for all k and $\theta \in [0,1]$, we can exchange the order of the limit and the integral, to obtain:

$$\lim_{k\to+\infty}\int_0^1 s_k(\theta)d\theta = \int_0^1 \lim_{k\to+\infty} s_k(\theta)d\theta = \int_0^1 \Lambda(aQD(\theta)Q^\top\lambda; x^\star), \tag{83}$$

establishing the claim of the lemma. □

Appendix D.

Derivations with Remark 6. Consider function $\overline{\Psi}(\lambda) = \Psi^\star(\lambda)+r(\lambda)$ in Lemma 6. We derive here a lower bound on rate function $\overline{I}$ in Theorem 1 that does not explicitly depend on $H(x^\star)$. In view of the fact that $\overline{I}$ is the Fenchel-Legendre transform of $\overline{\Psi}$, a lower bound on $\overline{I}$ is readily obtained by deriving an upper bound on $\overline{\Psi}(\lambda)$. Note that $r(\lambda)$ does not explicitly depend on $H(x^\star)$, hence we only need to derive an upper bound on $\Psi^\star$. By Assumption 5, we have, for any $\theta \in [0,1]$, that $\Lambda(aQD(\theta)Q^\top\lambda; x^\star) \leq \frac{C_1\,a^2}{2}\lambda^\top(QDQ^\top)^2\lambda \leq \frac{C_1\,a^2}{2}\|\lambda\|^2\,\|D(\theta)\|^2$, where we recall that $\|\cdot\|$ denotes the 2-norm of its vector or matrix argument. Next, note that $\|D(\theta)\| \leq \theta^{a\,\mu-1}$, for all $\theta \in [0,1]$, because all eigenvalues ρ_i's of $H(x^\star)$ belong to the interval $[\mu, L]$. Therefore, we obtain:

$$\Psi^\star(\lambda) \leq \frac{C_1\,a^2}{2}\|\lambda\|^2\int_0^1 \theta^{2\,a\,\mu-2}d\theta = \frac{C_1\,a^2}{2(2\,a\mu-1)}\|\lambda\|^2.$$

□

The case of random initial iterate X_1. Recall that, by definition, $\Gamma_1(\lambda) = \mathbb{E}\left[e^{\lambda^\top(X_1-x^\star)}\right]$. When X_1 is random, Γ_1, as a function of λ, is therefore the log-moment generating function of $X_1 - x^\star$. Provided its domain is $\mathbb{R}^d$, all arguments in the proof of Theorem 1 remain the same. In particular, in eq. (54), the factor $e^{\|\lambda\|\|X_1-x^\star\|}$ would be replaced by a (finite-valued) function (of λ), and the subsequent results would be unaltered; a similar comment applies for the statement and the proof of Lemma 5. □

Appendix E.

Proof of Theorem 2. It is easy to show that for the assumed quadratic form, the iterates X_k have the following representation:

$$X_{k+1} = A_{k0}X_1 + \sum_{l=1}^{k}\alpha_l A_{k,l+1}Z_l, \tag{84}$$

where $A_{k,l} = \prod_{j=l}^{k}(I - \alpha_j H)$. By the assumption that the noise realizations at different times are independent and with a constant distribution, we obtain:

$$\Gamma_{k+1}(\lambda) = e^{\lambda^\top X_1} e^{\sum_{l=1}^{k} \Lambda(\alpha_l A_{k,l+1}\lambda)}. \tag{85}$$

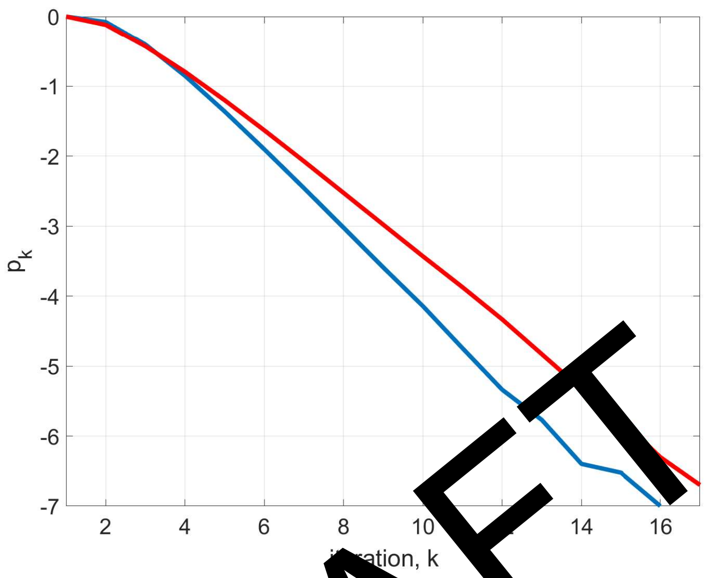
The proof now follows from Lemma 4 and the limit established in 56. □

Appendix F. Numerical results

We now illustrate the achieved results through a numerical simulation. We consider a strongly convex quadratic cost function $f : \mathbb{R}^d \to \mathbb{R}$, defined by $f(x) = \frac{1}{2}x^\top Ax + bx$, $d = 10$, where the symmetric $d \times d$ matrix A and the $d \times 1$ vector b are generated randomly. Specifically, we generate the entries of b mutually independently, according to the standard normal distribution. The matrix A is generated as follows. We let $A = Q\Lambda Q^\top$, where Q is the matrix whose columns are the orthonormal eigenvectors of matrix $(B + B^\top)/2$, and the entries of B are drawn mutually independently from the standard normal distribution; the matrix Λ is the diagonal matrix whose diagonal entries are drawn from the uniform distribution on the interval $[1, 2]$. Clearly, the optimal solution for the problem equals $x^\star = -A^{-1}b$.

We consider the gradient noise that is generated in an i.i.d. manner over iterations and over the gradient noise vector elements, independently from the solution iterate sequence. Two different noise distributions per gradient noise entry are considered, such that the per-entry noise variance is kept equal for the two distributions, equal to σ^2. In this way, we evaluate the effects of higher order moments on the performance of SGD. The first distribution is zero-mean Gaussian with variance σ^2. The second distribution is the zero-mean Laplacian with the same variance. We set $\sigma^2 = 0.04$.

We numerically estimate, via Monte Carlo simulations, the probability $P(\|X_k - x^\star\| > \delta)$ along iterations $k = 1, 2, ...$ We denote the corresponding numerical estimate by p_k. Two different values of δ are considered, $\delta = 0.3$, and $\delta = 0.03$. For each Monte Carlo run, X_1 is set to the zero vector. For the numerical example here, $\|x^\star\| = 2.342$, and hence $\delta = 0.3$ corresponds to the relative error level $\delta/\|x^\star\| \approx 0.13$, while $\delta = 0.03$ corresponds to $\delta/\|x^\star\| \approx 0.013$. Figure 1 plots p_k versus iteration counter k (in linear scale for the horizontal axis, and $\log_{10}$-scale for the vertical axis) for the Gaussian noise case (blue line) and the Laplacian noise case (red line). The top Figure is for $\delta = 0.3$, and the bottom Figure is for $\delta = 0.03$. We can see that, for a large value of δ, the two curves are very different: the Laplacian gradient noise case leads to a worse performance. This is because, for large δ, the argument λ of the LMGF Λ that corresponds to the minimizer in the rate function value $I^\star$ is large (see Theorem 4), and hence higher order polynomial coefficients ($\sim \lambda^3$ and higher) play a significant role. As the higher order moments of the Gaussian and Laplace distributions are very different (equal to zero for the Gaussian and strictly positive for the Laplacian), the result is the different large deviations performance (worse for the Laplacian case) as seen in Figure 1, top. On the other hand, for a small value of δ (bottom Figure), the argument λ of the LMGF that corresponds to the minimizer in the rate function expression $I^\star$ is small, and hence only the first two order polynomial coefficients of Λ play a significant role. As the two distributions here are both zero mean and have equal variance (hence having equal first and second order moments), the large deviation performance for the two noises matches, as seen in Figure 1, bottom. This behavior is in accordance with the theory derived.



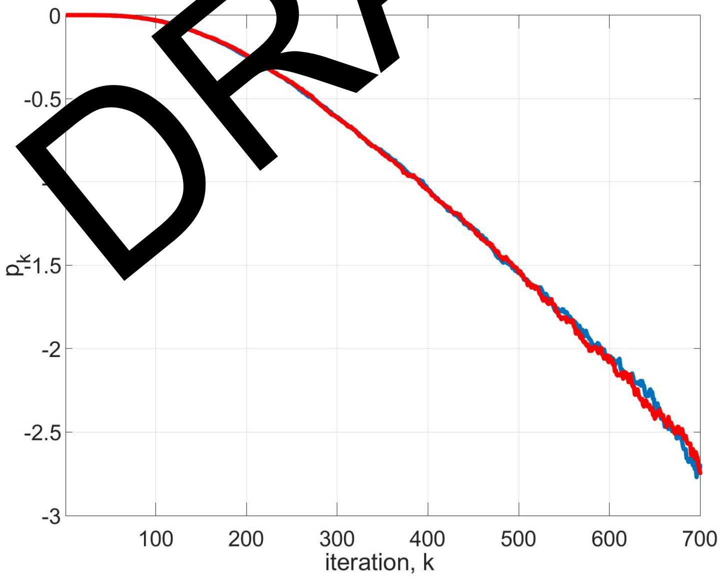


Figure 1: Monte Carlo estimate of $P\left(\|X_k - x^\star\| > \delta\right)$ along iterations $k = 1, 2, ...$ for SGD with Gaussian (blue line) and Laplacian (red line) gradient noise with equal per-entry variance $\sigma^2 = 0.04$. Top Figure: $\delta = 0.3$; Bottom Figure: $\delta = 0.03$.

G Inaccuracy Rates for Distributed Inference over Random Networks with Applications to Social Learning

The appended preprint follows.

Inaccuracy rates for distributed inference over random networks with applications to social learning

Dragana Bajović, *Member, IEEE*

Abstract

This paper studies probabilistic rates of convergence for consensus+innovations type of algorithms in random, generic networks. For each node, we find a lower and also a family of upper bounds on the large deviations rate function, thus enabling the computation of the exponential convergence rates for the events of interest on the iterates. Relevant applications include error exponents in distributed hypothesis testing, rates of convergence of beliefs in social learning, and inaccuracy rates in distributed estimation. The bounds on the rate function have a very particular form at each node: they are constructed as the convex envelope between the rate function of the hypothetical fusion center and the rate function corresponding to a certain topological mode of the node's presence. We further show tightness of the discovered bounds for several cases, such as pendant nodes and regular networks, thus establishing the first proof of the large deviations principle for consensus+innovations and social learning in random networks.

Index Terms

Large deviations, distributed inference, social learning, convex analysis, inaccuracy rates.

D. Bajović is with the Department of Power, Electronics and Communications Engineering, Faculty of Technical Sciences, University of Novi Sad. Email: dbajovic@uns.ac.rs.

Part of this work was done while the author was with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh 15213, PA, USA [1].

This work is partially supported by the European Union's Horizon 2020 Research and Innovation program under grant agreement No 957337. The paper reflects only the view of the author and the Commission is not responsible for any use that may be made of the information it contains.

I. Introduction

The theory of large deviations is the most prominent tool for studying *rare events* that occur with stochastic processes, offering a principled approach for estimating probabilities of such events. A typical setup concerns a sequence of probability measures induced by the studied process and parameterized by one of the process parameters (e.g., time, population size, learning rate etc.), with the goal of computing, or characterizing, the respective decay rate, for any given event (region) of interest. The practical value of such rates is in estimating the probability of a rare event of interest as an exponentially decaying function of the concerned process parameter, while neglecting the terms with slower than exponential dependence. The rates of rare events can additionally provide a ground for comparison of two statistical procedures, as originally proposed in the seminal work by Chernoff [2], and can therefore serve as a useful design criterion [3], [4], [5], [6]. This is of special interest in the cases when other performance metrics are intractable for optimization, such as probabilities of error with hypothesis testing.

In addition to the rate computation, large deviations analysis often reveals the most likely way through which the event of interest takes place, providing additional important insights that can guide system design. Most notable applications of large deviations theory are in statistics [7], communications and queuing theory [8], statistical mechanics [9], and information theory [10].

For example, in statistical estimation, the event of interest is the event that the estimator does not belong to a predefined close neighborhood of the parameter being estimated [11]. The decay rates of probabilities of such events are known in the estimation theory as *inaccuracy rates* and can, e.g., guide the decision on how many samples are needed for the estimator to reach the desired accuracy, with high probability [12]. To make the exposition concrete, let $X_t \in \mathbb{R}^d$, $t = 1, 2, ...$, be a sequence of estimators of a parameter $\theta \in \mathbb{R}^d$. Assuming that X_t converges to θ, an event of interest has the form $\{\|X_t - \theta\| \geq \epsilon\}$, where $\|\cdot\|$ denotes the l_2 norm (other vector norms can also be used). An equivalent way to represent this event is $\{X_t \in C_\epsilon\}$, where C_ϵ is the complement of the l_2 ball of diameter ϵ centered at θ, $C_\epsilon = B_\theta^c(\epsilon)$. Provided that X_t converges to θ, the probabilities of these events typically vanish exponentially fast with t. Large deviations analysis then aims at discovering the corresponding rate of decay, i.e., the inaccuracy rate $\mathbf{I}(C_\epsilon)$:

$$\mathbb{P}\left(X_t \in C_\epsilon\right) = e^{-t\mathbf{I}(C_\epsilon)+o(t)}, \tag{1}$$

where $o(t)$ denotes a function growing slower than linear with t. The inaccuracy rate $\mathbf{I}(C_\epsilon)$ has a very particular structure: it is given through the so called *rate function* $I : \mathbb{R}^d \mapsto \mathbb{R}$ by

$$\mathbf{I}(C_\epsilon) = \inf_{x \in C_\epsilon} I(x). \tag{2}$$

The rate function I is itself defined through the statistics of the inference sequence X_t. It should be noted that, in contrast with the set function $\mathbf{I}$, the rate function I does not depend on the inaccuracy region, i.e., when C_ϵ varies, only the domain of minimization on the right-hand side of (2) varies, while the rate function remains fixed. Also, this relation holds for an arbitrary set C_ϵ (e.g., not necessarily a ball complement). Hence, once the rate function is identified, the associated inaccuracy rate is readily computable through (2) for a new given region of interest, without the need to redo the large deviations analysis each time, i.e., for each new region. Large deviations rate for estimation were first studied by Bahadur in [12].

Another well-known application of large deviations analysis is hypothesis testing [2], where the sequence X_t is typically a decision statistics, e.g., obtained by summing up the log-likelihoods of the collected measurements up to the current time t, $X_t = 1/t\sum_{s=1}^{t}\log\frac{f_1(Y_s)}{f_0(Y_s)}$; f_0 and f_1 here are the marginal distributions of the measurements Y_s under the two hypotheses H_0 and H_1, respectively. If the acceptance threshold for H_1 at time t is γ_t, then rare events of interest are $\{X_t < \gamma_t\}$, when H_1 is true (i.e., when Y_s follow the distribution f_1) – resulting in missed detection, and $\{X_t \geq \gamma_t\}$, when H_0 is true (when Y_s follow the distribution f_0) – causing a false alarm. When C_ϵ in (1) is replaced by the preceding two events, the resulting large deviations rates $\mathbf{I}(C_\epsilon)$ are then the well-known *error exponents* that provide decay rates of the corresponding error probabilities.

In this paper we are concerned with large deviations rates of *distributed* statistical inference, where observations originate at different locations or different entities. Relevant works include algorithms such as consensus+innovations [13], [14], [15], [16], diffusion [17], [18], [19], and non-Bayesian or social learning [20], [21], [22], [23]. The common setup of the above works consists of networked nodes, each holding a local inference vector (parameter estimates, decision variables, beliefs) that is being updated over time. The updates are based on incorporating local, private signals that each agent observes over time, and then exchanging with immediate neighbors and averaging the received information through the well-known DeGroot averaging [24] (also known as consensus).

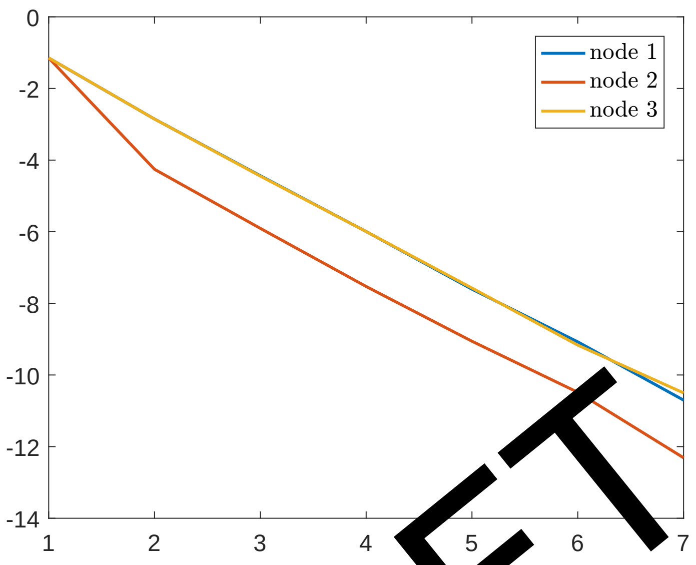
Asymptotic performance of distributed detection was studied in [13], for Gaussian observations, [14], for generic observations, and in [15], for networks with noisy communication links. In each of the named works, a randomly switching network topology is assumed and conditions for asymptotic equivalence of an arbitrary network node and a fusion center (with access to all observations) are studied. Reference [16] considers directed networks, both static and randomly varying, and studies the rate function for the vector of states, deriving the exact rate function for the case of static networks, and providing bounds on the exponential rates for randomly switching networks. The rate function for static networks is given as the weighted combination of the local rate functions, with weights being equal to the eigenvector centralities (i.e., the left Perron vector of the consensus matrix). Reference [17] studies distributed detection for static and symmetric networks and constant step size. For the limiting distribution of the local states, it proves the large deviations principle when the step size parameter decreases and shows that the rate function is equivalent to the centralized detector. These results are refined and extended in [18] by studying non-exponential terms and directed (static) networks. Reference [19] further considers distributed detection with 1-bit messages, while recent reference [6] addresses optimal aggregation strategies for social learning.

References [20], [21], [22], [23] study distributed M-ary hypothesis testing, where local updates are formed by applying Bayesian update on the vector of prior beliefs, based on the newly acquired local measurements. Assuming static, directed network, in [20] and [21], beliefs across immediate neighborhoods are merged through arithmetic average [20], while [22] adopts geometric average (or, equivalently, arithmetic average on the log-beliefs). A different merging rule is proposed and analyzed in [23], where instead of averaging, beliefs are updated by computing the minimum across the neighbors beliefs and the nodes' locally generated beliefs, showing improvement in the learning rate. Large deviations of the beliefs are addressed in [22], where it was proven that the log-ratios of beliefs with respect to the belief in the true distribution, satisfy the large deviations principle, with the rate function being equal to the eigenvector-centralities convex combination of the nodes' local rate functions, similarly as in [16] and [18]. Through the contraction principle, [22] also shows that the (log)-beliefs themselves satisfy the large deviations principle.

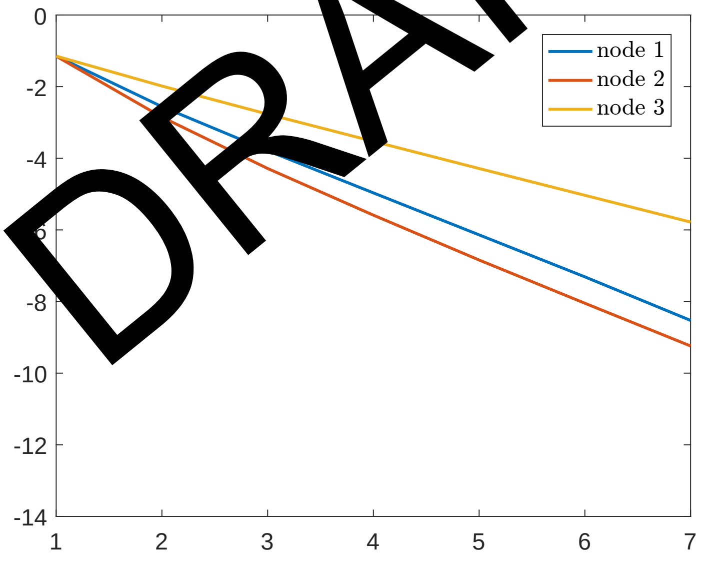
Contributions. In contrast with the works in [17]–[23], in this paper, we address computation of the rate function for distributed inference over *random networks*. This model shift from static to random networks has fundamental implications on the large deviations performance. To explain this at an intuitive level: when the underlying network is random, consensus mixing of local inference vectors might be disabled for an arbitrary long period of time due to the lack of communications. In general, the topology can then break down into several connected components of the original network[1]. When in this regime, neither of the nodes can "see" the observations beyond the connected component they belong to, and hence the resulting rate function will be strictly lower than that of the full network[2]. Figure 1 illustrates this effect with a toy example of a 3-node chain where each node produces scalar observations of standard Gaussian distribution.

[1]Note that this is very different from time-varying networks that are typically modelled by the assumption of the so called bounded intercommunication interval, which guarantees that the union graph formed of all communication links occurring in this interval is connected, after a strictly finite time, e.g., [23], [25]

[2]This is a consequence of the non-negativity of the rate function and the fact that it (roughly) scales linearly with the number of observation sources, as detailed in the paper.



(a) Static topology



(b) Randomly varying topology

Fig. 1: Decay of the log-probabilities in (1) for a fixed set C for static (top) and random (bottom) 3 node chain network.

In the top figure, we plot the logarithm of the probability in (1) for a ball complement inaccuracy set C, when the chain topology is fixed (static). In the bottom figure we plot the same probability, but when the two links of the chain graph alternate at random over time. We label the middle node as node 2, and we let the communication frequency between nodes 1 and 2 be higher (equal to 0.8) than the one between higher than the one between nodes 2 and 3 (equal to 0.2). It is clear from the figure that the static topology achieves much steeper decay, and, moreover, this decay is equal at each of the three nodes (and also equal to the decay of the hypothetical fusion center, cf. Section IV, as predicted by the theory). In contrast, in the random case, the difference between the nodes' decays is evident: node 2 achieves the steepest decay, followed by node 1, while node 3 has the worst performance.

In this work, we are interested in understanding the rate function of each node in the network and analytically expressing its dependence on the system parameters. For each node, we find a lower and a family of upper bounds on the rate function. This is achieved by carrying out node-specific large deviations analyses. We show that the two bounds match in several cases, such as for pendant nodes and also for nodes in a regular network. The family of upper bounds is indexed by different induced components of the given node, and each function in this family has the form of the convex envelope between the rate function of the full network and the rate function of the respective component, lifted up by the probability of the event that induces the component. The lower bound is given as the convex envelope between the rate function of the full network and the node's local rate function lifted up by the large deviations rate of consensus, whose existence was shown in [26]. With respect to references [13], [14], [15], there are several important novelties. First, we extend the results of [14] to the case of vector observations and vector inference states. Second, while [13]-[15] only provide a lower bound on the rate function, this work, as described in the above, finds also a family of upper bounds. This is achieved by carefully devising events that impact the rate function, and for which we develop novel large deviations techniques. The discovered upper bounds enable to establish, to the best of our knowledge, the first proof of the large deviations principle for nodes performing DeGroot-based distributed inference in randomly varying networks.

As an application of particular interest to this study, we consider social learning, specifically the form with the geometric average update [22]. We show that, with appropriate transformation of the belief iterates – namely, considering their log-ratios with respect to the belief in the true distribution, the algorithm studied in [22] exhibits full equivalence to the consensus+innovations algorithm that we analyze here. Building on this equivalence, we characterize the rate function of the beliefs in social learning and provide the first proof of the large deviations principle for social learning run over random networks.

A closely related work to ours is [27] that studies convergence properties of social learning over random networks. This reference shows that, almost surely, each node is able to correctly identify the true hypothesis. We similarly focus on the case of random networks, but we are additionally concerned with characterizing the *rates* of probabilistic convergence of the iterates in the sense of large deviations. Finally, we show that almost sure convergence of the beliefs follows from the obtained large deviations rates.

From the technical perspective, this paper contributes with a novel set of techniques and approaches that could be of interest for further studies of social learning, and more generally, distributed inference in random networks.

Notation. For arbitrary $d \in \mathbb{N}$ we denote by 0_d the d-dimensional vector of all zeros; by 1_d the d-dimensional vector of all ones; by e_i the i-th canonical vector of $\mathbb{R}^d$ (that has value one on the i-th entry and the remaining entries are zero); by I_d the d-dimensional identity matrix; by J_d the $d \times d$ matrix whose all entries equal to $1/d$. For a matrix A, we let $[A]_{ij}$ and A_{ij} denote its i,j entry and for a vector $a \in \mathbb{R}^d$, we denote its i-th entry by a_i, $i,j = 1,...,d$. For the set of indices $C \subseteq \{1,2,...,N\}$, we let $[A]_C$ (or A_C) denote the submatrix of A that corresponds to indices in C. For a function $f : \mathbb{R}^d \mapsto \mathbb{R}$, we denote its domain by $\mathcal{D}_f = \left\{x \in \mathbb{R}^d : -\infty < f(x) < +\infty\right\}$; for a set $D \subseteq \mathbb{R}$, $f^{-1}(D)$ is defined as $f^{-1}(D) = \{x \in \mathbb{R}^d : f(x) \in D\}$. log denotes the natural logarithm. For $N \in \mathbb{N}$, we denote by Δ_{N-1} the probability simplex in $\mathbb{R}^N$ and by α the generic element of this set, $\Delta_{N-1} = \left\{\alpha \in \mathbb{R}^N : \alpha_i \geq 0, \sum_{i=1}^N \alpha_i = 1\right\}$. We let $\lambda_{\max}$ and λ_2, respectively, denote the maximal and the second largest (in modulus) eigenvalue of a square matrix; $\|\cdot\|$ denotes the spectral norm. For a matrix $S \in \mathbb{R}^{N\times N}$, we let $\mathcal{R}(S)$ denote the range of S, $\mathcal{R}(S) = \left\{Sx : x \in \mathbb{R}^N\right\}$. An open Euclidean ball in $\mathbb{R}^d$ of radius ρ and centered at x is denoted by $B_x(\rho)$; the closure, the interior, and the complement of an arbitrary set $D \subseteq \mathbb{R}^d$ are respectively denoted by $\overline{D}$, D^{o}, and D^{c}; $\mathcal{B}(\mathbb{R}^d)$ denotes the Borel sigma algebra on $\mathbb{R}^d$; $\mathbb{P}$ and $\mathbb{E}$ denote the probability and the expectation operator; $\mathcal{N}(m, S)$ denotes Gaussian distribution with mean vector m and covariance matrix S. For a given graph H, $E(H)$ denotes the set of edges of H.

Paper organization. Section II describes the system model and the algorithm and Section III introduces the large deviations metric and defines the relevant large deviations quantities. Section IV states the main result of the paper, important corollaries and provides illustration examples. Section V provides applications of the results to social learning. Proofs of the main result are given in Section VI. Section VII concludes the paper.

II. System model

This section explains the system model and the consensus+innovations distributed inference algorithm accompanied by different application examples. Section II-A details the connection to social learning, while Section II-B provides certain preliminaries.

Communication model. We consider a network of N identical agents connected by an arbitrary communication topology. The topology is represented by an undirected graph $\overline{G} = (V, \overline{E})$, where $V = \{1, 2, ..., N\}$ is the set of agents, and $\overline{E} \subseteq \binom{V}{2}$ is the set of possible communication links between agents. We assume that during operation of the network each link $\{i, j\} \in \overline{E}$ may fail, and that correlations between failures of different links are possible. Realization (i.e., a snapshot) of the communication topology at time slot t is denoted by $G_t = (V, E_t)$, for $t = 1, 2, \dots$, where E_t is the set of links that are online at time t; note that $E_t \subseteq \overline{E}$. For an agent i, we let $O_{i,t}$ denote the set of neighbors of i at time t, $O_{i,t} = \{j \in V : \{i, j\} \in E_t\}$.

Consensus based distributed estimation. At each time t, each sensor i acquires a d-dimensional vector of measurements $Z_{i,t} \in \mathbb{R}^d$. We assume that the measurements $Z_{i,t}$ are independent and identically distributed across sensors and over time. The goal of each sensor is to estimate the state of nature θ, which is the expected value of sensor observations $Z_{i,t}$, $\theta = \mathbb{E}\left[Z_{i,t}\right]$. To achieve this, an agent i holds a local estimate, called also the state, $X_{i,t}$ and iteratively updates it over time slots t. At each slot t, agent i performs two steps: 1) the innovation step; and 2) the consensus step. In the innovation step, i acquires $Z_{i,t}$ and incorporates it into the current state $X_{i,t-1}$, by computing the following convex combination, forming an intermediate state:

$$\widehat{X}_{i,t} = \frac{t-1}{t} X_{i,t-1} + \frac{1}{t} Z_{i,t}. \tag{3}$$

It then subsequently transmits $\widehat{X}_{i,t}$ to (possibly a subset of) its neighbors in $\overline{G}$, and, at the same time, receives the intermediate states $\widehat{X}_{j,t}$, $j \in O_{i,t}$, from its current neighbors. In the second, consensus, step, agent i computes the convex combination (DeGroot averaging) between its own and the neighbors estimates:

$$X_{i,t} = \sum_{j \in O_{i,t} \cup \{i\}} W_{ij,t} \widehat{X}_{j,t}, \tag{4}$$

where $W_{ij,t}$ is the weight that agent i at time t assigns to the estimate of agent j. For neat exposition, the weights of all nodes are collected in an N by N matrix W_t, such that the i, j entry of W_t equals $W_{ij,t}$, when $j \in O_{i,t} \cup \{i\}$, and equals zero otherwise. Thus, W_t respects the sparsity pattern of G_t: if $\{i, j\} \notin E_t$, then $[W_t]_{ij} = [W_t]_{ji} = 0$. Also, since the weights at each node form a convex combination, matrix W_t is stochastic. In addition, we assume that, at any time t, for any i, j, the weights are symmetric at each link, i.e., $W_{ij,t} = W_{ji,t}$, implying that W_t is symmetric.

Denoting by $\Phi(t, s) = W_t \cdots W_s$ for $1 \leq s \leq t$, algorithm (3)-(4) can be written as:

$$X_{i,t} = \frac{1}{t} \sum_{s=1}^{t} \sum_{j=1}^{N} [\Phi(t, s)]_{i,j} \, Z_{j,s}. \tag{5}$$

We analyse algorithm (3)-(4) under the following assumptions on the matrices W_t and observations $Z_{i,t}$.

Assumption 1 (Network and observations random model)**.**

1) *Observations $Z_{i,t}$, $i = 1, \ldots, N$, $t = 1, 2, \ldots$ are independent, identically distributed (i.i.d.) across nodes and over time;*
2) *The sequence of matrices W_t, $t = 1, 2, \ldots$ is i.i.d. and for each t, every realization of W_t is stochastic, symmetric and has positive diagonals;*
3) $\lambda_2 (\mathbb{E}[W_t]) < 1$*, or, equivalently, the induced graph $\overline{G}$ of $\mathbb{E}[W_t]$ is connected.*
4) *Weight matrices W_t are independent from the nodes' observations $Z_{i,s}$ for all i, s, t.*

We now present different application examples of algorithm (3)-(4).

Example 2 (Estimating the distribution of opinions by social sampling)**.** *Consider the scenario where a group of N agents wishes to discover the distribution of opinions (e.g., about an event or phenomenon) across a certain, large population. To achieve this, agents continuously poll the population and register responses of individuals. We assume that the respondents' opinions are quantized to d preset opinion summaries: $\{r_1, ..., r_d\}$. We let $\mathcal{R}_{i,t}$ denote the opinion (summary) of the person that agent i interviewed at time t. Also, let p_l be the probability that the response of a person chosen uniformly at random is r_l. Consider now algorithm* (3)-(4) *and define the innovation vector $Z_{i,t}$ to be the vector of opinion indicators, $Z_{i,t} = \left(1_{\{\mathcal{R}_{i,t}=r_1\}}, ..., 1_{\{\mathcal{R}_{i,t}=r_d\}}\right)^\top$; again, let the W_t's be arbitrary stochastic matrices. Then, the states of all agents converge to the true opinion distribution, $(p_1, \ldots, p_d)$, as we show in Section IV, i.e., algorithm* (3)-(4) *is able to correctly identify the distribution of opinions across a given population, while the rates of this convergence will prove to be highly dependent on the frequency of agents' interactions and interaction patterns.*

Example 3 (Distributed event detection)**.** *Suppose that a wireless sensor network is deployed in a certain area to detect in which of the two possible states the environment is. This problem can be modeled as a binary hypothesis testing problem, where under the state of nature (hypothesis) $\mathbf{H}_1$, the sensors measurements follow the distribution f_1, and similarly for f_0, where f_1 and f_0 are assumed known. We let $Y_{i,t}$ denote the measurement of sensor i at time t. We assume that $Y_{i,t}$'s are independent both over time and across different sensors. This hypothesis testing problem can be solved by algorithm* (3)-(4) *as follows. For each i and t, define the innovation $Z_{i,t}$ as the log-likelihood ratio of the node i's measurement at time t: $Z_{i,t} = \log \frac{f_1(Y_{i,t})}{f_0(Y_{i,t})}$. Then, any sensor in the system can, at any given time, make a decision simply by comparing its state $X_{i,t}$ against a prescribed threshold γ:*

$$X_{i,t} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \gamma. \tag{6}$$

For further details on distributed detection application, see also [14].

A generalization of the preceding example to M-ary hypothesis testing and an application to social learning is given in the next subsection.

A. Social learning

The idea of social learning is for a group of people to distinguish between M different hypotheses, potentially indistinguishable by any given individual, through local Bayesian updates and collaborative information exchange. Each node i over time draws observations $Y_{i,t}$ from (the true) distribution $f_{i,M}$ (hypothesis $\mathbf{H}_M$); the remaining $M-1$ candidate distributions that compete at node i in hypothesis testing are $f_{i,m}$ (hypothesis $\mathbf{H}_m$), $m = 1, ..., M-1$. It is assumed that, conditioned on the true hypothesis $\mathbf{H}_M$, observations at each node are independent over time, and they are also independent from the observations that are generated at any different node.

We consider here the algorithm for social learning proposed in [22]. Each node i maintains over time two sets of values (vectors), $q_{i,t} \in \mathbb{R}^M$ and $b_{i,t} \in \mathbb{R}^M$, called, respectively, *private* and *public belief* vectors, quantifying node i's beliefs in each of the M hypotheses. The m-th entry of $q_{i,t}$, denoted by $q_{i,t}^m \in \mathbb{R}$, corresponds to the private belief of node i in the m-th hypothesis; similarly, the m-th entry of $b_{i,t}$, denoted by $b_{i,t}^m \in \mathbb{R}$, corresponds to the public belief of node i in the m-th hypothesis. The values of both public and private belief vectors are between 0 and 1: the closer an entry of a belief vector is to 1 (0), the stronger (weaker) is the confidence of the respective node that the corresponding hypothesis is true; e.g., if for some m, $b_{i,t}^m$ equals 1, this means that node i is fully confident that hypothesis $\mathbf{H}_m$ is true.

The algorithm starts at each node with initial private beliefs $q_{i,t}^m > 0$, $m = 1, ..., M-1$. Upon receiving new local observation $Y_{i,t}$, each node i updates its m-th public belief as follows:

$$b_{i,t}^m = \frac{f_{i,m}(Y_{i,t}) q_{i,t-1}^m}{\sum_{l=1}^M f_{i,l}(Y_{i,t}) q_{i,t-1}^l}, \tag{7}$$

for each $m = 1, ..., M$. The node then sends its updated public belief vector $b_{i,t} = (b_{i,t}^1, ..., b_{i,t}^M)^\top$ to all of its neighbors $O_{i,t}$. Upon receiving the neighbors' (public) beliefs, the node updates its private beliefs as follows:

$$q_{i,t}^m = \frac{e^{\sum_{j \in O_{i,t}} W_{ij,t} \log b_{j,t}^m}}{\sum_{l=1}^M e^{\sum_{j \in O_{i,t}} W_{ij,t} b_{j,t}^l}}, \tag{8}$$

for each $m = 1, ..., M$.

It is easy to verify that both $q_{i,t}$ and $b_{i,t}$ represent valid probability vectors, i.e., $q_{i,t}$, $b_{i,t} \in \Delta_{M-1}$.

Connection with algorithm (3)-(4)**.** Consider the update for the private belief $q^m_{i,t}$ in (8). Computing the log-ratios of $q^m_{i,t}$ with $q^M_{i,t}$ (belief in the true hypothesis $\mathbf{H}_M$), the recursion in (8) transforms into:

$$\log \frac{q^m_{i,t}}{q^M_{i,t}} = \sum_{j \in O_{i,t}} W_{ij,t} \log \frac{b^m_{j,t}}{b^M_{j,t}}. \tag{9}$$

Similarly, it is easy to see that the log-ratios of the public beliefs $b^m_{j,t}$ with $b^M_{j,t}$ can be expressed as:

$$\log \frac{b^m_{i,t}}{b^M_{i,t}} = \log \frac{q^m_{i,t-1}}{q^M_{i,t-1}} + \log \frac{f_{i,m}(Y_{i,t})}{f_{i,M}(Y_{i,t})}. \tag{10}$$

Dividing both sides in (9) and (10) by t, we recognize the form in (3)-(4). Further, denoting, for each $m = 1, ..., M-1$,

$$L^m_{i,t} = \log \frac{f_{i,m}(Y_{i,t})}{f_{i,M}(Y_{i,t}} \tag{11}$$

$$\widehat{X}^m_{i,t} = \frac{1}{t} \log \frac{q^m_{i,t}}{q^1_{i,t}} \tag{12}$$

$$X^m_{i,t} = \frac{1}{t} \log \frac{b^m_{i,t}}{b^1_{i,t}} \tag{13}$$

and stacking the per-hypothesis quantities in vector form: $L_{i,t} = \left(L^1_{i,t}, ..., L^{M-1}_{i,t}\right) \in \mathbb{R}^{M-1}$, and $\widehat{X}_{i,t} = \left(\widehat{X}^1_{i,t}, ..., \widehat{X}^{M-1}_{i,t}\right) \in \mathbb{R}^{M-1}$, and $X_{i,t} = \left(X^1_{i,t}, ..., X^{M-1}_{i,t}\right) \in \mathbb{R}^{M-1}$, the exact form in (3)-(4) is obtained, where the innovation vectors $Z_{i,t}$ that algorithm (3)-(4) is fed with are the log-likelihood ratio vectors $L_{i,t}$; note also that, in this application instance, $d = M-1$. Thus, the generic algorithmic form (3)-(4) subsumes also the social learning algorithm from (7)-(8) through the described variable transformation. Section V shows how results of this paper can be used to characterize convergence of beliefs and large deviations rates of social learning, specifically for the case when the weights $W_{ij,t}$ (neighborhoods $O_{i,t}$) in (8) are random.

B. Probabilistic rate of consensus $\mathcal{J}$

We next define certain concepts and quantities pertinent to the underlying graph process that are needed for later analyses.

Components in union graphs. Since the sequence of matrices W_t is i.i.d., the sequence G_t of their underlying topologies is i.i.d. as well. We let $\mathcal{G}$ denote the set of all topologies on V that have non-zero probability of occurrence at a given time t, i.e., $\mathcal{G} = \{(V, E) : \mathbb{P}(G_t = (V, E)) > 0\}$. For convenience, for any undirected, simple graph H on the set of vertices V we denote $p_H = \mathbb{P}(G_t = H)$. Thus, for any $H \in \mathcal{G}$, $p_H > 0$. It will also be of interest to consider different subsets of the set of feasible graphs $\mathcal{G}$. For a collection of undirected simple graphs $\mathcal{H}$ on V we let $\Gamma_{\mathcal{H}} = (V, E_{\mathcal{H}})$ denote the corresponding union graph, that is, $\Gamma_{\mathcal{H}}$ is the graph with the set of vertices V and whose edge set $E_{\mathcal{H}}$ is the union of edge sets of all the graphs in $\mathcal{H}$, $E_{\mathcal{H}} = \cup_{H \in \mathcal{H}} E(H)$. We let $p_{\mathcal{H}}$ denote the probability that G_t belongs to $\mathcal{H}$,

$$p_{\mathcal{H}} = \sum_{H \in \mathcal{H}} p_H.$$

We also introduce – what we refer to as – the component of a node in $\mathcal{H}$.

Definition 4 (Node component in union graph)**.** *Let $\mathcal{H}$ be a given collection of undirected simple graphs on V and let $C_1, ..., C_L$ be the components of the union graph $\Gamma(\mathcal{H})$. Then, the component of node i in $\mathcal{H}$, denoted by $C_{i,\mathcal{H}}$, is the component of $\Gamma(\mathcal{H})$ that contains i: i.e., if $i \in C_l$, then $C_{i,\mathcal{H}} = C_l$.*

Probabilistic rate of consensus $\mathcal{J}$. We recall here the rate of consensus, associated with a sequence of random stochastic symmetric matrices, introduced in [13] and subsequently analyzed in [26]. In [13] and [14] we showed that the quantity $\mathcal{J}$ below, termed the rate of consensus[3], captures well how the weight matrices W_t affect performance of the estimates $X_{i,t}$ when one is concerned with large deviations metrics:

$$\mathcal{J} := -\limsup_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(\|W_t \cdots W_1 - J\| > \frac{1}{t} \right). \tag{14}$$

Rate of consensus $\mathcal{J}$ is computed exactly in [26]:

Theorem 5 ([26])**.** *Let Assumption 1 part 3 hold. Then the* $\limsup$ *in* (14) *is in fact a limit and the rate of consensus $\mathcal{J}$ is found by*

$$\mathcal{J} = |\log p_{\mathcal{H}^\star}|,$$

where $p_{\max}$ is the probability of the most likely collection of feasible graphs whose union graph is disconnected,

$$\mathcal{H}^\star = \arg \max_{\mathcal{H} \subseteq \mathcal{G}\colon \Gamma_{\mathcal{H}} \text{ disc.}} p_{\mathcal{H}}. \tag{15}$$

In the next example we consider an important special case when links in $\overline{G}$ fail independently at random.

[3]The rate of consensus $\mathcal{J}$ (in (14)) is defined slightly differently than the corresponding quantity from [13] and [14]. In [13] and [14], in the event $\|W_t \cdots W_1 - J_N\| > 1/t$, the probability of which we wish to compute, there is a constant $\varepsilon \in (0, 1]$ in the place of $1/t$. However, as we show in [26], the two rate quantities coincide when the weight matrices are i.i.d., which is the case that we consider here.

Example 6 (Random topologies with i.i.d. link failures)**.** *Consider the random model for W_t defined by Assumption 1.2 where each link in $\overline{G}$ fails independently from other links with probability $1-p$. Applying Theorem 5, it can be shown that*

$$\mathcal{J} = \min\operatorname{cut}(\overline{G})|\log(1-p)|, \tag{16}$$

where $\min\operatorname{cut}(\overline{G})$ *is the minimum edge cut of the graph* $\overline{G}$*; for example, if* $\overline{G}$ *is a chain, then* $\min\operatorname{cut}(\overline{G}) = 1$*. The details of this derivation can be found in [26].*

For finite time analyses, of relevance is the following variant of (14): for any $\epsilon > 0$, there exists a positive constant K_ϵ such that for all t,

$$\mathbb{P}\left(\|W_t \cdots W_s - J_N\| > \frac{1}{t}\right) \leq K_\epsilon e^{-(t-s)\,(\mathcal{J}-\epsilon)}. \tag{17}$$

III. Problem formulation: The metric of large deviations

Section II illustrate uses of algorithm (3)-(4) for several applications: multi-agent polling with cooperation, in Example 2, fully distributed hypothesis testing, in Example 3, and social learning, in Section II-A. We now introduce the rates of large deviations that we adopt as performance metric for applications of algorithm (3)-(4).

Rate function I and the large deviations principle.

Definition 7 (Rate function I [28])**.** *Function $I : \mathbb{R}^d \mapsto [0,+\infty]$ is called a* rate function *if it is lower semicontinuous, or, equivalently, if its level sets are closed. If, in addition, the level sets of I are compact (i.e., closed and bounded), then I is called a good rate function.*

Definition 8 (The large deviations principle [28])**.** *Suppose that $I : \mathbb{R}^d \mapsto [0,+\infty]$ is lower semicontinuous. A sequence of measures μ_t on $\left(\mathbb{R}^d, \mathcal{B}\left(\mathbb{R}^d\right)\right)$, $t \geq 1$, is said to satisfy the large deviations principle (LDP) with rate function I if, for any measurable set $D \subseteq \mathbb{R}^d$, the following two conditions hold:*

1) $\limsup_{t\to+\infty} \dfrac{1}{t}\log\mu_t(D) \leq -\inf_{x\in\overline{D}} I(x)$;
2) $\liminf_{t\to+\infty} \dfrac{1}{t}\log\mu_t(D) \geq -\inf_{x\in D^o} I(x)$.

Differently than with the case of static topologies, when topologies and/or weight matrices W_t are random, finding the rate function of an arbitrary node performing distributed inference is a very difficult problem [14], [29]. (In fact, even the existence of the LDP is not known a priori.) Our approach is to find functions $\overline{I}_i$ and $\underline{I}_i : \mathbb{R}^d \mapsto \mathbb{R}$, such that, for any measurable set D:

$$\limsup_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in D\right) \le - \inf_{x\in \overline{D}} \underline{I}_i(x), \tag{18}$$

$$\liminf_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in D\right) \ge - \inf_{x\in D^{\circ}} \overline{I}_i(x). \tag{19}$$

At a high level, this is analytically achieved by carefully constructing events the probabilities of which upper and lower bound the probability of the event of interest in (18) and (19). We remark that functions $\underline{I}_i$ and $\overline{I}_i$ that we seek should satisfy (18) and (19) for any given set D, i.e., similarly as with the rate function I_i, to find bounds on the exponential rates for a given rare event $\{X_{i,t} \in D\}$, it suffices to perform minimizations of $\underline{I}_i$ and $\overline{I}_i$ over D. This property is very important, as once $\underline{I}_i$ and $\overline{I}_i$ are discovered, any inaccuracy rate can be easily estimated without the need to do any (further) large deviations analyses.

As we show in Appendix A, if for some node i the LDP holds, and (18) and (19) are satisfied for any D, then

$$\underline{I}_i(x) \le I_i(x) \le \overline{I}_i(x),\ x \in \mathbb{R}^d, \tag{20}$$

i.e., the graph of the LDP rate function I_i lies between the graphs of $\overline{I}_i$ and $\underline{I}_i$.

Log-moment generating function of observations $Z_{i,t}$ and its conjugate. We proceed standardly by introducing the log-moment generating function of the observation vectors $Z_{i,t}$, which we denote by Λ. The log-moment generating function $\Lambda : \mathbb{R}^d \mapsto \mathbb{R} \cup \{+\infty\}$ corresponding to $Z_{i,t}$ is defined by:

$$\Lambda(\lambda) = \log \mathbb{E}\left[e^{\lambda^\top Z_{i,t}}\right],\ \text{for}\ \lambda \in \mathbb{R}^d. \tag{21}$$

We make the assumption that Λ is finite at all points.

Assumption 9. *$\mathcal{D}_\Lambda = \mathbb{R}^d$, i.e., $\Lambda(\lambda) < +\infty$ for all $\lambda \in \mathbb{R}^d$.*

Besides the log-moment generating function Λ, the second key object in large deviations analysis is the Fenchel-Legendre transform, or the conjugate, of Λ, defined by

$$I(x) = \sup_{\lambda\in\mathbb{R}^d} x^\top \lambda - \Lambda(\lambda),\ \text{for}\ x \in \mathbb{R}^d. \tag{22}$$

Log-moment generating function and its conjugate enjoy many nice properties, such as convexity and differentiability in the interior of the function's domain [28], [30]. We list the properties that are relevant for the current analysis in the next lemma. Recall that $\theta = \mathbb{E}[Z_{i,t}]$.

Lemma 10 (Properties of Λ and I)**.**

1) *Λ is convex and differentiable on $\mathbb{R}^d$;*

2) $\Lambda(0) = 0$ *and* $\nabla\Lambda(0) = \theta$*;*
3) I *is strictly convex;*
4) *if* $x = \nabla\Lambda(\lambda)$ *for some* $\lambda \in \mathbb{R}^d$*, then* $I(x) = \lambda^\top x - \Lambda(\lambda)$*;*
5) $I(x) \geq 0$ *with equality if and only if* $x = \theta$*.*

Proofs of 1-5 (with a weaker form of the claim in part 3 – with strict convexity replaced by convexity, and with non-negativity only in part 5) can be found in [28]. The proof of strict convexity of I under Assumption 9 can be found in [31]. We briefly comment on properties 2 and 5, to give some (mathematical) intuition as to why these properties hold, where we note that of particular, practical relevance is 5. Plugging in $\lambda = 0$ in the defining equation of Λ, (21), it is easy to see that $\Lambda(0) = 0$. Similarly, it can be shown that, for any λ, $\nabla\Lambda(\lambda) = \mathbb{E}[Z_{i,t}e^{\lambda^\top Z_{i,t}}]/\mathbb{E}[e^{\lambda^\top Z_{i,t}}]$. Evaluating at $\lambda = 0$, the property $\nabla\Lambda(0) = \theta$ follows. Property 5 has a very intuitive meaning: the rate function is non-negative and also equals zero at the mean value. To see why the latter holds, it suffices to invoke properties from part 2 in 4; note also that, since I is non-negative, θ is a minimizer of I. The if and only if part then follows from strict convexity of I, which implies uniqueness of its minimizer θ. We will show practical implications of this property when considering large deviations rate of the sequence $X_{i,t}$.

The following result, proven in [16], gives fundamental large deviations upper and lower bound for the inference sequence $X_{i,t}$. The result holds for arbitrary stochastic weight matrices W_t and, in particular, for directed topologies as well. This result will be invoked when proving tightness and optimality of our rate function bounds for certain classes of networks, in Section IV-B.

Lemma 11 (Fundamental distributed inference bounds)**.** *Consider algorithm* (3)-(4) *under Assumptions 1 and 9. Then* (18) *and* (19) *hold with* $\overline{I}_i = NI$ *and* $\underline{I}_i = I$*, for all* i*.*

Closed convex hull of a function. We recall the definitions of the epigraph and closed convex hull of a function.

Definition 12 (Epigraph and closed convex hull of a function, [32])**.** *Let* $f : \mathbb{R}^d \mapsto \mathbb{R} \cup \{+\infty\}$ *be a given function.*

1) *The epigraph of* f*, denoted by* epi$f$*, is defined by*

$$\mathrm{epi} f = \left\{(x,r) : r \geq f(x), x \in \mathbb{R}^d\right\}. \tag{23}$$

2) *Consider the closed convex hull* $\overline{\mathrm{co}}\,\mathrm{epi}\, f$[4] *of the epigraph of* f*. The closed convex hull of* f*, denoted*

[4]The convex hull of a set A, where A is a subset of some Euclidean space, is defined as the set of all convex combinations of points in A [32].

by $\overline{\text{co}} f$, is defined by:

$$\overline{\text{co}} f(x) := \inf\{r : (x,r) \in \overline{\text{co}}\, \text{epi}\, f\}. \tag{24}$$

Hence, for a given function f, epigraph of f is the area above the graph of f. Closed convex hull of f is then constructed from $\text{epi}f$ by first finding the closed convex hull of the epigraph, $\overline{\text{co}}\, \text{epi}\, f$. Then, $\overline{\text{co}} f$ is defined as the function the epigraph of which matches $\overline{\text{co}}\, \text{epi}\, f$. Intuitively, $\overline{\text{co}} f$ is the best convex and lower semi-continuous (closed) approximation of f, as its epigraph contains (besides $\text{epi}\, f$) only those points that are needed for "convexification" and closure. Figure 2 further ahead gives an illustration of $\overline{\text{co}} f$, while construction of $\overline{\text{co}} f$ is explained in Section IV-A.

IV. Main result

The main result of this section, Theorem 13, finds functions $\underline{I}_i$ and $\overline{I}_i$ from (18) and (19). These functions enable computation of bounds on the exponential decay rate of an arbitrary rare event and, in the case of the existence of the LDP, by (20), provide approximations to the rate function I_i. A number of important corollaries of Theorem 13 is then presented in subsection IV-B, including the large deviations principle for regular networks and for pendant nodes. Section V then studies application of the derived results to distributed hypothesis testing and social learning.

Theorem 13. *Consider distributed inference algorithm* (3)-(4) *under Assumptions 1 and 9. Then, for each node i, for any measurable set D:*

1) $$\limsup_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in D\right) \leq - \inf_{x \in \overline{D}} I^\star(x), \tag{25}$$

   where $I^\star(x) = \overline{\text{co}} \inf\left\{I(x) + \mathcal{J}, NI(x)\right\}$;

2) *for any collection $\mathcal{H}$ of graphs on V:*

$$\liminf_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in D\right) \geq - \inf_{x \in D^\circ} I_{i,\mathcal{H}}(x), \tag{26}$$

   where $I_{i,\mathcal{H}}(x) = \overline{\text{co}} \inf\left\{|C_{i,\mathcal{H}}| I(x) + \left|\log p_{\mathcal{H}}\right|, NI(x)\right\}$.

In words, Theorem 13 asserts that, for a fixed set D, for any node i, the probabilities $\mathbb{P}\left(X_{i,t} \in D\right)$ decay exponentially fast over iterations t and it also finds bounds on the rate of this decay. We now make a couple of additional remarks and such that aim at gaining further insights and intuition about this result and the relevant quantities.

Remark 14. *Consider an arbitrary disconnected collection $\mathcal{H}$. By the construction of $C_{i,\mathcal{H}}$, for any node i, there holds $\{i\} \subseteq C_{i,\mathcal{H}}$ and, by non-negativity of I, it follows that $I \leq |C_{i,\mathcal{H}}| I$ (point-wise). Further, from Theorem 5 we know that $\mathcal{J} = |\log p_{\mathcal{H}^\star}| \leq |\log p_{\mathcal{H}}|$. Therefore, we have that for any disconnected* *collection $\mathcal{H}$, $I + \mathcal{J} \leq |C_{i,\mathcal{H}}| I + |\log p_{\mathcal{H}}|$. The latter obviously implies $I^\star \leq I_{i,\mathcal{H}}$, serving as a first feasibility check for* (20) *(and also* (18) *and* (19)*).*

Comparing the upper bound from Theorem 13 with (18), we see that (18) is satisfied for

$$\underline{I}_i \equiv I^\star, \text{ for all } i \in V. \tag{27}$$

That is, we have a uniform (lower) bound $I^\star$ on each of the nodes' rate functions I_i, $i \in V$.

With respect to the lower bound from Theorem 13, there is in fact a whole family of functions $\overline{I}_i$, one per each collection of graphs $\mathcal{H}$, that validate (19). To find the best bound for a given D, we might optimize the right hand side of (26) over all collections $\mathcal{H}$. This, however, might be computationally infeasible. Instead, we can focus only on those collections $\mathcal{P} \subseteq \mathcal{G}$ that have a certain property, e.g., $\mathcal{P} = \{\mathcal{H} : |C_{i,\mathcal{H}}| = n\}$, for some n, $1 \leq n \leq N$. Then, $\overline{I}_i$ from (19) can be found by finding $\mathcal{H} \in \mathcal{P}$ that yields uniformly lowest (i.e., closest to I_i) $I_{i,\mathcal{H}}$:

$$\overline{I}_i = \inf_{\mathcal{H} \in \mathcal{P}} I_{i,\mathcal{H}}, \text{ for } i \in V. \tag{28}$$

The following corollary follows directly from (20) and the definition of LDP.

Corollary 15. 1) *If, for a given i, the sequence $X_{i,t}$, $t = 1, 2, \ldots$ satisfies the LDP with rate function I_i, then, for any collection of graphs $\mathcal{H}$,*

$$I^\star \leq I_i \leq I_{i,\mathcal{H}}. \tag{29}$$

2) *If, for a given i, for some $\mathcal{P}$ (possibly, a single element set $\mathcal{P} = \{\mathcal{H}\}$), $I^\star \equiv \inf_{\mathcal{H} \in \mathcal{P}} I_{i,\mathcal{H}}$, then the sequence $X_{i,t}$, $t = 1, 2, \ldots$ satisfies the LDP with rate function $I_i = I^\star \equiv \inf_{\mathcal{H} \in \mathcal{P}} I_{i,\mathcal{H}}$.*

In the next remark, through simple convex analyses, we make a connection between Corollary 15 (Theorem 13) and Lemma 11, completing the established bounds in (29) with the general bounds from Lemma 11, hence establishing a coherent view of the derived results.

Remark 16 (Recovery of fundamental bounds in Lemma 11)**.** *From the point-wise non-negativity of I and non-negativity of $\mathcal{J}$, it is easy to see that $I \leq NI$ and $I \leq I + \mathcal{J}$. Thus,* $\operatorname{epi}\inf\{NI, I + \mathcal{J}\} \subseteq \operatorname{epi}I$. *Since I is closed and convex,* $\overline{\operatorname{co}}\operatorname{epi}I = \operatorname{epi}I$, *thus implying* $\overline{\operatorname{co}}\operatorname{epi}\inf\{NI, I + \mathcal{J}\} \subseteq \operatorname{epi}I$. *The latter directly implies $I \leq I^\star$. Similarly, we have $NI \geq \inf\{NI, |C_{i,\mathcal{H}}| I + |\log p_{\mathcal{H}}|\}$, where the latter holds for any disconnected collection $\mathcal{H}$. Thus* $\operatorname{epi}NI \subseteq \operatorname{epi}\inf\{NI, |C_{i,\mathcal{H}}| I + |\log p_{\mathcal{H}}|\}$, *which in turn implies* $\overline{\operatorname{co}}\operatorname{epi}NI \subseteq \overline{\operatorname{co}}\operatorname{epi}\inf\{NI, |C_{i,\mathcal{H}}| I + |\log p_{\mathcal{H}}|\}$. *Since NI is convex and closed (the properties inherited* *from I),* $\overline{\mathrm{co}}\,\mathrm{epi}NI = \mathrm{epi}NI$*, and therefore* $\mathrm{epi}NI = \overline{\mathrm{co}}\,\mathrm{epi}NI \subseteq \overline{\mathrm{co}}\,\mathrm{epi}\inf\{NI, |C_{i,\mathcal{H}}|I + |\log p_{\mathcal{H}}|\}$*. The latter implies* $NI \geq I_{i,\mathcal{H}}$*. Combining with* (29) *establishes:*

$$I \leq I^\star \leq I_i \leq I_{i,\mathcal{H}} \leq NI. \tag{30}$$

The above chain of inequalities is a capture of the so far established bounds in the literature on the large deviations rate function for consensus+innovations distributed inference iterates on random networks.

As a byproduct, we note in passing that (30) *verifies Lemma 11 for the special case of stochastic symmetric weight matrices.*

Remark 17 (Zero rate at θ)**.** *Since I is non-negative, both NI and $|C_{i,\mathcal{H}}|I(\theta) + |\log p_H|$ are also non-negative, implying $I_{i,\mathcal{H}} \geq 0$. Further, from Lemma 10, we have $I(\theta) = 0$, and noting now that $NI(\theta) = 0 < |C_{i,\mathcal{H}}|I(\theta) + |\log p_H|$, it follows that $I_{i,\mathcal{H}}(\theta) = 0$. It can be similarly shown that $I^\star(\theta) = 0$. From the preceding properties it follows that for any set C containing the mean value θ*

$$\inf_{x\in C} I^\star(x) = \inf_{x\in C} I_{i,\mathcal{H}}(x) = 0. \tag{31}$$

It follows that $\mathbf{I}_i(C) = 0$, i.e., the inaccuracy rate for any C containing θ equals zero. This means that probabilities of events that $X_{i,t}$ belong to C do not exhibit an exponential decay – specifically, for any norm ball centered at θ, and of an arbitrary radius $\rho > 0$, $B_\theta(\rho) > 0$, there holds

$$\lim_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in B_\theta(\rho)\right) = 0. \tag{32}$$

Observing the form of the algorithm, eq. (3)-(4)*, where innovations $Z_{i,t}$ – the mean vector of which is θ, are incorporated and mixed via weighted averaging (both over time and across nodes), it is intuitive to expect that $X_{i,t}$ will converge to θ (consider the ideal averaging case – $W_t = J_d$, for which $X_{i,t} = \frac{1}{t}\sum_{s=1}^{t}\sum_{j=1}^{N}\frac{1}{N}Z_{j,s}$, which converges to θ by the law of large numbers). Hence, the zero decay in* (32) *is intuitive, i.e., the probabilities that $X_{i,t}$ belongs to a neighborhood of θ should not vanish with t.*

We use the result of Theorem 13, together with the uniqueness of the minimizer of I, property 5 from Lemma 10, to establish a sort of a converse to (32) - i.e., whenever we seek the inaccuracy rate $\mathbf{I}_i(C)$ for a set C not containing θ, this rate will be strictly positive. Practical relevance of this (technical) property is given in Theorem 19 below, where almost sure convergence of $X_{i,t}$ to θ is formally established.

Remark 18 (Strictly non-zero rate at $x \neq \theta$)**.** *Consider an arbitrary point $x \neq \theta$. From Lemma 10, part 5 we know that $I(x) > 0$ for any $x \neq \theta$.*

Consider now an arbitrary set C such that $\theta \notin C$. By strict convexity of I and uniqueness of the minimizer of I, it follows that I is coercive [33]. Pick an arbitrary point $x_0 \in C$ and let $\alpha = I(x_0)$.

Define $S_\alpha = \{x \in \mathbb{R}^d : I(x) \leq \alpha\}$, i.e., S_α is the α-level set of I. By coercivity of I, it follows that S_α is compact. We now note

$$\inf_{x \in C} I(x) = \inf_{x \in C \cap S_\alpha} I(x) =: a. \tag{33}$$

Compactness of S_α implies compactness of $C \cap S_\alpha$ and since I is continuous and strictly greater than 0*, it follows by the Weierstrass theorem that the infimum of I over C is strictly greater than zero,* $a = \inf_{x \in C \cap S_\alpha} I(x) > 0$*. Finally, By the fact that $I^\star \geq I$ (the left-hand side inequality in* (30)*), we in turn obtain:*

$$\inf_{x \in C} I^\star(x) \geq \inf_{x \in C} I(x) = a > 0. \tag{34}$$

Therefore, for any set C such that $\theta \notin C$, we have

$$\limsup_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in C\right) \leq -a < 0, \tag{35}$$

where the constant a bounding the exponential decay rate depends on the chosen set C.

With preceding considerations at hand, almost sure convergence of nodes' iterates $X_{i,t}$ follows by standard arguments.

Theorem 19 (Almost sure convergence of $X_{i,t}$)**.** *Consider distributed inference algorithm* (3)-(4) *under Assumptions 1 and 9. Then, for each node i, the state vectors $X_{i,t}$ converge almost surely to $\theta = E[Z_{i,t}]$.*

Proof. Fix node $i \in V$. Pick an arbitrary $\epsilon > 0$ and consider $C = \mathbb{B}^{\mathrm{c}}_\theta(\epsilon)$. We start by noting that inequality in (35) implies existence of a finite $t_0 = t_0(C)$ such that, for all $t \geq t_0$, $\mathbb{P}(X_{i,t} \in C) \leq e^{-t\frac{a}{2}}$. Then, for all $t \geq t_0$, we have

$$\mathbb{P}\left(\|X_{i,t} - \theta\| \geq \epsilon\right) \leq e^{-t\frac{a}{2}}. \tag{36}$$

Thus,

$$\mathbb{P}\left(\|X_{i,t} - \theta\| > \epsilon, \text{ i.o.}\right) \leq \sum_{t=1}^{\infty} e^{-t\frac{a}{2}} < \infty, \tag{37}$$

where the last inequality follows from strict positivity of a. Applying the Borel-Cantelli lemma [34], the claim of the theorem follows. $\square$

A. A closer look at functions $I^\star$ and $I_{i,\mathcal{H}}$

This subsection finds closed form expressions for the functions $I^\star$ and $I_{i,\mathcal{H}}$ for the case when $Z_{i,t}$ is a Gaussian vector, and provides a graphical interpretation of the obtained result.

Lemma 20. *Let $Z_{i,t}$ be Gaussian with mean vector m and covariance matrix S. Then*

$$I^\star(x) = \begin{cases} NI(x), & x \in \mathcal{R}_1^\star \\ N\sqrt{2c_1}\,H(x) - Nc_1, & x \in \mathcal{R}_2^\star \\ I(x) + \mathcal{J}, & x \in \mathcal{R}_3^\star \end{cases}, \tag{38}$$

where $\mathcal{R}_1^\star = \{x : NI(x) \le c_1\}$, $\mathcal{R}_2^\star = \{x : c_1 < I(x) \le Nc_1\}$, and $\mathcal{R}_3^\star = \{x : I(x) > Nc_1\}$, $I(x) = \frac{1}{2}(x-m)^\top S^{-1}(x-m)$, $H(x) = \sqrt{(x-m)^\top S^{-1}(x-m)}$, and $c_1 = \frac{\mathcal{J}}{N(N-1)}$. Also, for any fixed collection of graphs $\mathcal{H}$

$$I_{i,\mathcal{H}}(x) = \begin{cases} NI(x), & x \in \mathcal{R}_1^{i,\mathcal{H}} \\ N\sqrt{2c_2}\,H(x) - Nc_2, & x \in \mathcal{R}_2^{i,\mathcal{H}} \\ |C_{i,\mathcal{H}}|\,I(x) + |\log p_{\mathcal{H}}|, & x \in \mathcal{R}_3^{i,\mathcal{H}} \end{cases}, \tag{39}$$

where $\mathcal{R}_1^{i,\mathcal{H}} = \left\{x : \frac{N}{|C_{i,\mathcal{H}}|} I(x) \le c_2\right\}$, $\mathcal{R}_2^{i,\mathcal{H}} = \left\{x : c_2 < I(x) \le \frac{N}{|C_{i,\mathcal{H}}|}c_2\right\}$, $\mathcal{R}_3^{i,\mathcal{H}} = \left\{x : I(x) > \frac{N}{|C_{i,\mathcal{H}}|}c_2\right\}$, and $c_2 = \frac{|C_{i,\mathcal{H}}||\log p_{\mathcal{H}}|}{N(N-|C_{i,\mathcal{H}}|)}$.

Proof of Lemma 20 is given in Appendix B.

Three regions of $I^\star$. We provide a graphical illustration for $I^\star$ in Figure 2. We consider an instance of algorithm (3)-(4) running on a $N = 3$-node chain, with i.i.d. link failures of probability $(1-p) = e^{-5}$, and where the observations $Z_{i,t}$ are standard Gaussian (zero mean and variance equal to one). For standard Gaussian, $I(x) = \frac{1}{2}x^2$, and we obtain from Example 6 that the rate of consensus equals $\mathcal{J} = |\log(1-p)| = 5$. Therefore the more curved blue dotted line plots the function $NI(x) = \frac{1}{2}Nx^2$, the less curved blue dotted line plots the function $I(x) + \mathcal{J} = \frac{1}{2}x^2 + 5$, and the solid red line plots $I^\star$. Observing the figure and the corresponding formula (38), we see that $I^\star$ is defined by three regions. In the region around the zero mean, $\mathcal{R}_1^\star$, $I^\star$ matches the optimal rate function NI. On the other hand, in the outer region, $\mathcal{R}_3^\star$, where values of x are sufficiently large, $I^\star$ follows the slower growing function, $I + \mathcal{J}$. Finally, in the middle region, $\mathcal{R}_2^\star$, $I^\star$ is linear (more generally, when $d > 1$, $I^\star$ will exhibit linear intervals over any direction that crosses the mean value). This linear part is the tangent line that touches both the epigraph of $NI(\cdot)$ and the epigraph of $I + \mathcal{J}$ and is responsible for the convexification of the point-wise infimum $\inf\{I + \mathcal{J}, NI\}$. Function $I_{i,\mathcal{H}}$ has similar properties.

B. Illustrations and LDP for special cases

In this subsection, we use Theorem 13 to establish the LDP for certain classes of random models. As explained in the remarks after Theorem 13, to prove the LDP at some node i, it is sufficient to show that $I^\star$ and $I_{i,\mathcal{H}}$ coincide for some collection $\mathcal{H}$.

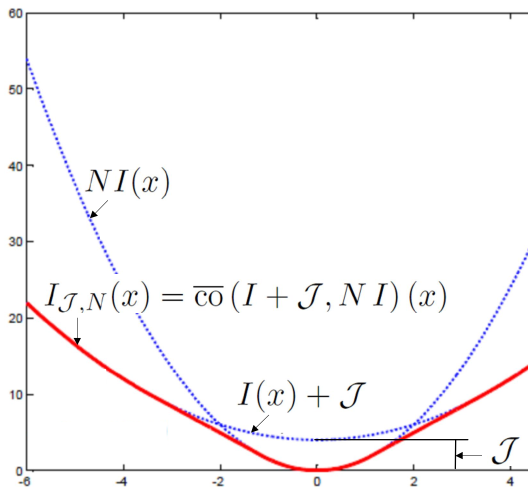


Fig. 2: Illustration of $I^\star$ for a chain network of size $N = 3$, with $\mathcal{J} = 5$, and $Z_{i,t} \sim \mathcal{N}(0,1)$. The more curved blue dotted line plots $NI(x) = \frac{1}{2}Nx^2$, the less curved blue dotted line plots $I(x) + \mathcal{J} = \frac{1}{2}x^2 + \mathcal{J}$. The solid red line plots $I^\star = \mathrm{co}\,(NI, I + \mathcal{J})$.

The first corollary of Theorem 13 asserts that if every realization of the network topology is connected, then, for any node i, the sequence of states $X_{i,t}$ satisfies the LDP with rate function NI. In our recent work [16], we prove that NI is the best (highest) possible rate function for any distributed inference algorithms of the form (3)-(4) with N nodes. It is also the rate function of a hypothetical fusion node that has access to all the observations. Thus, when every instance of the network topology is connected, then each node in the network is, in the asymptotic sense, effectively acts as a fusion center. Corollary 21 was, for the special case of Gaussian observations, previously proved in [35].

Corollary 21. *Let, for each t, G_t be connected. Then, for any $i \in V$, $X_{i,t}$ satisfies the large deviations principle with rate function NI.*

Proof. By Theorem 2 from [16], we know that, for any node i and for any set D

$$\liminf_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in D\right) \geq - \inf_{x \in D^\circ} NI(x). \tag{40}$$

Comparing with the conditions for LDP in Definition 8, we see that we only need to prove that $I^\star \equiv NI$. For the latter identity it suffices to show that $\mathcal{J} = +\infty$, because then $\inf\{NI, I + \mathcal{J}\} \equiv NI$, and since NI is closed and convex, we obtain $I^\star = \overline{\mathrm{co}}(NI) = NI$. Suppose for the sake of contradiction that there exists a disconnected collection of graphs $\mathcal{H}$ such that $p_{\mathcal{H}} > 0$. Then, there must be a graph $H \in \mathcal{H}$ such that both H is disconnected and $p_H > 0$. But this contradicts the assumption that every possible (i.e., non-zero probability) topology is connected. Thus, it must be that for every disconnected collection $p_{\mathcal{H}} = 0$ implying $\mathcal{J} = +\infty$, and proving the claim. $\square$

In particular, Corollary 21 implies that if the nodes' interactions are deterministic, i.e., $W_t \equiv A$, for some stochastic symmetric A, and A is such that $|\lambda_2(A)| < 1$, then, for each i, $X_{i,t}$ satisfy the LDP with the optimal rate function NI. This recovers the large deviations principle for deterministic networks, established in [16], for the special case of symmetric networks (cf. Theorem 1 in [16]).

LDP for critical nodes. Consider now a situation when there exists a node i such that $\mathcal{J} = |\log p_{i,\mathrm{isol}}|$, where $p_{i,\mathrm{isol}}$ denotes the probability that i operates in isolation due to network randomness, $p_{i,\mathrm{isol}} = \mathbb{P}\left(O_{i,t} = \emptyset\right)$. Comparing with Theorem 5, this means that the most likely way to disconnect $\overline{G}$ is to isolate i, i.e.,

$$p_{\max} = \sum_{H \in \mathcal{H}_{i,\mathrm{isol}}} p_H, \tag{41}$$

where $\mathcal{H}_{i,\mathrm{isol}} = \{H : p_H > 0,\ C_{i,H} = \{i\}\}$. Since $C_{i,\mathcal{H}_{i,\mathrm{isol}}} = \{i\}$, we have $\left|C_{i,\mathcal{H}_{i,\mathrm{isol}}}\right| = 1$. Consider now the lower bound in (26) for $\mathcal{H} = \mathcal{H}_{i,\mathrm{isol}}$. Noting that $|p_{\mathcal{H}_{i,\mathrm{isol}}}| = \mathcal{J}$, we see that the two functions $I^\star$ and $I_{i,\mathcal{H}_{i,\mathrm{isol}}}$ coincide, thus implying the LDP for node i. This is formally stated in the next corollary.

Corollary 22 (LDP for critical nodes)**.** *Suppose that for some i, $\mathcal{J} = |\log p_{i,\mathrm{isol}}|$. Then, the sequence of states $X_{i,t}$ satisfies the LDP with the rate function $\overline{\mathrm{co}}\,\{NI(x), I(x) + |\log p_{i,\mathrm{isol}}|\}$.*

In the next two corollaries we assume the random model from Assumption 1.2 where each link in the graph $\overline{G}$ fails independently with the same probability $1-p$, $p \in [0,1]$.

Corollary 23 (LDP for pendant nodes)**.** *Suppose that the random model for W_t is such that all links in $\overline{E}$ fail independently from each other with probability $1-p$. Then, for any node i whose degree in $\overline{G}$ is equal to one, its sequence of states $X_{i,t}$ satisfies the LDP with the rate function $\overline{\mathrm{co}}\,\{NI(x), I(x) + |\log(1-p)|\}$.*

Proof. Suppose that i is a degree one node. By Corollary 22, it suffices to show that $\mathcal{J} = |\log(1-p)|$. From Example 6, we know that $\mathcal{J}$ equals $|\log(1-p)|$ times the minimum edge cut of $\overline{G}$. In this case, minimum edge cut equals one (and is achieved, for instance, when the edge adjacent to i is removed from the network), which proves the result. $\square$

Corollary 24 (LDP for regular networks)**.** *Suppose that $\overline{G}$ is a circulant network in which each node is connected to $d/2$ nodes on the left and $d/2$ nodes on the right, where $d \leq N-1$ is even. We assume that each link, independently of all other links, fails with probability $1-p$. Then, for any node i its sequence of states $X_{i,t}$ satisfies the LDP with the rate function $\overline{\mathrm{co}}\,\{NI, I + d\log|1-p|\}$.*

Proof. Note that $p_{i,\mathrm{isol}} = (1-p)^d$ for any i. Hence, by Corollary 22, it suffices to show that $\mathcal{J} = d|\log(1-p)|$. Observing that the minimum cut in this case equals d, the result follows. □

V. Application to distributed hypothesis testing and social learning

In this subsection we show how results from Section IV can be used to characterize large deviations rates of distributed hypothesis testing and social learning that are run over random networks. We recall the algorithm and relevant quantities defined in Section II-A. We assume that the measurement distributions corresponding to the same hypothesis are equal across all nodes, i.e., when hypothesis $\mathbf{H}_m$ is true, the measurements at all nodes are drawn from the same distribution f_m: $Y_{i,t} \sim f_{i,m} \equiv f_m$, for all i.

Following the identified role of the vector of log-likelihood ratios $L_{i,t}$ as the innovation vector $Z_{i,t}$ in (3)-(4), we introduce the log-moment generating function Λ_M of $L_{i,t}$ at node i, when the measurements are drawn from f_M (hypothesis $\mathbf{H}_M$ is true):

$$\Lambda_M(\lambda) = \mathbb{E}\left[e^{\lambda^\top L_{i,t}} \middle| \mathbf{H} = \mathbf{H}_M \right] \tag{42}$$

$$= \mathbb{E}\left[e^{\sum_{m=1}^{M-1} \lambda_m \log \frac{f_m(Y_{i,t})}{f_M(Y_{i,t})}} \middle| \mathbf{H} = \mathbf{H}_M \right], \tag{43}$$

for $\lambda = (\lambda_1, ..., \lambda_{M-1})^\top \in \mathbb{R}^{M-1}$; we note that index M in Λ_M indicates the dependence on the assumed true distribution f_M. Similarly as in Section III, the conjugate of Λ_M is denoted by I_M. We assume that Λ_M satisfies Assumption 9.

A. Large deviations rates of the belief log-ratios

The following result follows as a direct application of Theorem 13 to the log-ratios $X_{i,t}$ of public beliefs, defined in Section II-A, eq. (13), $X_{i,t}^m = \frac{1}{t}\log\frac{b_{i,t}^m}{b_{i,t}^M}$, $m = 1, ..., M-1$.

Theorem 25. *Consider the social learning algorithm* (7)-(8) *under Assumptions 1 and 9, for* $\Lambda = \Lambda_M$. *Then, when* $\mathbf{H} = \mathbf{H}_M$, *for each node* i, *for any measurable set* D,

1)
$$\limsup_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in D\right) \leq -\inf_{x \in \overline{D}} I_M^\star(x), \tag{44}$$

where $I_M^\star(x) = \overline{\mathrm{co}}\, \inf\{I_M(x) + \mathcal{J}, N I_M(x)\}$*;*

2) *for any collection* $\mathcal{H}$ *of graphs on* V*:*

$$\liminf_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in D\right) \geq -\inf_{x \in D^{o}} I_{i,\mathcal{H};M}(x), \tag{45}$$

where $I_{i,\mathcal{H};M}(x) = \overline{\mathrm{co}}\, \inf\{|C_{i,\mathcal{H}}| I_M(x) + |\log p_{\mathcal{H}}|, N I_M(x)\}$.

Consequently, all considerations, corollaries and results from Section IV also carry over without any changes for the log-ratios $X_{i,t}$ of beliefs in social learning. In particular, the LDP results for regular networks and pendant nodes also carry over to the social learning setup.

Theorem 26 (Almost sure convergence of $X_{i,t}$ in social learning)**.** *Consider the social learning algorithm* (7)-(8) *under Assumptions 1 and 9, for $\Lambda = \Lambda_M$. Then, for each node i, for each $m = 1, ..., M-1$, $\frac{1}{t} \log \frac{b_{i,t}^m}{b_{i,t}^M}$ converges almost surely to $-D_{KL}(f_M || f_m) = -\mathbb{E}\left[\log \frac{f_m(Y_{i,t})}{f_M(Y_{i,t}} \middle| \mathbf{H} = \mathbf{H}_M \right]$.*

The result follows as a direct application of Theorem 19 for the case when the innovations $Z_{i,t}$ in (3)-(4) are instantiated by the log-likelihood ratios $L_{i,t}$ defined in (11), $L_{i,t}^m = \log \frac{f_m(Y_{i,t})}{f_M(Y_{i,t}}$, for $m = 1, ..., M-1$, and by recognizing that the expected value of $\log \frac{f_m(Y_{i,t})}{f_M(Y_{i,t}}$ under distribution f_M is the negative of the KL divergence between f_m and f_M.

To illustrate the setup and the relevant quantities, we consider the example of M scalar Gaussian distributions of different mean values and equal variances.

Example 27 (Gaussian case: different mean values and equal variances)**.** *Let $Y_{i,t}$ be Gaussian scalars, with mean value μ_m under hypothesis m, and (equal) variance σ^2. It is easy to show that, for this case, $L_{i,t}$ is computed as:*

$$L_{i,t} = \frac{1}{\sigma^2} (Y_{i,t} - \mu_M) d - D_{\mathrm{KL}}, \tag{46}$$

where $d = (d_1, ..., d_{M-1})^\top$, and each $d_m = \mu_m - \mu_M$ is the difference between the mean value for the m-th hypothesis and the mean value for the true hypothesis, and $D_{\mathrm{KL}} = (D_{\mathrm{KL},1}, ..., D_{\mathrm{KL},M-1})^\top$, where $D_{\mathrm{KL},m} = \frac{(\mu_m - \mu_M)^2}{2\sigma^2}$ is the KL divergence between the distribution f_m and the true distribution f_M, $m = 1, ..., M-1$. It is easy to see that, for each $i = 1, ..., N$ and each t, $L_{i,t}$ is Gaussian with mean vector $-D_{\mathrm{KL}}$ and covariance matrix $\frac{1}{\sigma^2} dd^\top$. Using the standard formula for the log-moment generating function of multivariate Gaussian distribution, we get:

$$\Lambda_M(\lambda) = -\lambda^\top D_{\mathrm{KL}} + \frac{(\lambda^\top d)}{2\sigma^2}. \tag{47}$$

Simple calculus shows that the conjugate function I_M is given by:

$$I_M(x) = \begin{cases} \frac{\zeta^2}{2\sigma^2}, & \text{if } x = \frac{\zeta}{2\sigma^2} d - D_{\mathrm{KL}}, \text{ for some } \zeta \in \mathbb{R} \\ +\infty, & \text{if } x + D_{\mathrm{KL}} \notin \operatorname{span}(d) \end{cases}. \tag{48}$$

Thus, I_M is essentially a one-dimensional quadratic function that changes only along the direction $-D_{KL} + \alpha d$, $\alpha \in \mathbb{R}$, while being equal to $+\infty$ in the rest of the $\mathbb{R}^d$ space. This is intuitive as the log-likelihood ratios for different m are coupled through a common (scalar) variable $Y_{i,t}$, and hence the events that vector $L_{i,t}$ lies outside of the line $-D_{KL} + \alpha d$ must have zero probability (and thus rate function equal to $+\infty$). The convex conjugates of I_M from Theorem 13, $I_M^\star$ and $I_{i,\mathcal{H};M}$, can be found similarly as in Section IV-A.

B. Large deviations rates for beliefs in social learning

For each $m = 1, .., M-1$, define $g_m : \mathbb{R}^{M-1} \mapsto \mathbb{R}$ as $g_m(x) = x_m - \max\{0, x_1, ..., x_{M-1}\}$, for $x \in \mathbb{R}^d$.

Theorem 28. *Consider the social learning algorithm* (7)-(8) *under Assumptions 1 and 9, for* $\Lambda = \Lambda_M$. *Then, for each node* $i \in V$ *and hypothesis* $m = 1, ..., M-1$, *for any given interval* $F \subseteq \mathbb{R}$*:*

1)
$$\limsup_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(\frac{1}{t}\log b_{i,t}^m \in F\right) \leq - \inf_{x: g_m(x)\in F} I_M^\star(x); \tag{49}$$

2) *for any disconnected collection* $\mathcal{H}$,
$$\liminf_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(\frac{1}{t}\log b_{i,t}^m \in F\right) \geq - \inf_{x: g_m(x)\in F} I_{i,\mathcal{H};M}(x). \tag{50}$$

The proof is very similar to the proof of Lemma 4 from [22]. The key distinction is that here full LDP for the log-ratios of the beliefs, $X_{i,t}$, is not available due to the complexity of the network model, and we have to work instead with the upper and the lower rate function bounds. However, the key arguments remain unaltered. For completeness, we provide the main steps of the proof in Appendix C.

The result in Theorem 28 is very general, as it holds for arbitrary distributions f_m, $m = 1, ..., M$, such that the log-moment generating function Λ_M satisfies Assumption 9; this is for example the case for Gaussian distributions from Example 27.

Remark 29. *It can be shown by carrying out the same analyses as in the proof of Theorem 28, that, if for some node* i *the sequence* $X_{i,t}$ *satisfies the LDP with rate function* I_i, *then, for each* $m = 1, ..., M$ *the sequence of log beliefs* $\log b_{i,t}^m$ *also satisfies the LDP with rate function*
$$R_{i,m}(z) = \inf_{x:\, g_m(x)=z} I_i(x), \tag{51}$$
for $x \in \mathbb{R}$.

We can see that, to find the large deviations rates of the beliefs, first the rate function I_i (or bounds on this function) for the log-belief ratios $X_{i,t}$ are found, and then the contraction principle is applied with functions g_m acting as the bridge between the two domains. This relation is established in [22] for static networks, but the same behaviour carries over to the general case, with the difference that the rate function of log-beliefs can differ across different nodes as a result of network randomness. To shed some light on function g_m, we revisit Example 27 for which we derive a closed form expression for g_m.

Example 30 (Computation of g_m for the Gaussian case)**.** *Consider the setup from Example 27. Recall that $g_m(x) = x_m - \max\{0, x_1, ..., x_{M-1}\}$ and also that $I_M(x) = +\infty$ outside of the line $-D_{KL} + \alpha d$, $\alpha \in \mathbb{R}$. Define also*

$$f(\zeta) = \max\{0, \zeta\frac{d_1}{\sigma^2} - D_{KL,1}, ..., \zeta\frac{d_{M-1}}{\sigma^2} - D_{KL,M-1}\}. \tag{52}$$

and note that

$$g_m(x) = \zeta\frac{d_m}{\sigma^2} - D_{KL,m} - f(\zeta), \tag{53}$$

for any $\zeta \in \mathbb{R}$ and $x \in \mathbb{R}^{M-1}$ such that $x = (\zeta\frac{d_1}{\sigma^2} - D_{KL,1}, ..., \zeta\frac{d_{M-1}}{\sigma^2} - D_{KL,M-1})$, for $m = 1, ..., M-1$.

Without loss of generality, assume that $\mu_1 < \mu_2 < ... < \mu_{M-1}$, implying also $d_1 < d_2 < ... < d_{M-1}$. Let $m^\star$ be the largest m such that $\mu_m < \mu_M$, $m^\star = \max\{m \in \{0, 1, ..., M-1\} : \mu_m < \mu_M\}$, wherein we additionally define $\mu_0 \equiv -\infty$ to account for the case that $\mu_M < \mu_1$. Then $d_1 < ... < d_m^\star < 0 < d_{m^\star+1} < ... < d_{M-1}$. By the preceding ordering, and exploiting also that $D_{KL,m} = \frac{d_m^2}{2\sigma^2}$, it can be easily verified that for any pair $l < m$, the intersection between the lines $\zeta\frac{d_l}{\sigma^2} - D_{KL,l}$ and $\zeta\frac{d_m}{\sigma^2} - D_{KL,m}$ occurs at $\frac{d_l+d_m}{2}$, with the l-indexed line dominating to the left of this point, for $\zeta < \frac{d_l+d_m}{2}$, while the m-indexed line dominates to the right. It also clearly follows that the first intersection point occurs for the first neighboring index, thus, as ζ increases, the lines must dominate in the same order as their d_m values. Summarizing, f is given in the following form:

$$f(\zeta) = \begin{cases} \zeta\frac{d_1}{\sigma^2} - D_{KL,1}, & \zeta < \frac{d_1+d_2}{2} \\ \zeta\frac{d_2}{\sigma^2} - D_{KL,2}, & \frac{d_1+d_2}{2} \le \zeta < \frac{d_2+d_3}{2} \\ 0 & \frac{d_{m^\star}}{2} \le \zeta < \frac{d_{m^\star+1}}{2} \\ ... & \\ \zeta\frac{d_m}{\sigma^2} - D_{KL,m}, & \frac{d_{m-1}+d_m}{2} \le \zeta < \frac{d_m+d_{m+1}}{2} \\ ... & \\ \zeta\frac{d_{M-1}}{\sigma^2} - D_{KL,M-1}, & \zeta \ge \frac{d_{M-2}+d_{M-1}}{2} \end{cases}. \tag{54}$$

From (53) *and* (54)*, we can obtain for $x = \frac{\zeta}{\sigma^2}d - D_{KL}$:*

$$g_m(x) = \begin{cases} \frac{(d_m - d_1)}{\sigma^2}\left(\zeta - \frac{d_m + d_1}{2}\right) & \zeta < \frac{d_1+d_2}{2} \\ \zeta\frac{d_2}{\sigma^2} - D_{KL,2} & \frac{d_1+d_2}{2} \leq \zeta < \frac{d_2+d_3}{2} \\ \dots & \\ \zeta\frac{d_m}{\sigma^2} - D_{KL,m} & \frac{d_{m^\star}}{2} \leq \zeta < \frac{d_{m^\star+1}}{2} \\ \dots & \\ 0 & \frac{d_{m-1}+d_m}{2} \leq \zeta < \frac{d_m + d_{m+1}}{2} \\ \dots & \\ \frac{(d_m - d_{M-1})}{\sigma^2}\left(\zeta - \frac{d_m + d_{M-1}}{2}\right) & \zeta \geq \frac{d_{M-2}+d_{M-1}}{2} \end{cases}. \tag{55}$$

The derived closed form expression for g_m is a step towards deriving the closed form expression for the rate function $R_{i,m}$, and, in particular, it suggests an analytical validation for the piece-wise behaviour of the rate function of beliefs discovered numerically in [22], Figure [illegible]. This is out of scope of the current paper and is left for future work. To provide an illustration towards characterizing $R_{i,m}$, we consider the value of the rate function at $-D_{KL,m}$. From (55)*, we see that $g_m(x) = -D_{KL,m}\}$ for $x = \frac{\zeta}{\sigma^2} - D_{KL}$ and $\zeta = 0$ (note that, by construction, $d_{m^\star} < 0$ and $d_{m^\star+1} > 0$), and hence $\zeta = 0 \in [\frac{d_{m^\star}}{2}, \frac{d_{m^\star+1}}{2})$). Thus, we have*

$$R_{i,m}(-D_{KL,m}) = \inf_{x: g_m(x) = -D_{KL,m}} I_i(x) \leq I_i(-D_{KL}). \tag{56}$$

the preceding inequality holds trivially by the fact that $-D_{KL} \in \{x : g_m(x) = -D_{KL,m}\}$. On the other hand, we have proved that $I^\star_M(-D_{KL}) = I_{i,\mathcal{H};M}(-D_{KL}) = 0$ (see Remark 17). By (29)*, we thus have $I_i(-D_{KL}) = 0$. It follows that $R_{i,m}(-D_{KL,m}) = 0$, i.e., the derived expression for g_m reveals that the value of the rate function $R_{i,m}$ at $-D_{KL,m}$ is zero. This is in accordance with almost sure convergence of $\frac{1}{t}\log b^m_{i,t}$ to $-D_{KL,m}$ which follows by combining Theorems 26 and 32.*

When the two functions from (44) and (45), namely, $I^\star_M$ and $I_{i,\mathcal{H};M}$ match, this implies that the corresponding $\limsup$ and the $\liminf$ are equal. Hence, whenever for a given node i its sequence $X_{i,t}$ exhibits LDP, this implies LDP for the sequence of beliefs $\frac{1}{t}\log b^m_{i,t}$, for each $m = 1, ..., M-1$. Here we give an example for regular networks.

Corollary 31 (LDP for social learning in regular networks)**.** *Suppose that $\overline{G}$ is a circulant network as in Corollary 24, i.e., each node is connected to $d/2$ nodes on the left and $d/2$ nodes on the right, where $d \leq N-1$ is even. We assume that each link, independently of all other links, fails with probability $1-p$. Then, for any node i, for each m, $\frac{1}{t}\log b^m_{i,t}$ satisfies the LDP with the rate function*

$$R_m(z) = \inf_{x \in \mathbb{R}^{M-1}: g_m(x) = z} \overline{\mathrm{co}}\left\{NI_M, I_M + d\log|1-p|\right\}(x). \tag{57}$$

A similar result holds also for pendant nodes with i.i.d. link failures.

Convergence to the correct hypothesis. The next result establishes, through the use of large deviations analysis, that the social learning algorithm (7)-(8) correctly identifies the true hypothesis. We remark that this recovers the result of [27] for the special case of identical distributions across nodes.

Theorem 32. *Consider the social learning algorithm* (7)-(8) *under Assumption 1 and 9, for* $\Lambda = \Lambda_M$*. Then, when* $\mathbf{H} = \mathbf{H}_M$*, for each node* i*, the sequence of beliefs* $b_{i,t}^M$ *converges to one almost surely.*

Proof. From the construction of the beliefs $b_{i,t}^m$, for each i, t, $b_{i,t}^M = 1 - b_{i,t}^1 - ... - b_{i,t}^{M-1}$. Combining this with the relations $X_{i,t}^m = \frac{1}{t}\log\frac{b_{i,t}^m}{b_{i,t}^M}$, yields

$$b_{i,t}^M = \frac{1}{1 + \sum_{m=1}^{M-1} e^{tX_{i,t}^m}}. \tag{58}$$

By Theorem 26, for each $m = 1, ..., M-1$, $X_{i,t}^m$ converges almost surely to $-D_{KL}(f_M||f_m) < 0$. Hence, each of the terms $e^{tX_{i,t}^m}$ in the sum above vanishes with probability one. Since M is finite, there exists a set of probability one such that $\sum_{m=1}^M e^{tX_{i,t}^m}$ vanishes, proving that $b_{i,t}^M$ converges to one almost surely. □

The next two sections prove Theorem 13; Section VI-A proves the upper bound (25) and Section VI-B proves the lower bound (26).

VI. Proof of Theorem 13

This section proves Theorem 13 by proving separately the upper and the lower bound. Before giving the respective proofs, we first give some important lemmas that are used in both the upper and the lower bound proof.

Lemma 33 will be used to find the log-moment generating function of the estimate $X_{i,t}$ from the log-moment generating functions of each of the terms in the sum (5). This result follows from convexity and zero value at the origin property of Λ.

Lemma 33. *For any set of convex multipliers* $\alpha \in \Delta_{N-1}$*, for each* $j = 1, ..., N$*, the log-moment generating function* Λ *satisfies,*

$$N\Lambda\left(1/N\lambda\right) \leq \sum_{i=1}^N \Lambda(\alpha_i\lambda) \leq \Lambda(\lambda), \tag{59}$$

for any $\lambda \in \mathbb{R}^d$*.*

The proof of Lemma 33 can be found in [16].

The claims in Lemma 34 are standard results from convex analysis, the proofs of which can be found, e.g., in [32]. Let the superscript * denote the conjugacy operation, i.e., for a given function $f : \mathbb{R}^d \mapsto \mathbb{R}$,

$$f^\star(x) = \sup_{s \in \mathbb{R}^d} s^\top x - f(s), \quad x \in \mathbb{R}^d. \tag{60}$$

The following relations hold between a function f and its conjugate $f^\star$.

Lemma 34. 1) *Let $f : \mathbb{R}^d \mapsto \mathbb{R}$ be a given a function. Then:*

   a) *$[f(\cdot) + r]^\star = f^\star(\cdot) - r$;*

   b) *for $\alpha > 0$ and $\beta \neq 0$, $[\alpha f(\beta(\cdot))]^\star = \alpha f^\star\left(1/(\alpha\beta)(\cdot)\right)$.*

2) *Let f_1 and f_2 be two given functions. Then, the conjugate of the pointwise supremum of f_1 and f_2 is the convex hull of the pointwise infimum of $f_1^\star$ and $f_2^\star$:*

$$[\sup\{f_1, f_2\}]^\star = \overline{\mathrm{co}} \inf\left\{f_1^\star, f_2^\star\right\}. \tag{61}$$

A. Proof of the upper bound (25)

In our previous work [16], we have proved that, at any node i, the sequence $X_{i,t}$ is exponentially tight. This intuitively means that the probabilities of the tail events of $X_{i,t}$ vanish sufficiently fast (i.e., the exponential rates of the tail probabilities grow unbounded when the tails move to infinity). Lemma 35 uses this result to derive an elegant sufficient condition for a certain function to satisfy the large deviations upper bound from Definition 8. In our case, this function will be the conjugate of a certain modification of Λ that accounts for the effects of intermittent communications. We remark that at the core of the proof of Lemma 35 is a modification of the finite cover argument from the proof of Cramér's theorem in $\mathbb{R}^d$ (see, e.g., [28]); the detailed proof of Lemma 35 is provided in Appendix D.

Lemma 35. *Let X_t be an arbitrary sequence of random variables where each X_t takes values in $\mathbb{R}^d$. Suppose that for some function f, for any measurable set D there holds*

$$\limsup_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(X_t \in D\right) \leq f(\lambda) - \inf_{x \in D} \lambda^\top x, \tag{62}$$

for any $\lambda \in \mathbb{R}^d$. Then, if f is finite for all $\lambda \in \mathbb{R}^d$, for any compact set F

$$\limsup_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(X_t \in F\right) \leq - \inf_{x \in F} f^\star(x), \tag{63}$$

where $f^\star$ is the conjugate of f. If in addition X_t is exponentially tight, then (63) *holds for any closed set F.*

Fix an arbitrary node $i \in V$. Replicating the steps of the proof of Theorem 5 from [14], we obtain that, for any measurable set D, and any fixed $\lambda \in \mathbb{R}^d$,

$$\begin{aligned}&\limsup_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in D\right)\\ &\leq \max\left\{N\Lambda\left(\frac{1}{N}\lambda\right), \Lambda(\lambda) - \mathcal{J}\right\} - \inf_{x\in D} \lambda^\top x.\end{aligned} \tag{64}$$

By Lemma 17 from [16], the sequence of estimates $X_{i,t}$ is exponentially tight. (We remark that this result is proven under more general assumptions on the weight matrices than assumed here.) Hence, to prove the upper bound (25), it only remains to show that $I^\star$ from Theorem 13 is the conjugate of $f(\lambda) := \max\left\{N\Lambda\left(1/N\lambda\right), \Lambda(\lambda) - \mathcal{J}\right\}$, $\lambda \in \mathbb{R}^d$. From part 2 of Lemma 34, we have that the conjugate of f is the closed convex hull of the infimum of the conjugates of $f_1(\lambda) := \lambda \mapsto N\Lambda\left(1/N\lambda\right)$ and $f_2(\lambda) := \lambda \mapsto \Lambda(\lambda) - \mathcal{J}$. Using the conjugacy rules from parts 1b and 1a of Lemma 34, we obtain that the respective conjugates of f_1 and f_2 are $NI(x)$, $x \in \mathbb{R}^d$, and $I(x) + \mathcal{J}$, $x \in \mathbb{R}^d$. The upper bound 25 follows by part 2 of Lemma 34.

B. Proof of the lower bound (26)

Fix an arbitrary node $i \in V$. Fix a collection of feasible graphs $\mathcal{H}$. To simplify the notation, we denote the component of i in $\mathcal{H}$, $C_{i,\mathcal{H}}$, by C. We also let M denote the number of nodes in C, $M = |C|$. For each fixed t, we define the family of events $\left\{\mathcal{E}_\theta^t : \theta \in [0,1]\right\}$, such that for any $\theta \in [0,1]$,

$$\begin{aligned}\mathcal{E}_\theta^t = \Big\{ G_s \in \mathcal{H},\ \lceil \theta t \rceil \leq s \leq t,\ \ &\|[\Phi(t, t-o_t)]_C - J_M\| \leq \frac{1}{t},\\ &\|\Phi(\lceil\theta t\rceil, \lceil\theta t\rceil - o_t) - J_N\| \leq \frac{1}{t}\Big\},\end{aligned} \tag{65}$$

where $o_t = \lceil \log t \rceil$; we recall that, for a square matrix A, A_C denotes the block of A corresponding to the intersection of columns and rows of A the indices of which belong to C. For convenience, we introduce $\mathcal{T}_\theta = \{\lceil\theta t\rceil, ..., t\}$.

Lemma 36. *Let θ be an arbitrary number in $[0,1]$. For any $\omega \in \mathcal{E}_\theta^t$,*

1) *for any $s \in \mathcal{T}_\theta$,*
$$[\Phi(t,s)]_{ij} = 0, \ \textit{for}\ \ j \notin C;$$

2) *for $t - o_t \geq s \geq \lceil\theta t\rceil$,*
$$\left|[\Phi(t,s)]_{ij} - \frac{1}{M}\right| \leq \frac{1}{t},\ \textit{for all}\ \ j \in C;$$

3) *for $\lceil\theta t\rceil - o_t \geq s \geq 1$,*
$$\left|[\Phi(t,s)]_{ij} - \frac{1}{N}\right| \leq \frac{1}{t},\ \textit{for all}\ \ j \in V.$$

Proof. Fix $\omega \in \mathcal{E}_\theta^t$ and, for $s = 1, ..., t$, denote $A_s = W_s(\omega)$. Consider first part 1, and suppose, without loss of generality, that $C = \{1, ..., M\}$. By construction of $\mathcal{E}_\theta^t$, none of the graphs that appear during $\mathcal{T}_\theta$ have links that connect C with the remaining part of the network $C^{\mathrm{c}} = V \setminus C$. Hence, each of the matrices A_s, $s \in \mathcal{T}_\theta$ has the following block diagonal form

$$A_s = \begin{bmatrix} [A_s]_C & 0_{M\times(N-M)} \\ 0_{M\times(N-M)} & [A_s]_{V\setminus C} \end{bmatrix}, \tag{66}$$

and the same structure is therefore preserved in their products $\Phi(t,s) = A_t \cdots A_s$, $s \in \mathcal{T}_\theta$, i.e.,

$$\Phi(t,s) = \begin{bmatrix} [A_t]_C \cdot \ldots \cdot [A_s]_C & 0_{M\times(N-M)} \\ 0_{M\times(N-M)} & [A_t]_{C^{\mathrm{c}}} \cdot \ldots \cdot [A_s]_{C^{\mathrm{c}}} \end{bmatrix}.$$

We next consider part 2. Since for an arbitrary matrix A, for any i, j there holds $|A_{ij}| \le \|A\|$, it is sufficient to show that $\|[\Phi(t,s)]_C - J_M\| \le 1/t$, for any fixed $s \in \mathcal{T}_\theta$ such that $s \le t - o_t$. By part 1, we know that for any $s_1, s_2 \in \mathcal{T}_\theta$, the C block of $\Phi(s_1, s_2)$ is computed as the product of blocks $[A_{s_1}]_C$ through $[A_{s_2}]_C$. Since each of these blocks is a symmetric, stochastic, M by M matrix, we have that $[\Phi(s_1,s_2)]_C$ is a doubly stochastic (M by M) matrix. Consider now a fixed $s \in \mathcal{T}_\theta$ such that $s \le t - o_t$. Factoring out $[\Phi(t,s)]_C$ as the product $[\Phi(t, t-o_t)]_C[\Phi(t-o_t-1, s)]_C$, and using the double-stochasticity of the latter two matrices, we obtain $[\Phi(t,s)]_C - J_M = ([\Phi(t,t-o_t)]_C - J_M)(\Phi(t-o_t-1,s)]_C - J_M)$. By construction of $\mathcal{E}_\theta^t$, the spectral norm of the first factor is not greater than $1/t$, while the double-stochasticity of $\Phi(t-o_t-1,s)]_C$ yields that the spectral norm of the second factor is not greater than 1. Using submultiplicativity of the spectral norm, the claim in part 2 follows:

$$\begin{aligned} &\|[\Phi(t,s)]_C - J_M\| \\ &\le \|[\Phi(t, t-o_t)]_C - J_M\| \, \|[\Phi(t-o_t-1,s)]_C - J_M\| \\ &\le 1/t. \end{aligned} \tag{67}$$

Part 3 can be proven by factoring out $\Phi(t,s)$ as the product $\Phi(t, \lceil\theta t\rceil)\Phi(\lceil\theta t\rceil - 1, \lceil\theta t\rceil - o_t)\Phi(\lceil\theta t\rceil - o_t - 1, s)$ and applying similar arguments as in the proof of part 2. $\square$

Fix $\theta \in [0,1]$ and consider the probability distribution $\nu_t^\theta : \mathcal{B}\left(\mathbb{R}^d\right) \to [0,1]$ defined by

$$\nu_t^\theta(D) = \frac{\mathbb{P}\left(\{X_{i,t} \in D\} \cap \mathcal{E}_\theta^t\right)}{\mathbb{P}\left(\mathcal{E}_\theta^t\right)}, \tag{68}$$

that is, ν_t^θ is the probability distribution of $X_{i,t}$ conditioned on the event $\mathcal{E}_\theta^t$ (we note that $\mathbb{P}\left(\mathcal{E}_\theta^t\right) > 0$ for t sufficiently large, as we show later in the proof, see Lemma 38 further ahead).

Let Υ_t be the (normalized) logarithmic moment generating function associated with ν_t^θ,

$$\Upsilon_t(\lambda) = \frac{1}{t} \log \mathbb{E}\left[e^{t\lambda^\top X_{i,t}} \left| \mathcal{E}_\theta^t \right.\right], \text{ for } \lambda \in \mathbb{R}^d. \tag{69}$$

Using the properties of entries of $\Phi(t,s)$ for different intervals on s listed in Lemma 36, we establish in Lemma 37 that the sequence of functions Υ_t has a point-wise limit for every $\lambda \in \mathbb{R}^d$. This will allow to apply the Gärtner-Ellis theorem [28] to compute the large deviations rate function for the sequence of measures ν_t^θ. We first state and prove Lemma 37.

Lemma 37. *For any $\lambda \in \mathbb{R}^d$ and any $\theta \in [0,1]$:*

$$\lim_{t\to+\infty} \Upsilon_t(\lambda) = (1-\theta) M \Lambda\left(\frac{1}{M}\lambda\right) + \theta N \Lambda\left(\frac{1}{N}\lambda\right), \tag{70}$$

where, we recall, $M = |C|$.

Proof. Fix $\theta \in [0,1]$, $\lambda \in \mathbb{R}^d$. We have:

$$\begin{aligned}\mathbb{E}\left[e^{t\lambda^\top X_{i,t}} \,|\mathcal{E}_\theta^t\right] &= \frac{1}{\mathbb{P}\left(\mathcal{E}_\theta^t\right)}\mathbb{E}\left[1_{\mathcal{E}_\theta^t} e^{t\lambda^\top X_{i,t}}\right]\\ &= \frac{1}{\mathbb{P}\left(\mathcal{E}_\theta^t\right)}\mathbb{E}\left[\mathbb{E}\left[1_{\mathcal{E}_\theta^t} e^{t\lambda^\top X_{i,t}} |W_1,...,W_t\right]\right]\\ &= \frac{1}{\mathbb{P}\left(\mathcal{E}_\theta^t\right)}\mathbb{E}\left[1_{\mathcal{E}_\theta^t}\mathbb{E}\left[e^{t\lambda^\top X_{i,t}}|W_1,...,W_t\right]\right],\end{aligned} \tag{71}$$

where in the last equality we used that the indicator $1_{\mathcal{E}_\theta^t}$ is a function of $W_1,...,W_t$. Further, as the summands in (5) are independent given $W_1,...,W_t$, we obtain

$$\mathbb{E}\left[e^{t\lambda^\top X_{i,t}}|W_1,...,W_t\right] = e^{\sum_{s=1}^{t}\sum_{j=1}^{N}\Lambda([\Phi(t,s)]_{ij}\lambda)}. \tag{72}$$

Consider now a fixed $\omega \in \mathcal{E}_\theta^t$. We split the sum in the exponent of (72) according to the intervals used in the construction of $\mathcal{E}_\theta^t$. With this in mind, we define also

$$\overline{\chi}_t := \max_{\alpha\in[1/M-1/t,1/M+1/t]} \Lambda\left(\alpha\lambda\right), \tag{73}$$

$$\underline{\chi}_t := \min_{\alpha\in[1/M-1/t,1/M+1/t]} \Lambda\left(\alpha\lambda\right), \tag{74}$$

and

$$\overline{\zeta}_t := \max_{\alpha\in[1/N-1/t,1/N+1/t]} \Lambda\left(\alpha\lambda\right), \tag{75}$$

$$\underline{\zeta}_t := \min_{\alpha\in[1/N-1/t,1/N+1/t]} \Lambda\left(\alpha\lambda\right), \tag{76}$$

for $\lambda \in \mathbb{R}^d$. We remark that, by the continuity of Λ and compactness of the intervals, in each of the preceding optimization problems there exists a maximizer. Further, as $t \to +\infty$, the corresponding intervals shrink to a single point, and by using again continuity of Λ, we obtain that $\overline{\chi}_t, \underline{\chi}_t \to \Lambda\left(1/M\lambda\right)$, and $\overline{\zeta}_t, \underline{\zeta}_t \to \Lambda\left(1/N\lambda\right)$, as $t \to +\infty$. Then, by part 1 of Lemma 36 and the fact that $\Lambda(0) = 0$, we have

$$\sum_{j\notin C} \Lambda\left([\Phi(t,s)]_{ij}\lambda\right) = 0, \;\text{ for each } s \in \mathcal{T}_\theta.$$

Further, by part 2 of Lemma 36

$$M\underline{\chi}_t \le \sum_{j\in C} \Lambda\left([\Phi(t,s)]_{ij}\lambda\right) \le M\overline{\chi}_t, \text{ for } t-o_t \ge s \ge \lceil \theta t\rceil,$$

and, similarly, by part 3 of Lemma 36

$$N\underline{\zeta}_t \le \sum_{j=1}^{N} \Lambda\left([\Phi(t,s)]_{ij}\lambda\right) \le N\overline{\zeta}_t, \text{ for } \lceil \theta t\rceil - o_t \ge s \ge 1.$$

As for the summands in the intervals $\{t,...,t-o_t\}$ and $\{\lceil\theta t\rceil,...,\lceil\theta t\rceil - o_t\}$, we apply Lemma 33 to get

$$M\Lambda\left(\frac{1}{M}\lambda\right) \le \sum_{j\in C} \Lambda\left([\Phi(t,s)]_{ij}\lambda\right) \le \Lambda(\lambda),$$

$$\text{for } t \ge s \ge t-o_t,$$

and

$$N\Lambda\left(1/N\lambda\right) \le \sum_{j=1}^{N} \Lambda\left([\Phi(t,s)]_{ij}\lambda\right) \le \Lambda(\lambda),$$

$$\text{for } \lceil\theta t\rceil \ge s \ge \lceil\theta t\rceil - o_t.$$

Summing out the upper and lower bounds over all s in the preceding five inequalities yields:

$$t\,\underline{\Upsilon}_t\left(\lambda\right) \le \sum_{s=1}^{t}\sum_{j=1}^{N} \Lambda\left([\Phi(t,s)]_{i,j}\right) \le t\,\overline{\Upsilon}_t\left(\lambda\right), \tag{77}$$

where

$$\underline{\Upsilon}_t(\lambda) = \frac{\lceil\theta t\rceil - o_t}{t} N\underline{\zeta}_t + \frac{o_t}{t}\left(N\Lambda\left(\frac{1}{N}\lambda\right) + M\Lambda\left(\frac{1}{M}\lambda\right)\right)$$
$$+ \frac{t-\lceil\theta t\rceil - o_t}{t} M\underline{\chi}_t,$$

and

$$\overline{\Upsilon}_t(\lambda) = \frac{\lceil\theta t\rceil - o_t}{t} N\overline{\zeta}_t + \frac{o_t}{t}\left(N\Lambda\left(\frac{1}{N}\lambda\right) + M\Lambda\left(\frac{1}{M}\lambda\right)\right)$$
$$+ \frac{t-\lceil\theta t\rceil - o_t}{t} M\overline{\chi}_t.$$

The inequalities in (77) hold for any fixed $\omega \in \mathcal{E}_\theta^t$. Thus,

$$1_{\mathcal{E}_\theta^t} e^{t\underline{\Upsilon}_t(\lambda)} \le 1_{\mathcal{E}_\theta^t}\mathbb{E}\left[e^{t\lambda^\top X_{i,t}}|W_1,...,W_t\right] \le 1_{\mathcal{E}_\theta^t} e^{t\overline{\Upsilon}_t(\lambda)}. \tag{78}$$

Finally, by monotonicity of the expectation:

$$\mathbb{P}\left(\mathcal{E}_\theta^t\right) e^{t\underline{\Upsilon}_t(\lambda)} \le \mathbb{E}\left[1_{\mathcal{E}_\theta^t}\mathbb{E}\left[e^{t\lambda^\top X_{i,t}}|W_1,...,W_t\right]\right]$$
$$\le \mathbb{P}\left(\mathcal{E}_\theta^t\right) e^{t\overline{\Upsilon}_t(\lambda)},$$

which combined with (71) implies

$$e^{t\underline{\Upsilon}_t(\lambda)} \leq \mathbb{E}\left[e^{t\lambda^\top X_{i,t}}|\mathcal{E}_\theta^t\right] \leq e^{t\overline{\Upsilon}_t(\lambda)}. \tag{79}$$

Now, taking the logarithm and dividing by t,

$$\underline{\Upsilon}_t(\lambda) \leq \Upsilon_t\,(\lambda) \leq \overline{\Upsilon}_t(\lambda),$$

and noting that

$$\begin{aligned}\lim_{t\to+\infty} \overline{\Upsilon}_t(\lambda) &= \lim_{t\to+\infty} \underline{\Upsilon}_t(\lambda)\\ &= (1-\theta)M\Lambda\left(\frac{1}{M}\lambda\right) + \theta N\Lambda\left(\frac{1}{N}\lambda\right),\end{aligned}$$

the claim of Lemma 37 follows. □

By the Gärtner-Ellis theorem it follows then that the sequence of measures ν_t^θ satisfies the large deviations principle[5], with the rate function equal to the conjugate of

$$f_\theta(\lambda) := (1-\theta)M\Lambda\left(\frac{1}{M}\lambda\right) + \theta N\Lambda\left(\frac{1}{N}\lambda\right), \tag{80}$$

for $\lambda \in \mathbb{R}^d$. Therefore, for every open set $E \subseteq \mathbb{R}^d$, there holds

$$\liminf_{t\to+\infty} \frac{1}{t}\log\mathbb{P}\left(X_{i,t} \in E|\mathcal{E}_\theta^t\right) \geq -\inf_{x\in E}\left\{\sup_{\lambda\in\mathbb{R}^d} \lambda^\top x - f_\theta(\lambda)\right\}. \tag{81}$$

We next turn to computing the probability of the event $\mathcal{E}_\theta^t$.

Lemma 38. *For any $\theta \in [0,1]$, for all t sufficiently large:*

$$\frac{1}{4}p_{\mathcal{H}}^{t-\lceil\theta t\rceil} \leq \mathbb{P}\left(\mathcal{E}_\theta^t\right) \leq p_{\mathcal{H}}^{t-\lceil\theta t\rceil}. \tag{82}$$

Proof. By the disjoint blocks theorem [34] applied to the matrices in $\mathcal{T}_\theta$ and its complement $\{1,...,t\}\backslash\mathcal{T}_\theta$, we obtain

$$\begin{aligned}\mathbb{P}\left(\mathcal{E}_\theta^t\right) = \mathbb{P}&\left(\|\Phi(\lceil\theta t\rceil, \lceil\theta t\rceil - o_t) - J_N\| \leq \frac{1}{t}\right) \times\\ \mathbb{P}&\left(G_s \in \mathcal{H},\ \text{for } s\in\mathcal{T}_\theta,\ \|[\Phi(t,t-o_t)]_C - J_M\| \leq \frac{1}{t}\right).\end{aligned} \tag{83}$$

[5]We use here the variant of the Gärtner-Ellis theorem which claims the (full) LDP for the case when the domain of the limiting function is the whole space $\mathbb{R}^d$, as given in [28]; see also Exercise 2.3.20 in [28] for the statement and the sketch of the proof of this result.

We show using (17) that the first term in the right-hand side of the preceding equality goes to 1 as $t \to +\infty$. Fix an arbitrary $\epsilon \in (0,1)$. Then, for all t sufficiently large,

$$\begin{aligned} &\mathbb{P}\left(\|\Phi(\lceil \theta t \rceil, \lceil \theta t \rceil - o_t) - J_N\| \leq \frac{1}{t} \right) \\ &\geq 1 - K_\epsilon e^{-t(\mathcal{J}-\epsilon)} \geq 1/2. \end{aligned} \tag{84}$$

Clearly, being a probability, this term is also smaller than 1 (for all t). Consider now the second factor in the right-hand side of (83). Conditioning on the event $\{G_s \in \mathcal{H},\ \text{for } s \in \mathcal{T}_\theta\}$, and using the fact that the probability of this event equals $p_{\mathcal{H}}^{t-\lceil \theta t \rceil}$ (note that the latter holds by the independence of weight matrices, Assumption 1.2), we obtain

$$\begin{aligned} &\mathbb{P}\left(G_s \in \mathcal{H},\ \text{for } s \in \mathcal{T}_\theta,\ \|[\Phi(t, t-o_t)]_C - J_M\| \leq \frac{1}{t} \right) = \\ &\mathbb{P}\left(\|[\Phi(t, t-c_t)]_C - J_M\| \leq \frac{1}{t} | G_s \in \mathcal{H},\ \text{for } s \in \mathcal{T}_\theta \right) p_{\mathcal{H}}^{t-\lceil \theta t \rceil}. \end{aligned}$$

Similarly as in (84), it can be shown that the conditional probability term in (85), for all t sufficiently large, greater than $1/2$. On the other hand, it is obviously smaller than 1 for all t. Summarizing the preceding findings, the claim of the lemma follows. ☐

To bring the two key arguments together – Lemma [illegible] and the lower bound (81), we start from the following simple bound

$$\begin{aligned} \mathbb{P}(X_{i,t} \in E) &\geq \mathbb{P}\left(\{X_{i,t} \in E\} \cap \mathcal{E}_\theta^t\right) \\ &= \nu_t^\theta(E)\mathbb{P}\left(\mathcal{E}_\theta^t\right). \end{aligned} \tag{85}$$

From superadditivity of the lim inf, followed by an application of (81) and (82), we obtain

$$\begin{aligned} &\liminf_{t \to +\infty} \frac{1}{t} \log \mathbb{P}(X_{i,t} \in E) \\ &\geq \liminf_{t \to +\infty} \frac{1}{t} \log \nu_t^\theta(E) + \lim_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(\mathcal{E}_\theta^t\right) \\ &\geq -\inf_{x \in E} \left\{ \sup_{\lambda \in \mathbb{R}^d} \lambda^\top x - f_\theta(\lambda) \right\} - (1-\theta)|\log p_{\mathcal{H}}|. \end{aligned}$$

The preceding inequality holds for each θ in $[0,1]$. Optimizing over all such values yields:

$$\begin{aligned} &\liminf_{t \to +\infty} \frac{1}{t} \log \mathbb{P}(X_{i,t} \in E) \geq \\ &-\inf_{\theta \in [0,1]} \left\{ \inf_{x \in E} \sup_{\lambda \in \mathbb{R}^d} \left\{ \lambda^\top x - f_\theta(\lambda) \right\} + (1-\theta)|\log p_{\mathcal{H}}| \right\} \\ &= -\inf_{x \in E} \inf_{\theta \in [0,1]} \left\{ \sup_{\lambda \in \mathbb{R}^d} \left\{ \lambda^\top x - f_\theta(\lambda) \right\} + (1-\theta)|\log p_{\mathcal{H}}| \right\}. \end{aligned}$$

Now, fix $x \in E$ and consider the function

$$g(\theta,\lambda) := \lambda^\top x - \,(1-\theta)\left(M\Lambda\left(\frac{1}{M}\lambda\right) - |\log p_{\mathcal{H}}|\right)$$
$$- \,\theta N\Lambda\left(\frac{1}{N}\lambda\right). \tag{86}$$

As an affine function of θ, g is convex in θ. Further, by convexity of Λ, g is concave in λ, for any $\theta \in [0,1]$. Finally, sets $[0,1]$ and $\mathbb{R}^d$ are convex and set $[0,1]$ is compact. Thus, conditions for applying the Minimax theorem [36] are fulfilled and we obtain:

$$\inf_{\theta\in[0,1]} \sup_{\lambda\in\mathbb{R}^d} \lambda^\top x - (1-\theta)\left(M\Lambda\left(1/M\lambda\right) - |\log p_{\mathcal{H}}|\right)$$
$$- \,\theta N\Lambda\left(1/N\lambda\right) =$$
$$\sup_{\lambda\in\mathbb{R}^d} \inf_{\theta\in[0,1]} \lambda^\top x - (1-\theta)\left(M\Lambda\left(1/M\lambda\right) - |\log p_{\mathcal{H}}|\right)$$
$$- \,\theta N\Lambda\left(1/N\lambda\right)$$
$$= \sup_{\lambda\in\mathbb{R}^d} \lambda^\top x - \max\left\{M\Lambda\left(1/M\lambda\right) - |\log p_{\mathcal{H}}|, \Lambda\left(1/N\lambda\right)\right\}.$$

Similarly as in the proof of the upper bound, using the conjugacy rules from Lemma 34,

$$\sup_{\lambda\in\mathbb{R}^d} \lambda^\top x - \min\left\{M\Lambda\left(\frac{1}{M}\lambda\right) - |\log p_{\mathcal{H}}|, \Lambda\left(\frac{1}{N}\lambda\right)\right\}$$
$$= \overline{\mathrm{co}}\inf\left(NI, MI + |\log p_{\mathcal{H}}|\right)(x),$$

which finally yields,

$$\liminf_{t\to+\infty} \frac{1}{t}\log \mathbb{P}\left(X_{i,t}\in E\right)$$
$$\geq - \inf_{x\in E} \overline{\mathrm{co}}\inf\left\{NI, MI + |\log p_{\mathcal{H}}|\right\}(x).$$

This completes the proof of the lower bound and the proof of Theorem 13.

VII. Conclusion

We studied large deviations inaccuracy rates for consensus+innovations based distributed inference for generic random networks. We assume vector measurements with possibly non-i.i.d. entries. Our goal was to find bounds or exact rate function for each node in the network, accounting for the specificities of the node's interactions. For each node, we found a node-specific family of lower bounds, induced by the family of network subgraphs in which the node participates. Specifically, each bound in the family is given as the convex envelope of the centralized rate function and the effective rate function corresponding to a given subgraph, and lifted by the probability that this subgraph remains isolated from the remainder of the network. The upper bound is defined as the convex envelope of the centralized rate function and the rate function corresponding to an isolated node, lifted by the rate of consensus. We show that, for certain cases such as pendant nodes and d-cyclic graphs, the two bounds match, hence proving the large deviations principle for these classes of random networks. We illustrate the results with an application to social learning, providing also the first proof of the large deviations principle for social learning beliefs with random network models.

Appendix A
Proof of (20)

Fix $i \in V$ and suppose that the inequalities in (18) and (19) hold for any set D. Suppose also that the sequence of node i's states, $X_{i,t}$, satisfies the LDP with rate function I_i.

We prove (20) by contradiction. Consider first the right hand side of (20) and suppose, for the sake of contradiction, that there exists a point x_0 such that $I_i(x_0) > \overline{I}_i(x_0)$. Let $\epsilon = I_i(x_0) - \overline{I}_i(x_0)$ and introduce $S = \left\{x \in \mathbb{R}^d : I_i(x) > \overline{I}_i(x_0) + \epsilon/2\right\}$. By the lower semi-continuity of I_i, S is open. Also, $x_0 \in S$. Thus, for $\delta > 0$ sufficiently small, the closed ball $\overline{B}_{x_0}(\delta)$ entirely belongs to S. Combining the LDP upper bound (1) for $D = \overline{B}_{x_0}(\delta)$, with the bound (18) for $D = B_{x_0}(\delta)$, we obtain:

$$- \inf_{x \in B_{x_0}(\delta)} \overline{I}_i(x) \leq \liminf_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in B_{x_0}(\delta)\right) \tag{87}$$

$$\leq \limsup_{t \to +\infty} \frac{1}{t} \log \mathbb{P}\left(X_{i,t} \in \overline{B}_{x_0}(\delta)\right) \leq - \inf_{x \in \overline{B}_{x_0}(\delta)} I_i(x). \tag{88}$$

Since $\inf_{x \in B_{x_0}(\delta)} \overline{I}_i(x) \leq \overline{I}_i(x_0)$, we have that the left hand side in (87) is greater than $-\overline{I}_i(x_0)$. On the other hand, for any $x \in \overline{B}_{x_0}(\delta)$, $I_i(x) > \overline{I}_i(x_0) + \epsilon/2$, implying $\inf_{x \in \overline{B}_{x_0}(\delta)} I_i(x) \geq \overline{I}_i(x_0) + \epsilon/2$. This finally yields contradiction since the right hand side in (87) cannot be smaller than $-\overline{I}_i(x_0)$.

Appendix B
Proof of Lemma 20

We start by noting that $\operatorname{epi}\inf\{NI, I + \mathcal{J}\} = S_1 \cup S_2$, where S_1 and S_2 are the epigraphs of NI and $I + \mathcal{J}$, $S_1 = \operatorname{epi}(NI)$ and $S_2 = \operatorname{epi}(I + \mathcal{J})$. To prove Lemma 20, we need to show that $\operatorname{epi}F = \overline{\operatorname{co}}\{S_1 \cup S_2\}$, where F is the function defined in the right hand side of eq. (38). To do this it suffices to show that: 1) $\operatorname{epi}F$ is a convex set, and 2) $\operatorname{epi}F \subseteq \operatorname{co}(S_1 \cup S_2)$. We first prove 1). It suffices to show that F is convex, which we do using generalized second order characterizations of convex functions, e.g. [37]. Note that F is continuous and that $\mathcal{D}_F = \mathbb{R}^d$. For each x and d, let $F'_+(x,d)$ and $F''_+(x,d)$

denote, respectively, the upper directional derivatives of the first and the second order at the point x and in the direction d,

$$F'_+(x;d) = \limsup_{\epsilon \downarrow 0} \frac{F(x+\epsilon d) - F(x)}{\epsilon} \tag{89}$$

$$F''_+(x;d) = \limsup_{\epsilon \downarrow 0} \frac{F(x+\epsilon d) - F(x) - F'_+(x;d)}{2\,\epsilon^2}. \tag{90}$$

We will show that F is in fact differentiable. Then, by Theorem 2.1. part (i) from [37], proving convexity of F would reduce to proving that $F''_+(x;d) \geq 0$ for any x and d. Note that I and H are differentiable, with their respective gradients given by $\nabla I(x) = S^{-1}(x-m)$ and $\nabla H(x) = S^{-1}(x-m)/\sqrt{(x-m)^\top S^{-1} x - m}$. Thus, F is differentiable in each of the three open sets (note that I is continuous and differentiable): $\{x : I(x) < c_1\}$, $\{x : c_1 < I(x) < Nc_1\}$, and $\{x : NI(x) > c_1\}$. It remains to show that F is differentiable for those x such that $I(x) = c_1$ and $I(x) = Nc_1$. Fix first x such that $I(x) = c_1$. It is easy to see that, for any d such that $d^\top S^{-1}(x-m) \geq 0$, $I(x+\epsilon d) > I(x)$ for all $\epsilon > 0$. Also, for any d such that $d^\top S^{-1}(x-m) < 0$, $I(x+\epsilon d) < I(x)$ for all sufficiently small $\epsilon > 0$. Thus, if $d^\top S^{-1}(x-m) \geq 0$, $F(x+\epsilon d) = N\sqrt{2c_1}H(x+\epsilon d) - Nc_1$, for all ϵ sufficiently small, and hence $F'_+(x;d) = N\sqrt{2c_1} d^\top \nabla H(x)$. Using now the fact that $I(x) = c_1$, we obtain that $F'_+(x;d) = N d^\top S^{-1}(x-m)$. Consider now the case when d is such that $d^\top S^{-1}(x-m) \leq 0$. Then, by the discussion above we have that for all ϵ, $F(x+\epsilon d) = NI(x+\epsilon d)$. Hence, $F'_+(x;d) = N d^\top \nabla I(x) = N d^\top S^{-1}(x-m)$. Since for any x s.t. $I(x) = c_1$ and for any d we have that $F'_+(x;d) = N d^\top \nabla I(x)$, we conclude that F is differentiable at any such x. We can in analogous manner prove differentiability of F at any x s.t. $I(x) = Nc_1$. Hence, we conclude that F is differentiable.

We now turn to proving that $F''_+(x;d) \geq 0$ for any x and d. Note that $\nabla^2 I(x) = S^{-1} \succeq 0$ and

$$\nabla^2 H(x) =$$
$$N \frac{\sqrt{2c_1}}{\sqrt{2I(x)}} \left(S^{-1} - \frac{1}{2I(x)} S^{-1}(x-m)(x-m)^\top S^{-1} \right),$$

for any x. To see that $\nabla^2 H(x) \succeq 0$, it suffices to observe that it can be rewritten as $\nabla^2 H(x) = N\sqrt{2c_1}/\sqrt{2I(x)} S^{-1/2}(I - qq^\top/(\|q\|^2))S^{-1/2}$, for $q = S^{-1/2}(x-m)$. Since the matrix inside the brackets is positive semidefinite, positive semidefiniteness of $\nabla^2 H(x)$ follows. Therefore, for any x in the interior of the three sets in (38), we have that $F''_+(x;d) \geq 0$. Consider now the case when x satisfies $I(x) = c_1$. Following the same steps as in the preceding paragraph, we obtain that for any d s.t. $d^\top S^{-1}(x-m) \geq 0$, $F''_+(x;d) = N^2 2c_1 d^\top \nabla^2 H(x) d \geq 0$ and for d s.t. $d^\top S^{-1}(x-m) \leq 0$, $F''_+(x;d) = N d^\top \nabla^2 I(x) d \geq 0$. To complete the proof of 1), it only remains to consider those x that satisfy $I(x) = Nc_1$. Analogously to the preceding case, we get that for d s.t. $d^\top S^{-1}(x-m) \geq 0$, $F''_+(x;d) = d^\top \nabla^2 I(x) d \geq 0$ and for

d s.t. $d^\top S^{-1}(x-m) \leq 0$, $F''_+(x;d) = n^2 2c_1 d^\top \nabla^2 H(x) d \geq 0$. Hence, since F is differentiable and $F''_+(x;d) \geq 0$ for any x and d, we conclude that F is convex.

To prove Lemma 20, it remains to prove part 2). For each unit norm $v \in \mathbb{R}^d$, $\|v\|=1$, let $\phi_v : \mathbb{R}^d \mapsto \mathbb{R}^d$ denote the projection of F along the direction v, started at point m: $\phi_v(\rho) := F(m+\rho v)$, $\rho \in \mathbb{R}$. Then, $\mathrm{epi}F = \cup_{v\in\mathbb{R}^d, \|v\|=1} \mathrm{epi}\phi_v$. For each fixed v, let $[S_l]_v$ denote the projection of S_l along the line $m+\rho v$, $[S_l]_v = S_l \cap \{m+\rho v : \rho \in \mathbb{R}\}$, $l=1,2$. Note that $[S_1]_v = \{(t, m+\rho v) : \, t \geq N\rho^2 v^\top S^{-1} v/2, \, \rho \in \mathbb{R}\}$, $[S_2]_v = \{(t, m+\rho v) : \, t \geq \rho^2 v^\top S^{-1} v/2 + \mathcal{J}, \, \rho \in \mathbb{R}\}$. Then, it is easy to see that, for each unit norm v, $\mathrm{epi}\phi_v = \mathrm{co}\left([S_1]_v \cup [S_2]_v\right)$. Finally, since $\mathrm{co}\left([S_1]_v \cup [S_2]_v\right) \subseteq \mathrm{co}\left(S_1 \cup S_2\right)$, the claim in 2) follows. This completes the proof of Lemma 20.

APPENDIX C
PROOF OF LEMMA 28

Fix an arbitrary node $i \in V$. For each $m = 1, ..., M-1$, $X^m_{i,t} = \frac{1}{t}\log\frac{b^m_{i,t}}{b^M_{i,t}}$, hence

$$\frac{1}{t}\log b^m_{i,t} = X^m_{i,t} + \frac{1}{t}\log b^M_{i,t}. \tag{91}$$

Further, the (private) beliefs by construction sum up to one: $\sum_{m=1}^M b^m_{i,t} = 1$. Dividing both sides by $b^M_{i,t}$ and exploiting the functional relation between $b^m_{i,t}$ and $X^m_{i,t}$, we obtain

$$\sum_{m=1}^{M-1} e^{tX^m_{i,t}} + 1 = \frac{1}{b^M_{i,t}}. \tag{92}$$

It follows that:

$$\frac{1}{M} e^{-t\max_{m=1,\dots,M} X^m_{i,t}} \leq b^M_{i,t} \leq e^{-t\max_{m=1,\dots,M} X^m_{i,t}}, \tag{93}$$

where $X^M_{i,t} \equiv 0$. From (91) and (93) we obtain

$$g_m(X_{i,t}) - \frac{1}{t}\log M \leq \frac{1}{t}\log b^m_{i,t} \leq g_m(X_{i,t}). \tag{94}$$

Consider now an arbitrary one-sided closed interval F on $\mathbb{R}$. Suppose that $F = [a, +\infty)$ (other intervals in $\mathbb{R}$ can be treated analogously). Fix $\epsilon > 0$. From (94), for all $t \geq t_0 = \log M/\epsilon$ there holds:

$$g_m(X_{i,t}) - \epsilon \leq \frac{1}{t}\log b^m_{i,t} \leq g_m(X_{i,t}), \tag{95}$$

and thus, for all $t \geq t_0$

$$\mathbb{P}(\frac{1}{t}\log b^m_{i,t} \geq a + \epsilon) \leq \mathbb{P}(g_m(X_{i,t}) \geq a) = P(X_{i,t} \in g_m^{-1}([a, +\infty)). \tag{96}$$

Taking the $\limsup$ over $t \to +\infty$,

$$\limsup_{t\to+\infty} \frac{1}{t}\log \mathbb{P}(\frac{1}{t}\log b^m_{i,t} \geq a + \epsilon) \leq \limsup_{t\to+\infty} \frac{1}{t}\log P(X_{i,t} \in g_m^{-1}([a, +\infty)). \tag{97}$$

The above inequality holds for all $\epsilon > 0$. Taking the supremum over $\epsilon > 0$ on the left hand side yields:

$$\limsup_{t\to+\infty} \frac{1}{t} \log \mathbb{P}(\frac{1}{t} \log b_{i,t}^m \geq a) \leq \limsup_{t\to+\infty} \frac{1}{t} \log P(X_{i,t} \in g_m^{-1}([a,+\infty)). \tag{98}$$

Applying now the upper bound in 44, the upper bound in (49) follows. The proof of the lower bound (49) is analogous.

Appendix D
Proof of Lemma 35

Suppose that $X_t \in \mathbb{R}^d$ is a sequence of random variables for which (62) holds for some function f. Fix a compact set $F \subseteq \mathbb{R}^d$. For each $\delta > 0$, introduce the function $f^{\star,\delta} : \mathbb{R}^d \mapsto \mathbb{R}$ obtained by truncating $f^\star$ to $1/\delta$:

$$f^{\star,\delta}(x) = \inf \left\{ \frac{1}{\delta}, f^\star(x) - \delta \right\}, \text{ for } x \in \mathbb{R}^d. \tag{99}$$

The family of functions $f^{\star,\delta}$, $\delta > 0$, satisfies that, for any set D,

$$\lim_{\delta\to 0} \inf_{x\in D} f^{\star,\delta}(x) = \inf_{x\in D} f^\star(x). \tag{100}$$

To show this, let $\xi := \inf_{x\in D} f^\star(x)$ and suppose first that $\xi = +\infty$, i.e., $f^\star$ at all points $x \in D$ takes the value $+\infty$. Then, for any $\delta > 0$, $f^{\star,\delta} = 1/\delta$ for all $x \in D$, and therefore, for any $\delta > 0$, $\inf_{x\in D} f^{\star,\delta}(x) = 1/\delta$. Computing the limit $\lim_{\delta\to 0} 1/\delta = +\infty$, identity (100) follows. We next consider the case $\xi \in \mathbb{R}$. For arbitrary fixed $\delta > 0$, the quantity under the limit in the left hand side of (100) equals:

$$\begin{aligned}\inf_{x\in D} f^{\star,\delta}(x) &= \inf_{x\in D} \inf \left\{ f^\star(x) - \delta, \frac{1}{\delta} \right\} \\ &= \inf \left\{ \inf_{x\in D} \left(f^\star(x) - \delta\right), \frac{1}{\delta} \right\}.\end{aligned} \tag{101}$$

The first argument of the infimum (101) equals $\xi - \delta$ and it is finite by our assumption. Hence, for all δ sufficiently small, the infimum (101) equals $\xi - \delta$, which after taking the limit $\delta \to 0$ yields the claim. The case $\xi = -\infty$ can be proven equivalently.

Having (100), it easy to see that (63) follows if we show that the following inequality holds for any given δ:

$$\limsup_{t\to+\infty} \frac{1}{t} \log \mathbb{P}\left(X_t \in F\right) \leq 2\delta - \inf_{x\in F} f^{\star,\delta}(x). \tag{102}$$

Thus, in what follows we focus on proving (102). To this end, fix $\delta > 0$. For any point $y \in F$ there exists a point λ_y (which depends on δ) such that

$$\lambda_y^\top y - \Lambda^\star(\lambda_y) \geq f^{\star,\delta}(y). \tag{103}$$

Existence of such a point follows directly from the definitions of $f^\star$ and $f^{\star,\delta}$. First, since for any fixed point y $f^\star(y)$ is computed as the supremum of functions $\lambda \mapsto h_y(\lambda) := \lambda^\top y - f(\lambda)$, it follows that the value $f^\star(y)$ can be approached arbitrarily close with $h_y(\lambda)$. Second, since $f^{\star,\delta}(y)$ is the infimum of $f^\star(y) - \delta$ and $1/\delta$, it must satisfy $f^\star(y) - \delta,\, 1/\delta \geq f^{\star,\delta}(y)$. For example, if, for some y, $f^\star(y)$ is finite, then there must exist a point λ such that $h_y(\lambda) \geq f^\star(y) - \delta$, and since the latter is greater than $f^{\star,\delta}(y)$, (103) follows.

Note now that (62) implies that, for any measurable set D, there exists $t_0 = t_0(\delta, D)$ such that

$$\frac{1}{t}\log \mathbb{P}\left(X_t \in D\right) \leq \delta + f(\lambda) - \inf_{x \in D} \lambda^\top x, \tag{104}$$

for all $t \geq t_0$. For any $y \in F$, let $r_y := \delta/\|\lambda_y\|$. Taking $D = \overline{B}_y(r_y)$ and $\lambda = \lambda_y$ in (104) yields for any $t \geq t_0(\delta, y)$

$$\begin{aligned}\frac{1}{t}\log \mathbb{P}\left(X_t \in \overline{B}_y(r_y)\right) &\leq \delta + f(\lambda_y) - \inf_{\|x-y\|\leq r_y} \lambda_y^\top x &(105)\\ &\leq \delta + \Lambda^\star(\lambda_y) - \lambda_y^\top y - \inf_{\|x\|\leq r_y} \lambda_y^\top x &(106)\\ &\leq 2\delta - f^{\star,\delta}(y) &(107)\end{aligned}$$

where the last inequality follows from (103) and the definition of r_y. Next, from the family of closed balls $\left\{\overline{B}_y(r_y) : y \in F\right\}$, a finite cover of F, $\left\{\overline{B}_{y_k}(r_{y_k}) : k = 1, ..., K\right\}$, is extracted, where, we note, $K = K(F, \delta)$. Then, by the union bound,

$$\begin{aligned}\frac{1}{t}\log \mathbb{P}\left(X_t \in F\right) &\leq \frac{1}{t}\log\left(\sum_{k=1}^{K} \mathbb{P}\left(X_t \in \overline{B}_{y_k}(r_{y_k})\right)\right)\\ &\leq \frac{1}{t}\log K + \frac{1}{t}\log \max_{k=1,\ldots,K} \mathbb{P}\left(X_t \in \overline{B}_{y_k}(r_{y_k})\right)\\ &\leq \frac{1}{t}\log K + \max_{k=1,\ldots,K} \frac{1}{t}\log \mathbb{P}\left(X_t \in \overline{B}_{y_k}(r_{y_k})\right).\end{aligned}$$

Combining the preceding inequality with (105) applied for every $k = 1, ..., K$, we have that for every $t \geq \max_{k=1,\ldots K} t_0(\delta, y_k)$

$$\begin{aligned}\frac{1}{t}\log \mathbb{P}\left(X_t \in F\right) &\leq \frac{1}{t}\log K + \max_{k=1,\ldots,K} 2\delta - f^{\star,\delta}(y)\\ &\leq \frac{1}{t}\log K + 2\delta - \inf_{y \in F} f^{\star,\delta}(y). &(108)\end{aligned}$$

Taking the limit $t \to +\infty$, and noting that K is finite, (102) follows. The last part of the claim, i.e., (102) for closed sets follows from (102) for compact sets, that we have just proved, and Lemma 1.2.18 in [28].

References

[1] D. Bajović, "Large deviations rates for distributed inference," Ph.D. dissertation, Carnegie Mellon University, 2013.

[2] H. Chernoff, "A measure of the asymptotic efficiency of tests of a hypothesis based on a sum of observations," *The Annals of Mathematical Statistics*, vol. 23, no. 4, pp. 493–507, Dec. 1952.

[3] A. Anandkumar and L. Tong, "Type-based random access for distributed detection over multiaccess fading channels," *IEEE Transactions on Signal Processing*, vol. 55, no. 10, pp. 5032–5043, 2007.

[4] D. Bajović, B. Sinopoli, and J. Xavier, "Sensor selection for event detection in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4938–4953, 2011.

[5] W. P. Tay, "Whose opinion to follow in multihypothesis social learning? A large deviations perspective," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 2, pp. 344–359, 2015.

[6] P. Hu, V. Bordignon, S. Vlaski, and A. H. Sayed, "Optimal aggregation strategies for social learning over graphs," 2022. [Online]. Available: https://arxiv.org/abs/2203.07065

[7] J. A. Bucklew, *Large Deviations Techniques in Decision, Simulation and Estimation*. New York: Wiley, 1990.

[8] A. Shwartz and A. Weiss, *Large Deviations for Performance Analysis: Queues, Communications, and Computing*. New York: Chapman and Hall, 1995.

[9] H. Touchette, "The large deviation approach to statistical mechanics," *Physics Reports*, vol. 478, no. 1, pp. 1 – 69, 2009.

[10] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: John Wiley and Sons, 1991.

[11] M. Arcones, "Large deviations for M-estimators," *Annals of the Institute of Statistical Mathematics*, vol. 58, no. 1, pp. 21–52, 2006.

[12] R. R. Bahadur, "On the asymptotic efficiency of tests and estimates," *Sankhya: The Indian Journal of Statistics, 1933-1960*, vol. 22, no. 3/4, pp. 229–252, 1960. [Online]. Available: http://www.jstor.org/stable/25048458

[13] D. Bajović, D. Jakovetić, J. Xavier, B. Sinopoli, and J. M. F. Moura, "Distributed detection via Gaussian running consensus: Large deviations asymptotic analysis," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4381–4396, Sep. 2011.

[14] D. Bajović, D. Jakovetić, J. M. F. Moura, J. Xavier, and B. Sinopoli, "Large deviations performance of consensus+innovations distributed detection with non-Gaussian observations," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 5987–6002, Nov. 2012.

[15] D. Jakovetić, J. M. F. Moura, and J. Xavier, "Distributed detection over noisy networks: Large deviations analysis," *IEEE Transactions on Signal Processing*, vol. 60, no. 8, pp. 4306–4320, 2012.

[16] D. Bajović, J. M. F. Moura, J. Xavier, and B. Sinopoli, "Distributed inference over directed networks: Performance limits and optimal design," *IEEE Transactions on Signal Processing*, vol. 64, no. 13, pp. 3308–3323, July 2016.

[17] V. Matta, P. Braca, S. Marano, and A. H. Sayed, "Diffusion-based adaptive distributed detection: Steady-state performance in the slow adaptation regime," *IEEE Trans. Information Theory*, vol. 62, no. 8, pp. 4710–4732, August 2016.

[18] ——, "Distributed detection over adaptive networks: Refined asymptotics and the role of connectivity," *IEEE Trans. Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 442–460, Dec 2016.

[19] S. Marano and A. H. Sayed, "Detection under one-bit messaging over adaptive networks," *IEEE Trans. Information Theory*, vol. 65, no. 10, pp. 6519–6538, October 2019.

[20] A. Jadbabaie, P. Molavi, A. Sandroni, and A. Tahbaz-Salehi, "Non-Bayesian social learning," *Games and Economic Behavior*, vol. 76, no. 1, pp. 210 – 225, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0899825612000851

[21] S. Shahrampour, A. Rakhlin, and A. Jadbabaie, "Distributed detection: Finite-time analysis and impact of network topology," *IEEE Transactions on Automatic Control*, vol. 61, no. 11, pp. 3256–3268, 2016.

[22] A. Lalitha, T. Javidi, and A. D. Sarwate, “Social learning and distributed hypothesis testing,” *IEEE Transactions on Information Theory*, vol. 64, no. 9, pp. 6161–6179, 2018.

[23] A. Mitra, J. A. Richards, and S. Sundaram, “A new approach to distributed hypothesis testing and non-bayesian learning: Improved learning rate and byzantine resilience,” *IEEE Transactions on Automatic Control*, vol. 66, no. 9, pp. 4084–4100, 2021.

[24] M. H. DeGroot, “Reaching a consensus,” *Journal of American Statistical Association*, vol. 69, no. 345, pp. 118–121, 1974.

[25] A. Nedić, A. Olshevsky, and C. A. Uribe, “Fast convergence rates for distributed non-Bayesian learning,” *IEEE Transactions on Automatic Control*, vol. 62, no. 11, pp. 5538–5553, 2017.

[26] D. Bajović, J. Xavier, J. M. F. Moura, and B. Sinopoli, “Consensus and products of random stochastic matrices: Exact rate for convergence in probability,” *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2557–2571, May 2013.

[27] R. Parasnis, M. Franceschetti, and B. Touri, “Non-Bayesian social learning on random digraphs with aperiodically varying network connectivity,” 2020. [Online]. Available: https://arxiv.org/abs/2010.06695

[28] A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications*. Boston, MA: Jones and Barlett, 1993.

[29] D. Li, S. Kar, J. M. F. Moura, H. V. Poor, and S. Cui, “Distributed Kalman filtering over massive data sets: Analysis through large deviations of random Riccati equations,” *IEEE Transactions on Information Theory*, vol. 61, no. 3, pp. 1351–1372, March 2015.

[30] F. den Hollander, *Large Deviations*. Fields Institute Monographs. American Mathematical Society, 2000.

[31] V. Vysotsky, “When is the rate function of a random vector strictly convex?” *Electronic Communications in Probability*, vol. 26, no. none, pp. 1 – 11, 2021. [Online]. Available: https://doi.org/10.1214/21-ECP409

[32] J.-B. Hiriart-Urruty and C. Lemarechal, *Fundamentals of Convex Analysis*, ser. Grundlehren Text Editions. Berlin, Germany: Springer-Verlag, 2004.

[33] R. T. Rockafellar, *Convex Analysis*. Princeton University Press, 2015. [Online]. Available: https://doi.org/10.1515/9781400873173

[34] A. F. Karr, *Probability*, ser. Springer Texts in Statistics. New York: Springer-Verlag, 1993.

[35] D. Bajović, D. Jakovetić, J. M. F. Moura, J. Xavier, , and B. Sinopoli, “Large deviations analysis of consensus+innovations detection in random networks,” in *Allerton’11, 49th Allerton Conference on Communication, Control, and Computing*, Monticello, Il, October 2011.

[36] M. Sion, “On general minimax theorems,” *Pacific Journal of Mathematics*, vol. 8, no. 1, pp. 171–176, March 1958.

[37] I. Ginchev and V. I. Ivanov, “Second-order characterizations of convex and pseudoconvex functions,” *Journal of Applied Analysis*, vol. 9, no. 2, pp. 261–273, June 2010.