A Distributed ML Framework for Service Deployment in the 5G-based Automotive Vertical

Vasilis Sourlas*, Amr Rizk†, Konstantinos V. Katsaros*, Panagiotis Pantazopoulos*, Georgios Drainakis*, and Angelos Amditis*

*Institute of Communication and Computer Systems (ICCS-NTUA), Athens, Greece.
†University of Duisburg-Essen, Germany.
Email: v.sourlas@iccs.gr, amr.rizk@uni-due.de, {k.katsaros, ppantaz, giorgos.drainakis, a.amditis}@iccs.gr

***Abstract*—5G is the convergence technology for the new generation of mobile networks, expected to be massively deployed in the coming years. Building on network slicing and edge computing capabilities, 5G promises to address the diverse and quite demanding performance requirements of a wide range of use cases (UCs). As a result of these technological transformations, vertical industries will have enhanced technical capacity available to trigger the development of new products and services. Driven by these advances, Machine learning (ML) applications are headed towards collaborative distributed (CDL) schemes, to exploit the abundance of clients' data. Contrary to the traditional cloud-based centralized solutions (CML), in CDL schemes, computational load is shifted to the intelligent edge and extends further beyond, to the user-equipment (including connected vehicles). Here, we present a distributed ML (DML) framework, that will provide functionalities for simplified management and orchestration of collections of ML service components and will allow ML-based applications to penetrate the Automotive world.**

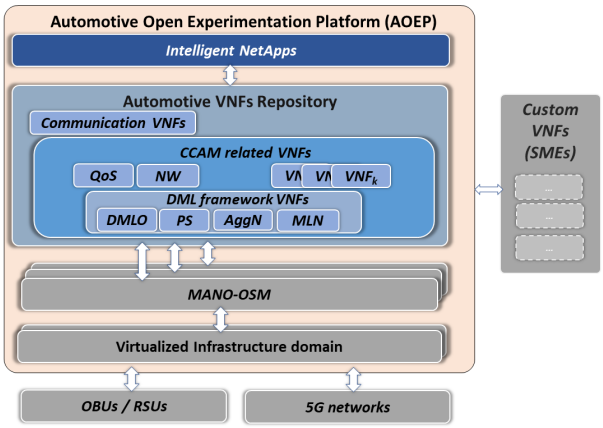

Fig. 1. The H2020 5G-IANA experimentation platform

I. Introduction

5G-based Automotive-related services (*i.e.*, Connected and Automated Mobility services) constitute a broad range of digital services in and around vehicles including both safety-related and other commercial services provided, enabled, or supported by 5G networks. The imminent rollout of 5G is expected to become a "game changer". For the first time, mobile networks will offer a broad range of connectivity features including gigabit speeds and mission critical reliability needed for the Automotive vertical. Most importantly, the prospect that 5G will be a unified multi-service platform, serving not only the traditional mobile broadband market but also enabling digital transformation in a number of vertical industries, is expected to result in the creation of unprecedented opportunities for innovation and economic growth. It is forecasted that more than 125 million passenger vehicles produced in the next four years will be equipped with embedded connectivity, out of a total of 1.2 billion motor vehicles in use worldwide. It is therefore expected that the economic and societal impact of connected mobility will be significant, and that mobile communication systems such as 5G will play a central role in the future transport ecosystem. Indeed, the majority of the 5G-based Automotive-related services will require a highly reliable and safe guidance infrastructure, which will have to combine all available technologies: sensors (in vehicles and on the ground), high accuracy localization, precise positioning, high definition mapping, converged AI on devices, at the mobile network edge and in the cloud, and, in particular high quality (*i.e.*, D2D) direct and network communications between all moving and fixed elements (vehicles, bikes, pedestrians and road infrastructure). Functional redundancy and complementarity in the architecture will be necessary to be able to meet the demanding KPIs of such services (*e.g.*, full automation, remote driving, *etc.*).

In view of these challenges, the H2020 5G-IANA project (https://www.5g-iana.eu/) aims at providing an open and enhanced experimentation platform (see Fig. 1) that will provide access to 5G network resources, on top of which third party experimenters (*i.e.*, SMEs) in the Automotive-related 5G-PPP vertical will have the opportunity to develop, deploy and test their services. In the context of 5G-IANA, an Automotive Open Experimental Platform (AOEP) will be specified, as the whole set of hardware and software resources that provides the compute and communication/transport infrastructure as well as the management and orchestration components tailored to the Automotive sector. The 5G-IANA platform will expose to experimenters secured and standardized APIs for facilitating all the different steps towards the production stage of a new service.

5G-IANA targets different virtualization technologies integrating different management and orchestration (MANO) frameworks for enabling the deployment of the end-to-end network services across different domains (vehicles, road infrastructure, MEC nodes and cloud resources). 5G-IANA proposes a new Automotive VNFs Repository including an extended list of ready to use open accessible Automotive-related VNFs and NetApp templates, that will form a repository for SMEs to

use and develop new applications. Additionally, 5G-IANA will develop a distributed ML (DML) framework (*i.e.*, composed of a set of ready to use VNFs), that will provide functionalities for simplified management and orchestration of collections of ML service components. This DML framework, will allow ML-based applications to penetrate the Automotive world, due to its inherent privacy preserving nature.

In this paper, we provide a description of the 5G-IANA DML framework as well as the Use Case where it will be facilitated to perform network status monitoring in Section II and Section III accordingly. In Section IV an indicative preliminary experiment of the framework's capabilities is presented, whereas Section V concludes the paper.

II. Distributed ML Framework for the Automotive Vertical and Beyond

A. State of the art

The research and innovation areas of Artificial Intelligence (AI) and Machine Learning (ML) have recently witnessed wide attention due to significant advantages fuelled by the availability of high volumes of training data, as well as progress made on the hardware front *e.g.*, GPUs. Typically, data required for ML training is by nature associated with myriad distributed end-user devices, which are usually restricted by their battery life and network connectivity. On the other hand, computational power is inherently centralized, residing for instance in high-performance cloud servers.

In view of this gap, traditional approaches employ centralized machine learning (CML), where clients transmit their acquired raw data to a central server that is responsible thereafter to perform the computationally-heavy model training task. Driven by the network's evolution towards transferring intelligence and processing power to the edge and beyond [1], reaching the user equipment (UE), collaborative-distributed learning (CDL) schemes have emerged as an alternative to CML. Relevant examples include split [2] or peer-to-peer (server-less) learning [3], whereby computational tasks are offloaded to the edge or to the clients themselves *i.e.*, *the far-edge*. That trend is also evident on an application level for far-edge analytics [4].

Along these lines and motivated by privacy concerns, Federated Machine Learning (FML), a Google driven distributed learning scheme [5] has recently emerged. FML allows a ML model to be (a)synchronously dispatched to distributed data source locations so as to be locally trained. The resulting updated ML models are subsequently aggregated (averaged) at a central location *i.e.*, the trained model parameters are transferred instead of the data, delivering a new updated global model ready for subsequent dispatching cycles. As a result, potential sensitive data is not exposed to the entity that maintains the global ML model. In certain cases, this further yields network resource savings, where training data transfer volume exceeds that of the ML model *e.g.*, video stream data. Finally, middle-ground approaches, such as hybrid learning [6] which seek to strike a balance between the benefits of CML and FML techniques, have also been added to the ML scheme solution space.

Focusing on the inherently distributed nature of CDL and FML in particular, as well as the dynamic nature of potential training data sources in mobile networks *i.e.*, vehicles in the Automotive world, a particular research challenge relates to the selection of the client/node/UE/vehicle population participating a training/averaging cycle and the corresponding interplay between global model accuracy, convergence speed and resource utilization. In a work focused on vehicle-captured image classification, client selection is based on the estimated local data quality *i.e.*, image blur as a function of vehicle velocity, and the availability of computation resources [7]. Elsewhere, client selection is driven by limitations in computation and wireless network resources, and corresponding estimations on response time, seeking to maximize the training efficiency [8]. Further challenges relate to the tradeoff of model consistency and training performance.

In our work, we argue that in order to facilitate any ML scheme, all the participating actors, namely the cloud, the core network, the intelligent edge and the involved clients (*i.e.*, vehicles and road equipment) are to be considered; therefore, an end-to-end resource analysis is required and a versatile framework to enable the most appropriate scheme.

B. The proposed DML framework

The 5G-IANA DML framework identifies and targets two particular limitations in the current state-of-the-art, both related to the practical application of the CDL concept in future mobile networks. The first relates to the currently oversimplification of the system model, with the vast majority of related work adopting one or more of the following unrealistic assumptions: i) homogeneous network conditions *e.g.*, bandwidth availability [9], ii) stable network conditions throughout a training/aggregation cycle, iii) node availability [10], iv) homogeneous data availability (volume) per client/node, v) homogeneous data quality/importance that may be challenged in the sense that non-iid data have been frequently studied while they can conditionally be seen as of low quality/importance, vi) fixed distributed (federated) learning configuration *e.g.*, fixed training schedule. The second limitation relates to the lack of a system design solution that will enable the intelligent orchestration of distributed learning in mobile network environments. Such solutions require the operation of key monitoring information regarding the overall resource and data availability in a distributed and inherently non-uniform environment. Hence, the applicable solution is not merely the concatenation/combination of individual heuristic/algorithmic solutions encountered in literature, but has to go through a rigorous assessment/verification of underlying assumptions.

The corresponding system level requirements point out an interplay between resource level conditions *e.g.*, bandwidth availability, intermittent connectivity, data storage, *etc.*, and application level conditions *e.g.*, data volume and quality. As a result, we argue, that a holistic distributed machine learning orchestration framework is required to establish the following functionalities, in tight integration with the overall network MANO architecture (designed in 5G-IANA for the Automotive vertical): i) the monitoring mechanisms at discrete resource

and data/application levels, ii) the model and/or data transfer mechanisms, taking into account device heterogeneity, iii) the decision making mechanisms that based on the input information, will have the role to orchestrate the distributed learning process, providing adaptations to the varying conditions of the considered environment.

The proposed DML framework provides multiple functionalities to optimize parameter and gradient compression, *e.g.*, through quantization and sparsification. The framework also provides functionalities for model consolidation that span different parameter consistencies, with different rates of parameter updates and synchronization techniques that start with basic elastic model averaging up to *e.g.*, synchronous and asynchronous Stochastic Gradient descent. For adaptive configurations, *i.e.*, when the distributed architecture is allowed to be changed by the DML orchestrator, an architecture search will be conducted based on online optimization techniques such as reinforcement learning and Sequential Model-Based Optimization. The framework implements a novel DML representation that provides functionalities such as ML topology selection and various performance and privacy configurations along the spectrum of ML model/parameter consistency and data distribution, respectively. For example, configurations include synchronization options for decentralized training as well as placement restrictions for the ML nodes. These functionalities are mapped into chained VNFs that provide different ML capabilities to other VNFs. ML VNFs can be categorized into different types such as Model Nodes, Aggregation Nodes, Parameter Server Nodes and Orchestrator Nodes (see Section III).

A ML related VNF can be deployed on different virtualized environments such as OBUs (On Board vehicle Units), RSUs (Road Side Units) or MEC (Multi-access Edge Computing) nodes in the corresponding Automotive vertical. In particular, to facilitate the composition of distributed ML topologies, new ML service requests will provide a specification of the required ML functionality as well as the grade of distribution, *e.g.*, choosing model and/or data parallelism or hybrid parallelism and pipelining. Further, when a service request is annotated with the distribution grade, it also may choose a corresponding topology for the distributed ML service, *i.e.*, tree-like, or geographically distributed P2P-like topologies connecting the ML nodes. The framework calculates a default number of required ML nodes and their placement in the network (or network slice), *e.g.*, on the OBUs, RSUs or the MEC nodes. The placement of the ML nodes is carried out by the ML Orchestrator which provides for simplicity first a basic ML topology that chains the ML VNFs in a predefined topology if a standard service is requested. The orchestrator also calculates and deploys a placement of ML VNFs for customized service requests that adhere to certain privacy constraints such as ML VNFs on OBUs that only communicate certain parts of the results to the aggregation node. This allows for example for shared model parameters between different chained services. The framework allows the ML service to request adaptive distributed ML configurations such that the ML Orchestrator may change the configuration of the DML service, *e.g.*, the number of ML nodes, to increase the accuracy of the model. Note that this requires feedback from the initialized service.

The DML framework will be showcased in the course of the H2020 5G-IANA project through a distributed network monitoring service. The Network monitoring service provides an overview of the status of network components such as OBUs and draws conclusions and predictions with respect to the performance of the monitored components. More details on the network monitoring service can be found in Section III.

III. Network Monitoring Service for the Automotive Vertical

As 5G networks promise a broad range of network slicing possibilities (including different V2X slices for different services such as autonomous driving of various vendors) a network monitoring service is required not only to obtain resource status and statistics but also within a slice to enable tying the (spatial) network connectivity status to the expected performance of distributed applications. This allows adaptive services of the automotive vertical to make fine-grained decisions based on the network spatiotemporal state. This could also support vehicles to switch if necessary from the home network (or home slice) into a different network (or slice) without sacrificing service quality. However, to obtain accurate spatiotemporal network information a large amount of network measurement data needs to be continuously collected and analyzed.

The Network Monitoring Service (NMS) has the goal to minimize the data collection effort by utilizing a distributed Machine Learning approach, *i.e.*, instead of collecting large amounts of network monitoring data to be centrally analysed, the ML analysis/prediction model is distributed on the VNFs located at the RSUs and the vehicles *i.e.*, OBUs. To this end, the NMS utilizes V2X communications (*i.e.*, through the corresponding VNFs, see Fig. 1) to deliver predictions of the network quality to a central computation entity at the cloud or a MEC server. In essence, the goal of the NMS is i) to learn data traffic patterns for data traffic prediction, ii) to learn network condition models to provide QoS predictions, and iii) to learn to distinguish between normal and abnormal network behaviours to detect and predict faults.

The NMS chains the following VNFs from the 5G-IANA VNFs Repository:

- **Distributed ML Orchestrator (DMLO):** The DMLO VNF initializes and operates the DML service. It decides on the DML topology as well as the roles of the involved ML-nodes.
- **DML Parameter Server (PS):** The DML PS VNF provides a parameter server for distributed ML scenarios. It allows each of a set of model replicas to share parameters in synchronous or asynchronous communication.
- **DML Aggregation Node (AggN):** The DML AggN aggregates the local models and broadcasts the averaged result back to the local ML-node VNFs. The DML AggN can also work as Ensemble Aggregator.

- **ML node -Training Agent (MLN):** The MLN runs the local ML model (training agent) on local data and connects to one or many of the following VNFs: PS for synchronized model parameters, DMLO for initialization and operation.
- **Network (NW) monitoring:** The NW VNF monitors the wireless network conditions as seen from the OBU or RSU against relevant KPIs for the NetApps and keeps a history of the NW conditions.
- **QoS prediction:** The QoS VNF uses the NW monitoring VNF data to fit the NetApps needs for wireless QoS with the actual network conditions.

Note that the NW and QoS VNFs are broader to the ML framework and are essential for the NMS service. Also, other VNFs are chained for the implementation of the Network Monitoring Service *e.g.*, for the communication between the devices of the network but are not presented here since they are not part of the service itself. The proposed 5G-IANA DML framework presented in Section II consists by the DMLO, the PS, the AggN and the MLN VNFs (see Fig. 1).

The Network Monitoring Service will be initiated at the DML Orchestrator that will chain multiple ML nodes as VNFs on a subset of the available OBUs and/or RSUs. These VNFs (NW monitoring) will collect network state information such as response times and packet data rates to update network performance models, as well as, models of normal network behaviour. The ML nodes will train local models that will be combined for consistency using two methods i) synchronous and ii) asynchronous parameter server (PS) VNF that is located at RSU or MEC or that is hierarchically built using multiple aggregation node (AggN) VNFs at RSUs with root PS at MEC. Finally, a network monitoring orchestrator VNF (NMO) decides on the hierarchy of the network monitoring service, *i.e.*, the grade of the distribution of the ML training based on the OBUs/RSUs network traffic logs.

In the course of 5G-IANA project we will also show a centralized Ensemble Learning case where the Aggregation VNF at the MEC server or the cloud collects data logs from the OBUs or RSUs first to perform model stacking, *i.e.*, training multiple classifiers on the dataset. We will compare the computing load on the MEC server for both cases. Traffic and QoS predictions will be shown to be deduced from the local model replicas as well as from the centralized model. There we will explore the prediction latency *vs.* accuracy trade-off in comparing the distributed case *vs.* a centralized model at the edge or the cloud.

We will show that distributed and predictive Network Monitoring supports 5G based applications to make efficient use of their data and resources. The result of this service will support the deployment of new types of 5G mobile services such as autonomous driving. For example, the NMS will additionally show how network monitoring information can be provided with and without guarantees to a MEC-side application, *e.g.*, predicting vehicle trajectories. In case of guarantees on the delay of network monitoring results, through using a URLLC slice, the MEC-side application is expected to have a higher prediction accuracy. This will demonstrate the potentials of Distributed ML schemes in 5G-PPP verticals like the Automotive one where the network is volatile and privacy concerns is of utmost importance.

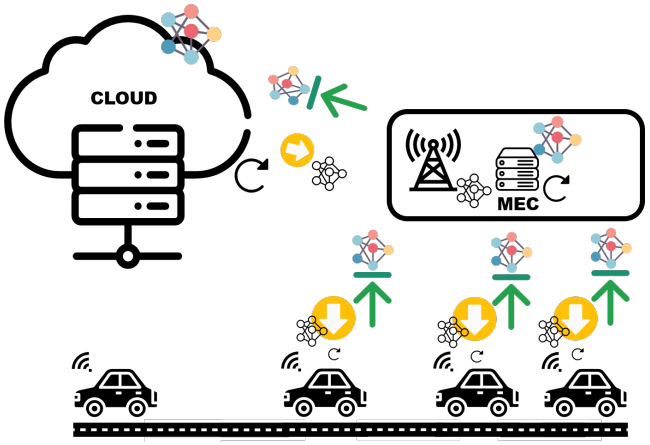

Fig. 2. A representative DML illustration, where the ML task can take place at the user or the MEC level

IV. Distributed ML Framework Initial Experimentation

In order to demonstrate the capabilities and the versatility of the DML framework, regardless of its use in the Network Monitoring Service, we experimentally evaluate a Machine Learning (ML) task, when performed in a data centre/cloud (Centralized Learning - CML) or offloaded (enabling the DML framework) to the edge of the network (EML) or at the mobile devices *i.e.*, OBUs and RSUs (Federated Machine Learning - FML). The EML approach functions as a middle-ground solution between CML and FML. In each learning round, each edge node receives the central model from the central server and the data from the clients in its service area. Thereafter, the models are trained and the respective model parameters are sent to the central server, which in turn performs the aggregation (similarly to the FML case).

A. ML task

We have selected an image-classification problem, as a representative ML task of automotive relevance *e.g.*, licence plates/traffic signs recognition scenarios, and in specific digit recognition from given images taken from the Street View House Numbers (SVHN) dataset (http://ufldl.stanford.edu/housenumbers/). SVHN, is based on a set of real-world images, with digits taken from natural scenes (house numbers in Google Street View). It contains a training dataset of 531K 32x32 colour training images (of 1.3 GB size) split in 10 classes (for digits 0-9) and a test dataset of 26K test images. The original SVHN training dataset is replicated 10 times, adding a random Gaussian noise factor (blurring) to the images vectors, essentially resulting in a total synthetic 13GB dataset.

A neural network has been developed to address the above-mentioned task, comprised of an input layer of 3072 neurons, which correspond to the total pixels of the input SVHN images (32x32x3), an output layer of 10 neurons, equal to

the total output classes of SVHN and a hidden layer of 512 neurons. Rectified Linear Unit (ReLU) activation is applied on the hidden linear layer (ReLU functions as a filter, allowing only positive values to pass through), while on the output layer LogSoftmax activation [11] is selected, being more effective for N-element classification tasks. Regarding hyper-parameter settings, a batch size of 64 samples was chosen, along with a learning rate of 0.1, based on the default settings for similar image classification tasks in PySyft library [12]. The total model size reaches 6.1MB. Federated Averaging (FedAvg [13]) algorithm is used in all distributed learning cases in our PySyft implementation [12].

B. Network model

As 5G networks are currently under deployment, we assume a mobile Long-Term Evolution (LTE) cellular network comprised of several mobile clients *i.e.*, vehicles equipped with LTE-enabled RSUs, each holding an amount of training data. In each cell, a wireless link connects the clients with the base station unit (BS). Also, BSs are able to communicate with the edge nodes and with a central data centre (DC) cloud server, via the intermediate core (wired) network (Fig. 2). The system's aim is to perform a ML task, utilizing available client data through the three mentioned schemes (*i.e.*, CML, FML and EML). In the EML we assume that within the intermediate core (wired) network exist a few edge node MEC servers. Real-world traces were selected to capture our network's dynamics. In specific, the Shanghai Telecom Dataset [14] was used, which contains records (LTE traces) of UEs accessing the Internet through a BS in a period of 15 days (the corresponding network assumes 1853 BSs).

The UEs throughput in both the uplink (UL) and the downlink (DL) (in MBytes/sec) is modelled as a Gaussian random variable. Its mean value is assumed equal to the average cell throughput, which is 5.9 (UL)/7.73 (DL) MBytes/sec for 2.5 GHz LTE according to [15], divided by the number of online clients. Here for simplicity we assume that all BSs are equally loaded. In an actual network, the identification of the MEC server locations would ideally be the outcome of a facility location problem, accounting for various factors *i.e.*, spatial characteristics, cellular architecture, average user demand, *etc.* As our baseline approach, a uniform distribution of cells across the MEC nodes is considered. Although non-realistic, it is insightful as it reduces the involved problem parameters. A more skewed distribution is left for future work. For our case we assume 5 MEC nodes within the network.

C. Training process and entities computational characteristics

The training dataset, including data and labels, is firstly shuffled and then uniformly divided into Z partitions. Then, a fixed number of per round participating clients K_N is chosen ($K_N < Z$) and each one is assigned a dataset partition. In the CML case, the selected clients upload their local datasets to the central server (running DMLO, PS, AggN, MLN VNF instances) in a parallel manner. The central server, thereafter merges the datasets into a single super-dataset and performs the ML training task, marking the end of the round. Same settings overall apply to the FML case. Here, the central server first shares the training model to the clients (DMLO, PS and AggN VNFs). Then, each client (MLN VNF) trains the model using only its available (local) data and finally uploads the updated model back to the server, again in a parallel manner. The server is the one to perform aggregation of all the collected models. Finally, in EML each participating client uploads its data to its serving MEC node (MLN VNF); the latter has already received the training model from the central server and then the training process occurs per edge node. Subsequently, updated models are sent back to the central server, similar to the FML case.

The computational capacity of a UE (in Automotive world a vehicle OBU or an RSU) to perform a ML task, measured in (processed) training samples/sec depends on the dataset content *e.g.*, images pose different requirements than natural language, the UE's capabilities and the training model's complexity. A good approximation for popular large-scale classification tasks can however be deducted from [16]. For our case, we use a reference (average) value of 125 training samples/sec. For the MEC nodes/servers we have considered as our main reference the latest commercial solutions specified by Amazon's AWS Wavelength services [17]. It offers cloud services specialized for ML (Amazon EC2 P3 instances) and is equipped with an NVIDIA Tesla V100 Graphics Processing Unit (GPU). GPU computational capacity values for ML can be found in [18], where we select an average value of 6000 training samples/sec. Finally, the computational tasks for the cloud server include training (in the CML case) and model parameter aggregation (FML, EML cases). For the former we select an average value of 40,000 training samples/sec, assuming a Data Center is equipped with a Tensor Processing Unit (TPU). For the latter (model parameter aggregation), no reference values can be found in the literature, thus we rely on an empirical approach; we measure the average capacity for training and aggregation tasks in our personal computer (PC) setup (*i.e.*, 6250 training samples/sec and 1.56 model aggregations/sec respectively) and compare against the training capacity reference value of 40,000 training samples/sec that was selected, according to [18]. Assuming a linear relation, the average cloud aggregation capacity is calculated as 10 model aggregations/sec.

D. Simulation environment and results

The simulation environment was set-up in a single desktop machine with the following characteristics: Intel Core i7-10700 CPU @ 2.9 GHz, 64-bit, RAM 16 GB, OS Windows 10. We have chosen the following scenario, based on our previous work [19]; $Z = 500$, accounting for a per client data to model ratio $r \approx 4$. Per round participants (K_N) is fixed to 50. To evaluate the performance of each ML scheme the following metrics are considered: 1) Testing Accuracy, representing the ratio (%) of successful to total classifications. 2) Traffic Volume overhead, calculated as the sum of total data exchanged between the central server and the clients (FML, CML) or between the central server, the MEC nodes and the clients (EML). For the sake of clear representation, traffic

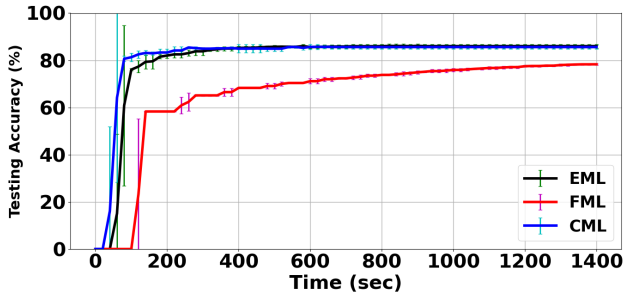

Fig. 3. Testing accuracy w.r.t time

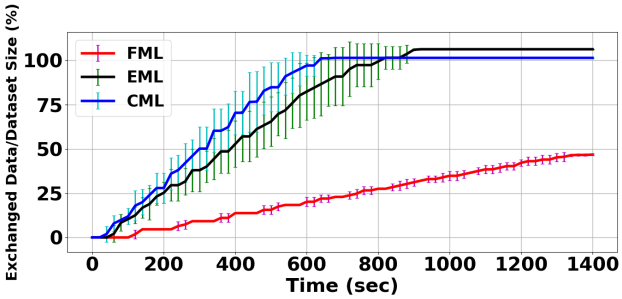

Fig. 4. Traffic overhead w.r.t time

volume overhead is normalized to the total training dataset size (*i.e.*, 13 GB).

In terms of the resulted accuracy *i.e.*, the accuracy achieved at the end of training, EML exhibits similar performance to CML, both outperforming that of FML by an average value of 8% (Fig. 3). This observation matches the intuitive expectation of EML approaching the behavior of CML, since in both cases large amounts of data are collected and trained centrally. A similar observation can be made in regards to the resulted bandwidth expenditure (Fig. 4), where both EML and CML demonstrate similar values of traffic overhead. In the CML case, the value of total data exchanged approximates the 100% of the training dataset, which stems from the fact that all client data is uploaded centrally. EML exceeds slightly 100% (by an average amount of 5%) since apart from all the available data uploaded to the edge servers, the (lightweight) model parameters and aggregated result are exchanged with the central server. Note though, that this traffic is accumulated towards the edge of the network, where MEC nodes reside and has less impact on the network performance and resource allocation (we leave this for future investigation). FML, on the other hand, appears as more bandwidth-efficient by (at least) an amount of 50%, compared to CML-EML. Such behavior is expected, since only models are exchanged between the central server and the clients, whose size is 4 times less, compared to the (actual) per client data (r=4). Overall, a trade-off is emerging; a direction towards centralized methods provides better results in terms of accuracy, applying however higher (twice as much in our case) communication costs. From the viewpoint of convergence speed, CML and EML are able to reach their peak accuracy during the very first training rounds (in less than 200 secs). At this stage, they outperform FML by 23% (Fig. 3). The slow accuracy improvement of FML compared to other schemes is mainly dictated by the low client processing capacity, as opposed to an edge node (GPU) or the cloud server (TPU). The faster convergence of CML-EML comes at a cost in bandwidth expenditure (Fig. 4). In specific, CML and EML at 200 secs require 5 times more data as compared to FML.

V. Conclusions

In this work we introduced a Distributed ML framework that will provide functionalities for simplified management and orchestration of collections of ML service components to the Automotive 5G-PPP vertical. Besides the functionality of the DML components we also presented the Network Monitoring Service where the DML will be mainly used as well as an initial experiment that showcases the framework's potentials in distributing ML tasks across a network.

Acknowledgements

This paper is part of the 5G-IANA project, co-funded by the EU under the H2020 Research and Innovation Programme (grant agreement No 101016427).

References

[1] W. Shi *et al.*, "Edge computing: Vision and challenges,"IEEE Internet of Things Journal,vol. 3, no. 5, 2016.
[2] M. G. Poirot *et al.*, "Split learning for collaborative deep learning in healthcare,"arXiv preprint arXiv:1912.12115, 2019.
[3] S. Savazzi *et al.*, "Federated learning with cooperating devices: A consensus approach for massive IoT networks,"IEEE Internet of Things Journal, vol. 7, no. 5, 2020.
[4] K. A. Alam *et al.*, "Enabling far-edge analytics: performance profiling of frequent pattern mining algorithms,"IEEE Access, vol. 5, 2017.
[5] B. McMahan and D. Ramage, "Google AI blog: Federated learning: Collaborative machine learning without centralized training data," https://ai.googleblog.com/2017/04/federated-learning-collaborative.html, April 2017, (Accessed on 10/27/2020)
[6] A. M. Elbir, "Hybrid federated and centralized learning,"arXiv preprint arXiv:2011.06892, 2020
[7] D. Ye *et al.*, "Federated Learning in Vehicular Edge Computing: A Selective Model Aggregation Approach," in IEEE Access, 2020.
[8] T. Nishio *et al.*, "Client selection for federated learning with heterogeneous resources in mobile edge," IEEE ICC, 2019.
[9] L. Liu *et al.*, "Edge-assisted hierarchical federated learning with non-iid data," arXiv preprint arXiv:1905.06641.
[10] G. Zhu *et al.*, "Low-latency broadband analog aggregation for federated edge learning," arXiv preprint arXiv:1812.11494.
[11] PyTorch Neural Network API. [Online]. Available: https://pytorch.org/docs/stable/nn.html
[12] T. Ryffel *et al.*, "A generic framework for privacy preserving deep learning," arXiv preprint arXiv:1811.04017, 2018.
[13] B. McMahan *et al.*, "Communication-efficient learning of deep networks from decentralized data," in Artificial Intelligence and Statistics, 2017.
[14] S. Wang *et al.*, "Edge server placement in mobile edge computing," Journal of Parallel and Distributed Computing, vol. 127, 2019.
[15] M. R. Akdeniz *et al.*, "Millimeter wave channel modeling and cellular capacity evaluation," IEEE Journal on Selected Areas in Communications, vol. 32, no. 6, 2014.
[16] J. Liu *et al.*, "Performance analysis and characterization of training deep learning models on mobile device," in 25th IEEE Intern'l Conf. on Parallel and Distributed Systems, 2019.
[17] Amazon AWS Wavelength - EC2 instance types. [Online]. Available: https://aws.amazon.com/ec2/instance-types/
[18] Y. Kochura *et al.*, "Batch size influence on performance of graphic and tensor processing units during training and inference phases," in International Conference on Computer Science, Engineering and Education Applications. Springer, 2019.
[19] G. Drainakis *et al.*, "Federated vs. centralized machine learning under privacy-elastic users: A comparative analysis," in IEEE 19th Intern'l Symposium on Network Computing and Applications, 2020.