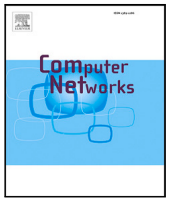
Computer Networks 242 (2024) 110239

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



A federated learning approach to QoS forecasting in cellular vehicular communications: Approaches and empirical evidence

Nehal Baganal-Krishna [a,b], Ralf Lübben [c], Eirini Liotou [d], Konstantinos V. Katsaros [d], Amr Rizk [b,*]

[a] University of Ulm, Germany
[b] University of Duisburg–Essen, Germany
[c] Flensburg University of Applied Sciences, Germany
[d] Institute of Communications and Computer Systems (ICCS), Greece

ARTICLE INFO

Keywords:
Predictive QoS
Vehicular communications
Federated learning
5G measurements

ABSTRACT

QoS forecasting for cellular vehicular communications allows cooperative, connected and automated mobility applications to tailor their behavior to the expected communication conditions on the road. In a nutshell, vehicles may, for example, execute cooperative maneuvers if the communication quality of service is only above a certain quantitative level whereas if not they revert to the individual autonomous mode. In this paper, we propose and show empirical methods for estimating packet-based QoS metrics obtained from 5G network measurements with a direct application to vehicular applications. As many distributed vehicular applications possess strict QoS requirements, we focus here on bounding packet-based statistical QoS quantiles, specifically for latency and loss. Our approach is based on training regression neural networks in a federated learning fashion and show that it can obtain predictions on par with centralized training without the vehicles needing to transmit raw measurement data. In contrast to QoS prediction using physical layer information, we briefly discuss the embedding of such much simpler application-level service within the 5G architecture. We also validate our approach through recovering classical closed-form delay quantiles that are obtained from analytical models of simple queueing systems. We show that our approach goes beyond these simple models in that it provides quantile estimates for the complex scenario of cellular vehicle communications and under different application traffic patterns including empirical data traffic traces as well as 5G testbed measurements.

1. Introduction

In Cooperative Adaptive Cruise Control (CACC) and similar distributed applications within the automotive domain, strict Quality of Service (QoS) constraints on delay, throughput, and loss pose significant challenges for the cellular network in addition to imposing high expectations on the ability to predict network conditions. For example, CACC employs Cooperative Perception Messages (CPM) to share real-time information on the vehicle's surroundings, and Maneuver Coordination Messages (MCM) to enable vehicles to coordinate trajectories. A critical prerequisite to the success of this coordination is the reliability of the communication network. Therefore, before initiating any cooperative maneuvering routine the vehicles should be capable to estimate the network QoS offered to transmit all CPMs and MCMs during the routine.

In this paper, we consider the problem of providing vehicles with timely QoS metric predictions based on a federated learning approach. In particular, we are interested in predicting tail quantiles of QoS conditions, not averages, due to the strict nature of the application constraints. We assume that raw measurement results are not transferred from the vehicles to the edge for central processing but rather the model parameters for federated learning iterations. To enable a robust QoS forecasting method, we evaluate the applicability of quantile regression neural networks (QRNN) [1,2]. Here, we develop QRNN to achieve reliable estimates for packet delays or throughput for forecasting the quantiles as $P[\text{delay} > x] \leq \epsilon$ and $P[\text{throughput} < x] \leq \epsilon$, whereas ϵ is typically small, but depends on the application.

Our approach is related to adaptation schemes that derive the transmission rate from system models such as the one used in TCP BBR and congestion control algorithms for real-time communication [3–5]. These methods empirically estimate through probing the QoS operating point for the appropriate amount of data to transmit, however, on a round-trip-time (RTT) based time scale. In contrast given the Cooperative, connected and automated mobility (CCAM) scenario at inference time, we directly perform regression on the desired QoS

* Corresponding author.
E-mail address: rizk@ieee.org (A. Rizk).

https://doi.org/10.1016/j.comnet.2024.110239
Received 26 December 2023; Accepted 5 February 2024
Available online 14 February 2024

parameter locally in the vehicle on a much smaller time scale. Our approach is also related but goes beyond methods that use asymptotic models that describe the steady state of a network link of a path. For example, analytical models describe the steady state behavior of link or path metrics such as the sojourn and waiting times. Similar models can be found in [6–9] to describe the available bandwidth. We use such analytical models to verify that the QRNN data-driven approach delivers congruent results, however, we go beyond such models to obtain tail forecast values that are hard to obtain analytically.

In this work, we are bridging time-series prediction with the description of the forecast as a probabilistic bound. Essentially, we estimate an upper bound for the probability that the delay of the $n + k$th packet overshoots a delay quantile $W^{\varepsilon}_{RTT}(n + k)$ conditioned on prior observed packet delays. Similarly, we estimate an upper bound for the probability that the throughput associated with the $n+k$th packet falls below a throughput quantile $W^{\varepsilon}_{TP}(n+k)$ conditioned on prior observed packet throughput. The use of QRNN for the prediction allows for a non-parametric modeling, which drops assumptions on the statistical distribution of packet arrivals, the wireless link service process as well as on the error terms in a classical time series model. The model further allows the evaluation of input features beyond the past metrics, e.g., given a prediction of future packet transmission patterns by the application or path information to improve the QoS prediction.

Both centralized and distributed machine learning approaches are employed in this work to train the QRNN to predict network QoS metrics. Our evaluation relies, first, on synthetic latency measurements and MLAB [10] traffic traces to benchmark against well-known analytical performance bounds, before additionally, applying our approach to training and prediction given real-world traffic collected through CCAM vehicles at a 5G standalone test network spanning a south German city.

In contrast to an earlier version of this work [11], this article extends on tail forecasting for Predictive QoS in cellular vehicular communications. Additionally, we collect real-world traffic using connected vehicles and train the QRNN in a federated fashion to enable PQoS in this CCAM scenario.

The outline of the paper is as follows: Section 2 focuses on elucidating the support for predictive QoS within the 5G architecture while Section 3 delves into the problem statement and our strategy for estimating QoS metric quantiles. We explore two machine learning approaches used to train the neural network for quantile prediction, supplemented by analytical connections to analytical but simple system models. Section 4 presents the delay quantile predictions generated by centrally trained neural networks, analyzing scenarios involving synthetic data and MLAB traces for more complex systems. Section 5 encapsulates the results from federated machine learning (FML) training, encompassing predictions for both delay and throughput quantiles in a 5G network setting. Section 6 reviews the related work. Section 7 concludes this paper with a discussion of strengths and limitations.

2. Predictive QoS for distributed automotive applications

In this section, we discuss how a 5G architecture can seamlessly integrate Predictive QoS (PQoS) for scalable service deployment. 5G represents a paradigm shift beyond conventional networks, offering not only enhanced bandwidth and reduced latency, but also a cohesive framework for service deployment. The extensive network of connected nodes, such as on-board units (OBUs) and roadside units (RSUs) within a 5G environment, paves the way for distributed machine learning (federated machine learning, as applied in our context). This leverages cooperative data collection, resulting in elevated statistical analysis. Moreover, within the 5G architecture, components like the Policy Control Function provide avenues for services like PQoS to manage their dedicated network slices effectively.

Moreover, the 5G Service Based Architecture (SBA) includes an elaborate framework for the support of network analytics services, mainly through the operation of the Network Data Analytics Function (NWDAF) [12]. NWDAF encompasses both ML inference and training functionality, including interfaces for the management of Federated Learning training processes. Through these primitives the 5G SBA supports the provisioning of a series of analytics services, including the prediction on QoS metrics. Such services become available to consuming entities e.g., applications, through the Network Exposure Function (NEF). When integrated with PQoS, the NEF gains the capability to not only expose network services but also to do so in a way that considers and accommodates the specific QoS requirements of these applications.

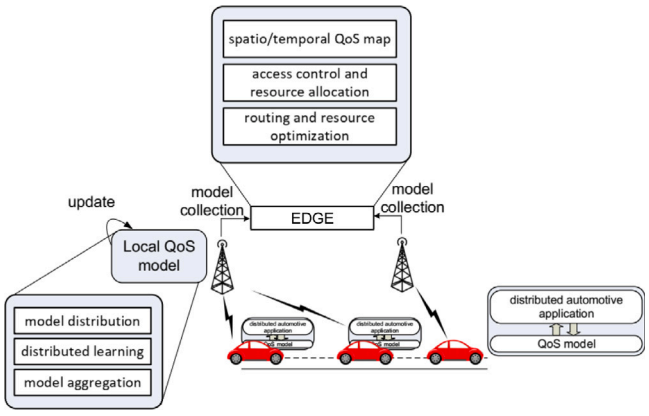


Fig. 1. Predictive QoS for 5G enabled distributed automotive applications.

For instance, the NEF can dynamically expose network resources and functionalities tailored to the predicted needs of a particular application. This means that an application with stringent QoS demands, such as cooperative maneuver coordination that uses MCM messages, can receive priority access to low-latency, high-bandwidth connections, ensuring seamless and reliable performance. Moreover, the NEF can facilitate the dynamic allocation of resources based on real-time conditions and PQoS predictions, further optimizing network performance. Finally, the application on the OBU side can choose based on the PQoS inference whether to initiate cooperative maneuvers that require stringent QoS along a planned trajectory in a road segment or not.

We depict the concept that underpins our approach in Fig. 1. The regression model is disseminated to vehicles, specifically to OBUs, for local training on their respective datasets. Subsequently, the models from each OBU are forwarded to the edge server for aggregation. The edge server leverages this aggregated model, which encapsulates spatio-temporal QoS. Most importantly, vehicles receive prediction models from the edge tailored to forthcoming road sections, providing anticipated network performance. Furthermore, vehicles retrain these models with new training data and relay them back to the edge for subsequent aggregation.

3. Technical problem description and approach

We consider the problem of predicting the QoS provided to a connection by a network service. In principle, this requires a (data-driven) model of the network performance that is based on the analysis of the historical performance of the network to anticipate its future behavior. Here, we opt for a quantile-based approach instead of predicting mean values as we explain below. This choice enables a richer probabilistic model of network status and allows more nuanced insight into the uncertainty associated with network performance, especially, in Cooperative Connected, and Automated Mobility (CCAM) applications. The QoS metrics that we consider here are packet delay, respectively, round-trip time (RTT), and throughput. Specifically, we aim to determine the network conditional upper bounds for RTT and conditional lower bounds for throughput at a specific location.

For the quantile delay predictions, we consider the system depicted in Fig. 2, where observers obtain timestamps at ingress and egress of

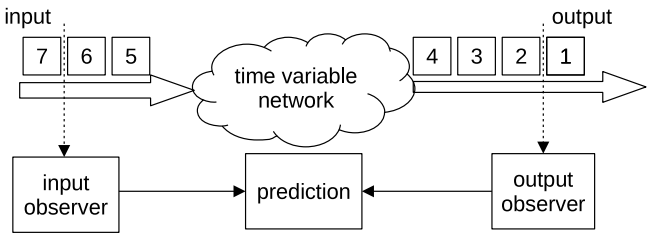


Fig. 2. System model for delay prediction: packet timestamps are observed at the input and output of the system to predict the delay quantile for future arriving packets.

an end-to-end network path or a network segment and hence observe delays for individually transmitted packets. In the CCAM context, the sender and receiver of the numbered data packets are services that are already running on the vehicle on-board unit (OBU) and the edge or in case of RTT estimation they both reside on the vehicle. The observers may then be monitoring services running on the OBU and/or the edge that collect traffic statistics such as timestamps and packet sizes. This setup allows, hence, for active and passive network probing [13,14], where for CCAM services we opt here for the passive variant. In contrast to the active variant that uses synthetic traffic, the passive version obtains predictions based on observations of the CCAM service traffic. The realization of such a system can be implemented, for example, using smart network interface cards that provide time-synchronization and high-speed packet capturing, through in-network telemetry capable switches along the network path or, as we demonstrate in Section 4.2, by round trip time (RTT) measurements over CCAM OBUs. Now, the delay of packet n is described as

$$W(n) = T_D(n) - T_A(n) \tag{1}$$

where $T_D(n), T_A(n)$ are the departure (output) time and the arrival (input) time of this packet from and into the network path, respectively. The task of the observers and the prediction is to take these delay samples and provide a conditional upper bound on the delay quantile $W^{\epsilon}(n+k)$ for the $n+k$th packet of the form

$$P[W(n+k) > W^{\epsilon}(n+k) | W(n), \dots, W(1)] \leq 1 - \epsilon \tag{2}$$

for $k \geq 1$. Note that the estimate sought here is pointwise in the sense that it holds for a predefined future packet with index k.

In the following, we take an empirical approach towards the quantile estimation problem above. In contrast to purely analytical approaches as discussed e.g. in [9,15], we design a supervised machine learning approach for the prediction of the quantiles $W^{\epsilon}(n)$. Here, we train a neural network (NN)[1] to predict the quantile from past observations.

To estimate the packet delay quantiles, we train a quantile regression neuronal network using the pinball loss function

$$L_{Delay}(y, \hat{y}) = \begin{cases} (1-\epsilon)(\hat{y} - y) & \text{if } y < \hat{y} \\ \epsilon(y - \hat{y}) & \text{if } y \geq \hat{y} \end{cases} \tag{3}$$

The form of the loss function clearly resembles the accuracy of the quantile estimates. We minimize the expectation of this loss function with respect to the unknown delay distribution that provides the ϵ-quantile [1,2,16]. By the use of this loss function, the neural network learns to predict the ϵ-quantile of the delay.

We perform a hyperparameter optimization using the parameters listed in Table 2 for a deep feed forward neural network (DNN) architecture and parameters given in Table 3 for a long short term memory (LSTM) neural network. The selection of the hyperparameters is based on the Hyperband search [17]. Since the search algorithm selects

[1] The code is available at https://gitlab.com/ralfluebben/tailing.

Table 1
Input features.

Parameter	Values
Delays of pkt#	[[300 : 400), [399 : 400)]
Interarrival time of pkt#	[{∅}, [400 : 600)]

Table 2
Hyperparameters used for training the DNN.

Parameter	Value/Setting
Number of layers	[1, 2, 3]
Neurons per layer 1	[10, 20, 30, 40, 100, 200]
Neurons per layer 2	[0, 10, 20, 30, 40, 100]
Neurons per layer 3	[0, 10, 20, 30, 40]
Learning rate	[adaptive, 0.01, 0.001, 0.0001]
l2 regularization	[0.01, 0.001]
Drop out	[0.0, 0.5]
Optimizer	adam
Epochs	600 with early stopping
Batch size	2048

Table 3
Hyperparameters used for training the LSTM NN.

Parameter	Value/Setting
Number of cells	[10, 20, 30, 40, 100, 200]
Learning rate	[adaptive, 0.01, 0.001, 0.0001]
Drop out	[0.0, 0.5]
Optimizer	adam
Epochs	200 with early stopping
Batch size	2048

hyperparameters based on the progress of the optimization criteria on a small number of epochs, we perform the optimization for each learning rate individually. The rationale here is that a small learning rate may have a slower progress than a high learning rate but may still perform better for a high number of epochs.

To benchmark the predictions of the trained NN, we use, in addition to the pinball loss function that leads to the prediction of the statistical upper bound, two further metrics. Firstly, the mean absolute error (MAE), in the following denoted as *distance*, which returns the absolute distance (difference) between the predicted bound and the measured delay values as we seek a tight bound in the sense of a small MAE. Secondly, we compare the predicted quantile to empirical quantiles, i.e., the empirical quantile is extracted packet-wise for each packet from all available sample paths of delay or throughput traces. Similarly, we train a neural network model with a variation in pinball loss function (in Section 5.2) for quantile throughput estimation.

Neural networks can be trained in two broad approaches: First, centralized training of neural networks depends on centralized data gathering and processing, which strains the network resources and potentially raises privacy issues. Moreover, centralized machine learning systems might need help to adapt effectively to dynamically changing networks. In contrast, secondly, distributed machine learning (DML) emerged as a powerful solution to these limitations. By leveraging the cooperative and decentralized nature of data collection in CCAM, DML reduces the burden on network resources and alleviates privacy concerns. This approach also enhances scalability, allowing the analysis of large and diverse datasets.

As depicted in Fig. 3 we use Federated Machine Learning (FML), a form of DML, to train the QoS prediction model on each vehicle using its locally collected data. This eliminates the need for vehicles to transmit their data to a central training location, mitigating privacy concerns. The locally trained models are subsequently transmitted to an aggregator function running on the network edge to create a unified global model, which is then disseminated to all vehicles for a new round of training. Note that as depicted in Fig. 3 the local models are trained on local network QoS data, such as delays or throughout measurements. In contrast to the centralized training dataset diversity and balancing

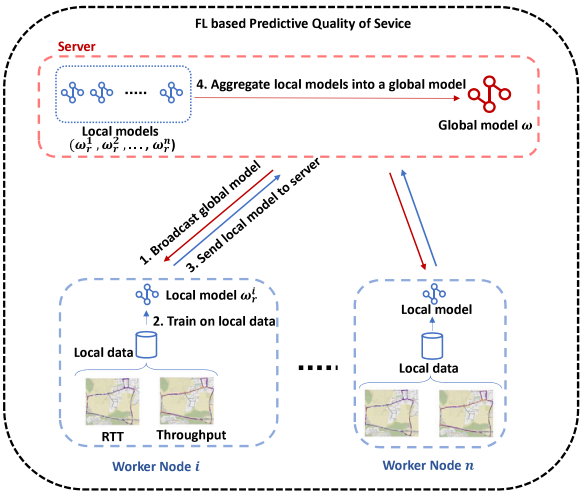


Fig. 3. Federated Learning for predictive QoS in CCAM scenarios. Vehicles, denoted as worker nodes, share locally trained prediction models based on passive, trajectory-specific QoS (RTT, throughput) measurements with a central aggregation service at the 5G network edge. Locally trained spatio-temporal predictive QoS models are aggregated at the network edge and distributed on vehicles for a new round of training. Trained models are supplied by the network edge as a spatial inference service to fresh CCAM-enabled vehicles when entering road sections.

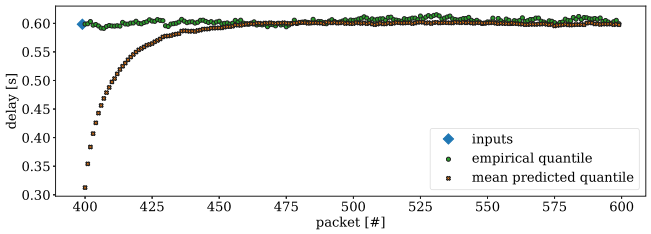


Fig. 4. M/M/1 system: Example of the empirical delay quantile vs. mean predicted quantile for delay input sample of packet #399.

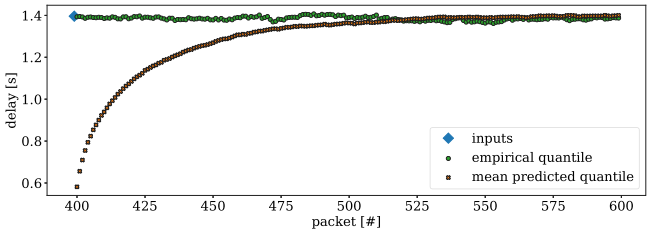


Fig. 5. W/W/1 system: Example of the empirical delay quantile vs. mean predicted quantile. Input features are the delay samples for packets # [399 : 400). Model parameters: Scale σ and shape k as $\sigma = 1.5, k = 0.6647$ (interarrival times) and $\sigma = 0.0375, k = 0.6647$ (service times). The utilization is $\rho = 0.75$ and the violation probability for the quantile estimates is set to $\epsilon = 0.05$.

are crucial as worker nodes, i.e., individual vehicle OBUs, only observe local data (as illustrated through the spatio-temporal RTT and throughput maps). If this observed data is not statistically mixing the aggregate model and the quantile prediction may be rendered useless. To this end, we evaluate in Section 5 the impact of different forms of data segmentation and balancing that are directly related to the spatio-temporal trajectories of the vehicles.

Next, we describe how we train the developed model using both methods (centralized and federated) to compare and emphasize FML for predictive QoS. Specifically, we test centralized training solely to estimate the delay quantiles. Additionally, we differentiate between two input feature sets: the first feature set only includes delay measurements while the second one also includes anticipated future packet arrival times, i.e., the model anticipates the application packet injection patterns. We conjecture that having knowledge or estimates of the network's future packet arrivals enhances the prediction of delay quantiles.

4. Centralized delay predictions: Analytical results and empirical traces

Next, we evaluate the delay quantile prediction approach that is described above on different systems, starting with synthetic queueing systems to empirical network data traces.

4.1. Synthetic queueing systems

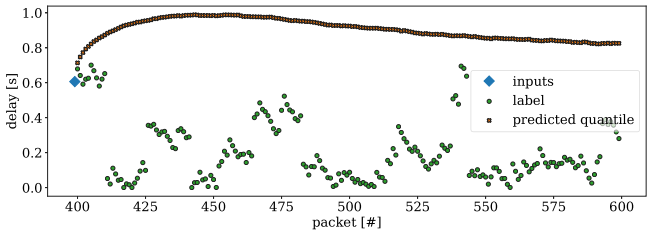
First, we show the predictions of packet delays for synthetic queueing systems with service and interarrival times drawn from analytical distributions before going over to trace-based evaluations. These analytical examples of well understood queueing systems serve well to validate the prediction method and results. We show in the appendix the closed form expression and the prediction results for an M/M/1 system, i.e., one network link, with one server and both exponentially distributed packet inter-arrival times and service times. As depicted in the example in Fig. 4 the empirical prediction result, i.e., packet-wise quantiles W^{ϵ}, quickly converges and recovers the mean predicted quantile for the test data set in addition to recovering the analytical quantile as calculated in the appendix.

Next, we consider a single system with packet inter-arrival times and service times from a Weibull distribution (denoted as W), i.e., we perform experiment for a M/W/1 system and a W/W/1 system. The Weibull distribution leads to a slower than exponential tail of the service or interarrival times leading to an intuitive increase of the delay quantiles. The optimal parameters for the neural networks after training are given in Tables 6 and 7 in the appendix for the M/W/1 and W/W/1 system, respectively. We only show results for the DNN architecture since the LSTM architecture performs similarly. We observe that for all variants of the input features, the trained neural networks provide an empirically valid prediction. We also observe that the inclusion of information of the packet arrival pattern leads to tighter delay bounds.

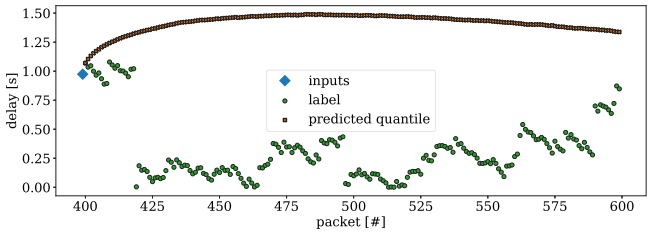
Fig. 5 shows a strong congruence of the mean of the predicted delay quantiles and the empirical quantiles for a W/W/1 queueing system. In comparison to the light tailed exponential service and interarrival times in the example in Fig. 4, we observe that the predicted quantiles here converge slower to their empirical steady state counterparts. Note that, the distance values given in Table 5 to Table 7 allow only for a comparison between results related to one specific system, i.e., to compare the tightness of the predicted bound of that specific system. Since different queueing systems such as M/M/1 and W/W/1 exhibit a different burstiness and thereby different upper bounds for the same value of ϵ the distance metric is not comparable between different systems.

Next, we extend the prediction to systems where the packet arrivals are not independently and identically distributed (iid) but come from an ON-OFF Markov source where in the ON state (state 1) the source produces packets at a constant rate P and in the OFF state (state 2) the packet arrivals stop. The extension weakens the synthetic assumption on the traffic and allows to incorporate a larger class of application traffic patterns. The packet arrivals are characterized through three parameters, i.e., the probability to change states P_{ij} (here denoting moving from state i to state j), the peak rate P, and the mean time T to change states twice [15]. The last metric is considered as proxy for the burstiness of the flow.

We train the neural network and optimize the hyperparameters from Tables 2 and 3. We configure the packet arrival stream such that the packet rate in the ON state is 20 packets per second and the probability

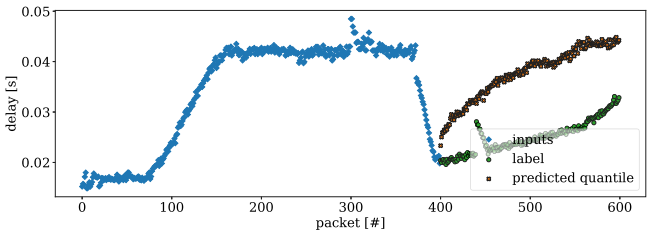


(a) Burstiness T=50

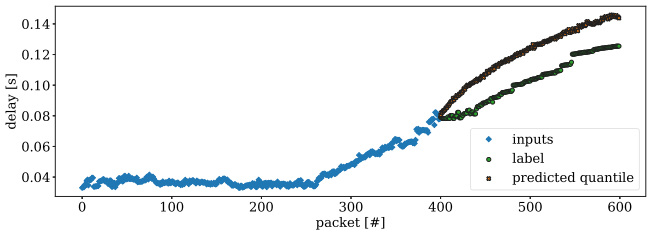


(b) Burstiness T=100

Fig. 6. Delay quantile predictions for a exemplary trace of Markov ON-OFF packet arrivals with burstiness T: The quantile estimator learns the burstiness property. The rising curve for a larger burstiness lasts longer.



(a) Example trace 1



(b) Example trace 2

Fig. 7. Delay quantile predictions using the LSTM model for round-trip time delays obtained from MLAB traces.

to be in the ON state $p_{ON} = \frac{p_{21}}{p_{12}+p_{21}} = 0.75$. The mean time to change states twice is set to 5 s. The service time increments are exponentially distributed with mean of 0.05 s.

We use the delay of packets in the range $[n-r, n)$, where $r \geq 1$ specifies the length of the input feature sequence, to predict future delay quantiles for the packets in the range $[n, n+f)$ for $f > 0$. Note that the quantile prediction provided in Section 3 is point-wise, i.e., we obtain one prediction for a certain packet index. We select $n = 400$ to assume steady state delays, $f = 200$ for a sufficiently large prediction interval. As input feature sequence, we select $r = 100$ and $r = 1$, i.e. the delay sample ranges are from packets $[300 : 400)$ and $[399 : 400)$ for comparison. For both input features we obtain valid quantiles of 0.051 and corresponding test performance of 0.05 and 0.055, respectively. We omit the hyperparameters here, and again the DNN and LSTM network perform similarly good. Fig. 6 shows examples of sampled delay traces in comparison to the predicted bound. Note that in contrast to presenting the *empirical delay quantile* of the traces in comparison to the *predicted quantile bound* in Figs. 4 and 5, here we present the packet delays in comparison to the quantile bound. Observe that the sampled delays in these sub-figures differ in their burst durations. The neural network predicts delay quantile series that incorporate these burst period length, i.e., the rising curve for a larger burstiness lasts longer in Fig. 6b.

4.2. Trace-based evaluation

The experiments in the previous section use synthetic stationary data traces. In contrast, empirical data traces do not show classical statistical properties regarding distribution and load. Hence, we consider next a data set containing empirical traces from MLAB [10]. We defer the real-world evaluation in a CCAM setup using live 5G measurements to the next section. Next, in the consider MLAB real-world traces and instead of one-way delays we use round trip time measurements as training input since clock synchronization cannot be guaranteed in these empirical data sets.

We extract $2 \cdot 10^5$ RTT samples from MLAB pcap-files from Nov. 2020 from one MLab measurement server in Hamburg, Germany. To obtain non-client specific predictions we train a delay quantile prediction neural network based on measurements from all clients connected to this server. We train a DNN and LSTM network as explained before to predict the RTT. Table 4 presents the results after hyperparameter optimization. Both architectures (DNN and LSTM) provide predictions that fulfill the quantile definition. We note that using longer input packet trains tightens the prediction. Fig. 7 shows delay quantile predictions for sample traces using the LSTM model for round-trip time delays obtained from the MLAB traces.

5. Distributed predictive QoS in a city-wide 5G network

In this section, we evaluate trained predictive QoS models that were learned in a federated fashion using network measurements from a city-wide 5G standalone test network. We utilize a federated experimental setup to train the model and then present the results of our training.

5.1. Training the prediction model in a federated setup

To train the ML model in a federated machine learning (FML) setup for PQoS, we employ FLOWER [18], an FL framework for large-scale FL training across heterogeneous devices. FLOWER comprises two crucial components: the Flower Server (or Aggregator) and Flower Clients (or working nodes). The OBUs host the flower clients, which iteratively train the model using locally collected data for multiple local epochs and forward the locally trained model to the Flower server. Meanwhile, the Flower server is deployed on the edge, consolidating the locally trained models from all OBUs, employing the selected aggregation strategy. Subsequently, the edge transmits the aggregated global model to all flower clients, completing one global round.

Here, we only train the LSTM model with the parameters mentioned in Section 3 for PQoS, as the performance of DNN and LSTM showcase similar results. For simplicity, we only use FedAvg [19] aggregation strategy at the flower server. However, the flexibility offered by Flower allows us to choose other aggregation strategies as well. Given the complexity of conducting a hyperparameter search in a federated setting, we opted to perform hyperparameter tuning for our model in a centralized setting and use those parameters while training the model in the federated setup.

Table 4
MLAB data.

Input features		Hyperparameters							Distance		Quantile	
Delay	Interarrivals	Arch	Units 1	Units 2	Units 3	Dropout	Lear. rate	l2 reg	Validation	Test	Validation	Test
[399 : 400)	∅	Dense	200	0	0	0.0	Adaptive	0.001	0.227	0.23	0.05	0.049
[300 : 400)	∅	Dense	200	0	0	0.0	Adaptive	0.001	0.219	0.22	0.052	0.053
[0 : 400)	∅	Dense	30	40	0	0.0	Adaptive	0.001	0.218	0.218	0.054	0.055
[399 : 400)	∅	LSTM	200	0	0	0.0	Adaptive	0.001	0.225	0.228	0.052	0.05
[300 : 400)	∅	LSTM	70	0	0	0.0	Adaptive	0.001	0.219	0.219	0.05	0.049
[0 : 400)	∅	LSTM	70	0	0	0.0	Adaptive	0.001	0.219	0.219	0.051	0.052

5.2. Experimentation setup

For our extensive experimentation, we deploy the Flower Server on an Intel Xeon E5-2620v2 server with 64 GB RAM and Flower Clients on two GPU servers, each hosting five clients. To precisely replicate the FML training on the testbed as the number of actual OBUs deployed on the vehicles is low, we collect real-world application-level network metrics from OBUs driving predefined trajectories in the city-wide 5G standalone testbed. Subsequently, we distribute these metrics among all clients, allowing each client to treat the data as if it had collected its own training data. We distribute the data to emulate specific spatio-temporal measurement runs.

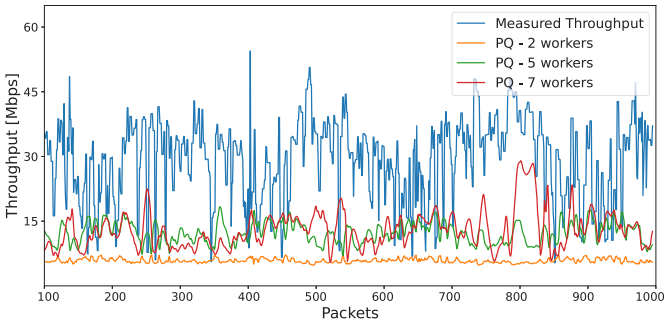
To collect training data, we conducted a data measurement campaign over a private 5G standalone network that spans a medium size city in south Germany. This testbed is a standalone setup which does not carry production traffic that may interfere with the measurement results. Hence, the measurement topology spans from a measurement OBU in a driving vehicle that is connected to the 5G RAN and core. We built a network monitoring tool capable of passively intercepting application traffic to quantify throughput and RTT. For the application running on the OBUs we use a video streaming application and an FFmpeg-server deployed at the 5G edge. The rationale behind the video streaming application is that it arises naturally with 5G based automotive applications such as vehicle teleoperation while providing a continuous stream of network observations. The recorded network metrics, along with the corresponding GPS coordinates of the vehicle, are stored. We conducted multiple data collection sessions spanning various times of the day, resulting in the acquisition of approximately $25K$ data points.

We partitioned the collected data points into three segments: 80% for training, 10% for testing, and the remaining 10% for validation. The training set is further divided among the worker nodes using two distinct splits, namely time segmentation and spatial segmentation. In time segmentation, we divide the training data into 10 time segments, assigning each working node a training set collected during the one time segment. In spatial segmentation, we split the 5G coverage in the city into seven smaller subzones, with each working node exclusively receiving the training set from one subzone.

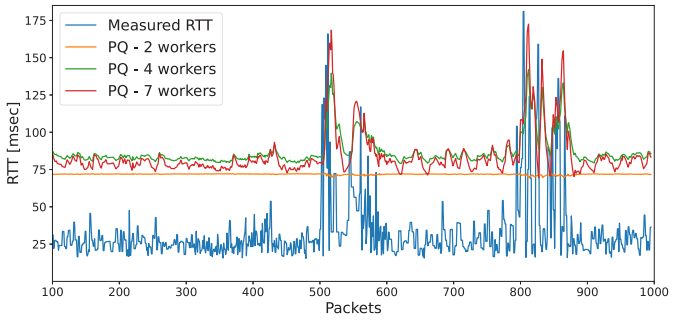
For RTT quantile predictions, we use the RTT of the packet, timestamp, and the GPS coordinates (latitude and longitude) as input features and pinball loss in (3). Similarly, for the throughput quantile predictions, we substitute throughput for RTT in the above-mentioned list of input features and slightly modify the throughput quantile pinball loss (4) to estimate the conditional lower bond for the throughput quantile as

$$L_{Throughput}(y, \hat{y}) = \begin{cases} (1-\varepsilon)(y-\hat{y}) & \text{if } y < \hat{y} \\ \varepsilon(\hat{y}-y) & \text{if } y \leq \hat{y} \end{cases} \tag{4}$$

In all experiments, we fixed the number of local epochs and global rounds to 10 and set training parameters to $n = 0$, $r = 100$, and $f = 5$. Hence, the model uses the previous 100 s of network status measurements, from $[t_0, t_{99}]$ to predict the quantile for the 104th second (t_{104}). This gives the CCAM application adequate time to make a decision after it receives the predicted quantiles.



(a) Throughput sample path



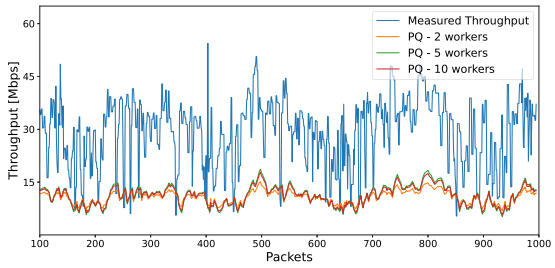
(b) RTT sample path

Fig. 8. Throughput and RTT quantile predictions (lower and upper, respectively) of the LSTM model that are trained over *spatially segmented* training data. Predictive QoS (PQ) for increasing number of workers in the federated learning setting where workers obtain measurements corresponding to vehicle trajectories at different geographical areas.

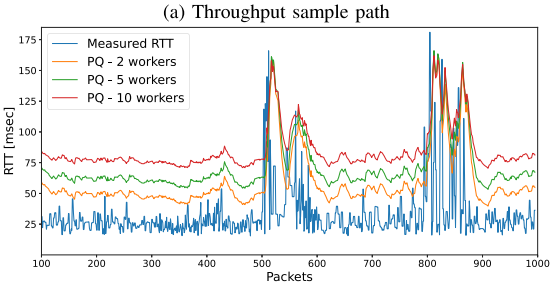
5.3. Results

We start the experimentation by training the LSTM model with sequentially increasing number of worker nodes, using spatially segmented data splits. Note that increasing the number of nodes also increases the number of geographical regions from which data is obtained for training as we consider one worker OBU per subzone. Fig. 8 illustrates an example trajectory of the RTT and throughput quantile predictions generated by the trained model on the test data. Recall that the quantile predictions produce *upper bounds for the RTT and lower bounds for the throughput at the given location in the road segment expected after* 5 *s*. As we increase the number of worker nodes, the model accuracy improves. This improvement arises from the ability to train the model on data spanning all spatial regions within the data collection area. Conversely, with fewer workers, the model is trained on data from a small number of spatial regions which is not statistically mixing. When subjected to the test data, the model performs poorly and the underfitting of the trained model becomes apparent.

Next, we proceed with a similar experiment, training the model as described above, but with time-segmented data splits. The quantile predictions of the model in Fig. 2 for an example trajectory demonstrate that with only 2 worker nodes, it achieves the same level of precision as the model trained with 10 workers even though the model trained with 2 workers lacks data points spanning all collected time spans



(a) Throughput sample path



(b) RTT sample path

Fig. 9. Throughput and RTT quantile predictions over *time segmented* training data. Predictive QoS (PQ) for increasing number of workers in the federated learning setting where workers obtain measurements corresponding to vehicle trajectories at different times..

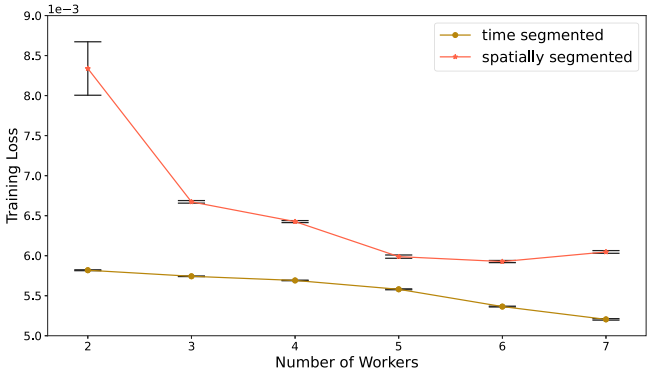


Fig. 10. Pinball loss for RTT training vs. the number of workers in the federated learning setup.

(see Fig. 9). This observation in comparisons to the first set of results highlights that network metrics exhibit more pronounced variations based on spatial factors rather than temporal ones.

To go beyond sample trajectories of the experiment results above we also show the pinball training loss against the number of working clients for the above-mentioned experiments in Fig. 10. We only show the loss for the RTT training as the result for the throughput training is similar. Observe that the loss improves significantly when federated learning is conducted over many spatial regions (note that one worker corresponds to one OBU in one region). In contrast when workers (OBUs) are only separated over time, the improvement in loss is apparent but not as strong as when including more spatially diverse measurements.

Overall, the results suggest that federated learning is a viable method to produce accurate QoS quantile forecasts in CCAM scenarios. The results suggest that statistical multiplexing of the worker vehicles is much important over the spatial component than the time component.

6. Related work

Forecasting the traffic behavior in communication networks using machine learning is successfully applied in various works for which we refer to the surveys [20,21]. Here, we specifically bring into focus approaches that target forecasting the QoS parameters on end-hosts related to our end-to-end prediction of delay quantiles and examples in which the prediction of delays is applied in non end-to-end use cases.

Often, forecasting is applied for throughput or available bandwidth prediction, which is related to adaptive bitrate (ABR) algorithms in streaming applications. These algorithms continuously select variations of video chunks of different quality and thereby size where low quality chunks of smaller size require less throughput. Typically, the optimization goal is to transmit the highest quality chunk subject to it arriving before the content is played out to avoid video stalling. In [22], an ABR algorithm for video streaming is proposed using reinforcement learning which was based among other parameters on download times and measured network throughput. Also in [23] a reinforcement learning approach is applied to ABR to optimize the quality of experience. In [24], a neural network is used to predict the transmission time, which is used to select a suitable chunk in an ABR algorithm. These works go essentially back to [3] in which different throughput predictors are proposed that are not based on machine learning. These works differ from the paper at hand as they are optimized towards available bandwidth estimation for a specific application, namely, video streaming. An online throughput prediction for ABR selection in cellular networks is illustrated in [25] that further includes a prediction of the user's environment such as public transport, indoor, and open air environments, since characteristics of cellular networks differ strongly in these environments. The application and advantage of machine learning to the field of available bandwidth estimation is shown in [26–28]. Further applications may be to estimate the link service as it can be inferred from delay measurements, see e.g. [8,9]. Here, we obtain spatio-temporal bounds on the delay and throughput quantiles in a 5G network mainly for automotive applications.

A bandwidth prediction approach that outputs quantiles of the expected bandwidth at geolocations in automotive scenarios using physical layer, data link layer, speed, traffic, and weather information is described in [29]. Similarly, passive probing parameters from lower network layers are used in [30] to predict the mean end-to-end latency in automotive scenarios. The work in [29] is closest to ours, where in contrast to our work the following differences exist: (i) the raw data measurements in [29] include physical layer characteristics that require additional measurement hardware, where the work at hand circumvents that. (ii) Training in [29] is centralized where this work concentrates on a federated learning approach where workers, i.e., different vehicles observe individual trajectories and individually update their own models. (iii) the authors of [29] only show predictions for the throughput while this work mainly focuses on delay predictions. Finally, (iv) the measurements in [29] are taken in an LTE network with no control of the cross traffic and the cell load where the measurements reported in this paper are in a city-wide 5G standalone testbed which does not carry production traffic. We note that our method is, hence, available as network function in the 5G architecture as illustrated in Section 2. Also closely related to our work is [31] in which QoS distributions for, e.g., the delay are predicted from traffic samples by a conditional variational autoencoder neural network. Here too the following differences exist: (i) The authors use a variant of the Wasserstein distance to compute the loss. Due to its high dimensional integral form the Wasserstein metric is known to be difficult to compute in practice and the stability of the approximation in [31] is not shown in addition to its exact computation being NP-hard. (ii) The authors of [31] do not consider a spatio-temporal QoS estimation but rather QoS prediction for traffic matrices obtained in data center and overlay networks. The traffic types are different from a 5G network and the control of load is missing as the neural network are trained on production traces. In contrast, in the paper at hand we use a computationally much lighter and exactly computable and stable loss function that is geared towards spatio-temporal QoS predictions for packet delay and throughput.

Table 5
M/M/1: Optimal hyperparameters and related empirical quantiles and distances.

Input features		Hyperparameters						Distance		Quantile	
Delay	Interarrivals	Units 1	Units 2	Units 3	Dropout	Lear. rate	l2 reg	Validation	Test	Validation	Test
[399 : 400)	ø	100	0	0	0.0	Adaptive	0.001	0.601	0.601	0.049	0.051
[300 : 400)	ø	200	0	0	0.0	Adaptive	0.001	0.602	0.602	0.049	0.051
[399 : 400)	[400 : 600)	1200	0	0	0.0	Adaptive	0.001	0.505	0.506	0.051	0.053
[300 : 400)	[400 : 600)	200	0	0	0.0	Adaptive	0.001	0.508	0.509	0.052	0.053

Table 6
M/W/1: Optimal hyperparameters and related empirical quantiles and distances.

Input features		Hyperparameters						Distance		Quantile	
Delay	Interarrivals	Units 1	Units 2	Units 3	Dropout	Lear. rate	l2 reg	Validation	Test	Validation	Test
[399 : 400)	ø	1200	0	0	0.0	Adaptive	0.001	1.009	1.012	0.05	0.049
[300 : 400)	ø	200	0	0	0.0	Adaptive	0.001	1.008	1.01	0.05	0.049
[399 : 400)	400 : 600	40	0	0	0.0	Adaptive	0.001	0.934	0.936	0.05	0.049
[300 : 400)	400 : 600	1200	0	0	0.0	Adaptive	0.001	0.939	0.942	0.05	0.05

Table 7
W/W/1: Optimal hyperparameters and related empirical quantiles and distances.

Input features		Hyperparameters						Distance		Quantile	
Delay	Interarrivals	Units 1	Units 2	Units 3	Dropout	Lear. rate	l2 reg	Validation	Test	Validation	Test
[399 : 400)	ø	1200	0	0	0.0	Adaptive	0.001	1.327	1.328	0.051	0.049
[300 : 400)	ø	1200	0	0	0.0	Adaptive	0.001	1.324	1.326	0.051	0.05
[399 : 400)	400 : 600	1200	0	0	0.0	Adaptive	0.001	1.137	1.139	0.051	0.05
[300 : 400)	400 : 600	1200	0	0	0.0	Adaptive	0.001	1.119	1.123	0.053	0.053

On a different note, a throughput forecast is implicitly integrated into transport layer protocols, typically, for congestion control. In [5], the sender predicts a sending rate, so that packets arrive with a delay below a certain value with high probability. Also the congestion control designed in [32] uses machine learning to optimize the sending rate under delay constraints. In [33] congestion control algorithms are designed automatically by training. For simplicity our work here does not integrate on kernel level but is rather a separate application layer function that runs on the OBU and as well as on the network edge and can be queried either in an inter-process communication manner or e.g. through the 5G NEF function.

For data flow rate prediction and optimization, machine learning is applied to SDNs, the survey [34] classifies and summarizes various approaches. For example as early as the work in [35], a routing optimization is conducted based on the delay information obtained from measurements by the destination node. A method of predicting traffic load in the link based on the past traffic samples is shown in [36,37] where the latter work utilizes this estimate to optimize a streaming application. Further, a statistical learning approach for throughput prediction in wireless network environments is described in [38]. The authors in [39] envision the prediction of delays for the optimization of SDNs topology and show that delay can be predicted with a small error, especially the mean end-to-end delay is predicted for various scenarios including variations in topology, network size, traffic distribution, traffic intensity, and routing configurations. Also in [40–42] neural networks, specifically, graph neural networks, are trained to predict performance indicators such as throughput, delay, and jitter for network topologies with input parameters such as traffic, topology, and routing configuration.

The reviewed related work shows the advantages of the application of machine learning techniques to the estimation of end-to-end QoS parameters. Often, the approaches comprise throughput estimation in conjunction with ABR algorithms or with congestion control protocols for end-to-end approaches. Delay and throughput prediction is among other performance metrics used for optimization in SDNs. Complementary, we present the prediction of delay and throughput quantiles for end-to-end traffic flows using quantile regression.

7. Conclusions

Predictive Quality of Service (QoS) plays a pivotal role in enabling seamless cooperative, distributed automotive applications. This study demonstrates that the availability of predicted QoS metrics at vehicles offers valuable insights into the anticipated network behavior along roads, empowering vehicles to adapt automotive application behavior in terms of future information transmissions and cooperative maneuvers accordingly. Employing quantile regression neural networks in a federated learning fashion, this work reliably estimates spatio-temporal delay and throughput quantiles. Initially, we show results for centralized training, showcasing empirical validation and demonstrating the model capacity to recover classical results from queueing theory concerning delay quantiles. Further, beyond empirical validation, the study extends its assessment to encompass more intricate scenarios involving varying load dynamics, mixed arrival patterns, and diverse service processes. Subsequently, the investigation leverages federated machine learning, an apt methodology for CCAM applications. It employs neural network models trained on a city-wide 5G standalone testbed network data to showcase application-level delay and throughput quantile regression. Future work includes expanding the quantile predictions from point-wise estimates to sample path predictions.

CRediT authorship contribution statement

Nehal Baganal-Krishna: Data curation, Investigation, Methodology, Software, Writing – original draft, Visualization. **Ralf Lübben:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Writing – original draft, Visualization. **Eirini Liotou:** Conceptualization, Writing – review & editing, Funding acquisition. **Konstantinos V. Katsaros:** Conceptualization, Funding acquisition, Writing – review & editing. **Amr Rizk:** Conceptualization, Formal analysis, Funding acquisition, Methodology, Supervision, Writing – original draft, Writing – review & editing.

Data availability

The authors do not have permission to share data.

Acknowledgments

This research was performed in the context of the 5G-IANA project, co-funded by the European Commission under the Horizon 2020 Research and Innovation Programme (grant agreement No 101016427).

Appendix A. Comparison with analytical results

In the following, we empirically show that the estimates of the quantile estimation approach coincide with analytical results obtained for a tractable analytical example.

A.1. M/M/1 System

We start with an example of a well understood queueing system, the M/M/1 system, with one server having exponentially distributed service times with parameter μ, as well as, exponentially distributed inter-packet arrival times with parameter λ. It is known that an M/M/1 queueing system has a steady-state response time distribution of

$$P[W > a\overline{W}] = e^{-a} \tag{5}$$

with expected response time $\overline{W} = \frac{1}{1-\rho}\frac{1}{\mu}$ and the shorthand notation $\rho := \lambda/\mu$ for $a \geq 0$. Now, fixing the violation probability $e^{-a} = 1 - \epsilon$, our approach estimates the corresponding delay quantile $a\overline{W}$, which we denoted above as W^{ϵ}. For $1 - \epsilon = 0.95$, the related response time quantile is 0.599.

To validate the empirical quantile estimation approach we use training data generated from simulations and compare the quantile estimate $W^{\epsilon}(n)$ of packet n to the analytical delay quantile $a\overline{W}$. We obtain simulation data using the discrete event simulator Omnet++, where we simulate an M/M/1 queueing system and record arrival and departure times $T_A(n), T_D(n)$, respectively, to compute packet delays.

We simulate $3 \cdot 10^5$ packet traces for different utilizations. We split the data into a training, validation, and test set. The training set comprises 80%, the validation set 10%, and the test set 10% of the traces. For the prediction of the delay quantile, we extract values from the steady state delay. Hence, the comparison with the analytical steady state quantile $a\overline{W}$ from (5) is meaningful for packets far enough in future such that they can be considered in steady state. We select $n = 400$ to assume steady state delays, $f = 200$ for a sufficiently large prediction interval, and $r = 1$ and $r = 100$, respectively.

For our estimation approach, we use the input features shown in Table 1, i.e., solely the packet delays and in comparison the combination of the packet delays and known future packet arrival time points. Table 5 shows the optimal hyperparameters, related quantiles, and distance metrics of the validation and test sets.

Fig. 4 shown in Section 4.1 compares the empirical packet-wise quantiles W^{ϵ} to the mean predicted quantile for the test data set. Recall that the analytical value for this example accounts to $a\overline{W} = 0.599$. The prediction series converges to the analytical and empirical quantile quickly.

Overall, we find that the trained neural networks are able to predict packet delays for that the quantile condition (2) holds. We also note that the prediction improves if knowledge or estimates of future arrivals, i.e., the next packet interarrival times, are included for the prediction (Table 5). The delay quantile is still correctly predicted according to the given ϵ, but the distance decreases, i.e., the prediction returns a tighter bound. The improvement using this additional information is relevant for applications which influence future packet arrivals, e.g., through selecting videos qualities to be transmitted, selecting sensor status sending times, or encoder settings in video streaming scenarios.

A.2. Selected hyperparameters

Tables 5, 6, and 7 display the selected parameters after hyperparameter tuning and the distance as well as the empirical quantile for the queueing systems M/M/1, M/W/1, and W/W/1. For each system four experiments were conducted with a short and long delay sequence as well as inter-arrivals of upcoming packets. For all systems a valid quantile is predicted. The experiments were conducted with a DNN and LSTM architecture, which perform similarly, where we only show the results of the DNN architecture.

References

[1] R. Koenker, A. Chesher, M. Jackson, Quantile Regression, in: Econometric Society Monographs, Cambridge University Press, 2005.
[2] J.W. Taylor, A quantile regression neural network approach to estimating the conditional density of multiperiod returns, J. Forecast. 19 (4) (2000) 299–311.
[3] X. Yin, A. Jindal, V. Sekar, B. Sinopoli, A control-theoretic approach for dynamic adaptive video streaming over HTTP, SIGCOMM Comput. Commun. Rev. 45 (4) (2015) 325–338.
[4] K. Spiteri, R. Urgaonkar, R.K. Sitaraman, BOLA: Near-optimal bitrate adaptation for online videos, IEEE/ACM Trans. Netw. 28 (4) (2020) 1698–1711.
[5] K. Winstein, A. Sivaraman, H. Balakrishnan, Stochastic forecasts achieve high throughput and low delay over cellular networks, in: Proc. USENIX NSDI, 2013, pp. 459–471.
[6] X. Liu, K. Ravindran, D. Loguinov, A queueing-theoretic foundation of available bandwidth estimation: Single-hop analysis, IEEE/ACM Trans. Netw. 15 (4) (2007) 918–931.
[7] X. Liu, K. Ravindran, D. Loguinov, A stochastic foundation of available bandwidth estimation: Multi-hop analysis, IEEE/ACM Trans. Netw. 16 (1) (2008) 130–143.
[8] R. Lübben, M. Fidler, J. Liebeherr, Stochastic bandwidth estimation in networks with random service, IEEE/ACM Trans. Netw. 22 (2) (2014) 484–497.
[9] A. Rizk, M. Fidler, On the identifiability of link service curves from end-host measurements, in: Network Control and Optimization, Euro-NF Workshop, NET-COOP, in: Lecture Notes in Computer Science, vol. 5425, 2008, pp. 53–61.
[10] Measurement Lab, The M-Lab NDT Data Set.https://measurementlab.net/tests/ndt.
[11] R. Lübben, A. Rizk, Tailing: tail distribution forecasting of packet delays using quantile regression neural networks, in: IEEE International Conference on Communications (ICC), 2023, pp. 377–383.
[12] 3GPP, 3Rd generation partnership project; technical specification group services and system aspects; architecture enhancements for 5G system (5GS) to support network data analytics services (release 16): TS23. 288 V16. 4.0, 2020.
[13] A. Pásztor, D. Veitch, Active probing using packet quartets, in: Proceedings of the 2nd ACM SIGCOMM Internet Measurement Workshop, 2002, pp. 293–305.
[14] M. Jain, C. Dovrolis, End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput, IEEE/ACM Trans. Netw. 11 (4) (2003) 537–549.
[15] M. Fidler, A. Rizk, A guide to the stochastic network calculus 17(1), 2015, pp. 92–105.
[16] R. Koenker, G. Bassett, Regression quantiles, Econometrica 46 (1) (1978) 33–50.
[17] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, A. Talwalkar, Hyperband: A novel bandit-based approach to hyperparameter optimization, J. Mach. Learn. Res. 18 (185) (2018).
[18] D.J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K.H. Li, T. Parcollet, P.P.B. de Gusmão, N.D. Lane, Flower: A friendly federated learning research framework, 2022, arXiv:2007.14390.
[19] H.B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, 2023, arXiv:1602.05629.
[20] R. Boutaba, M.A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, O.M. Caicedo, A comprehensive survey on machine learning for networking: evolution, applications and research opportunities, J. Internet Serv. Appl. 9 (1) (2018).
[21] C. Zhang, P. Patras, H. Haddadi, Deep learning in mobile and wireless networking: A survey 21(3), 2019, pp. 2224–2287.
[22] H. Mao, R. Netravali, M. Alizadeh, Neural adaptive video streaming with pensieve, in: Proc. ACM SIGCOMM, 2017, pp. 197–210.
[23] M. Gadaleta, F. Chiariotti, M. Rossi, A. Zanella, D-DASH: A Deep Q-Learning Framework for DASH Video Streaming, IEEE Transactions on Cognitive Communications and Networking 3 (4) (2017) 703–718.
[24] F.Y. Yan, H. Ayers, C. Zhu, S. Fouladi, J. Hong, K. Zhang, P. Levis, K. Winstein, Learning in situ: a randomized experiment in video streaming, in: Proc. USENIX NSDI, 2020, pp. 495–511.
[25] C. Qiao, G. Li, J. Wang, Y. Liu, NEIVA: Environment identification based video bitrate adaption in cellular networks, in: Proc. IEEE/ACM IWQoS, 2019.
[26] S.K. Khangura, M. Fidler, B. Rosenhahn, Machine learning for measurement-based bandwidth estimation, Comput. Commun. 144 (2019) 18–30.
[27] S.K. Khangura, S. Akin, Measurement-based online available bandwidth estimation employing reinforcement learning, in: Proc. IEEE ITC, 2019, pp. 95–103.

[28] S.K. Khangura, S. Akin, Online available bandwidth estimation using multiclass supervised learning techniques, Comput. Commun. 170 (2021) 177–189.
[29] D. Schäufele, M. Kasparick, J. Schwardmann, J. Morgenroth, S. Stańczak, Terminal-side data rate prediction for high-mobility users, in: Proc. IEEE VTC, 2021.
[30] D.F. Külzer, F. Debbichi, S. Stańczak, M. Botsov, On latency prediction with deep learning and passive probing at high mobility, in: Proc. IEEE ICC, 2021.
[31] S. Xiao, D. He, Z. Gong, Deep-Q: Traffic-driven QoS inference using deep generative network, in: Proc. ACM Workshop NetAI, 2018, pp. 67–73.
[32] M. Dong, T. Meng, D. Zarchy, E. Arslan, Y. Gilad, B. Godfrey, M. Schapira, PCC vivace: Online-learning congestion control, in: Proc. USENIX NSDI, 2018, pp. 343–356.
[33] A. Sivaraman, K. Winstein, P. Thaker, H. Balakrishnan, An experimental study of the learnability of congestion control, SIGCOMM Comput. Commun. Rev. 44 (4) (2014) 479–490.
[34] R. Etengu, S.C. Tan, L.C. Kwang, F.M. Abbou, T.C. Chuah, AI-assisted framework for green-routing and load balancing in hybrid software-defined networking: Proposal, challenges and future perspective, IEEE Access 8 (2020) 166384–166441.
[35] P. Baran, On distributed communications networks, IEEE Trans. Commun. Syst. 12 (1) (1964) 1–9.
[36] T. Anjali, C. Scoglio, J.C. De Oliveira, L.C. Chen, I.F. Akyildiz, J.A. Smith, G. Uhl, A. Sciuto, A new path selection algorithm for MPLS networks based on available bandwidth estimation, in: International Workshop on Quality of Future Internet Services, Springer, 2002, pp. 205–214.
[37] D. Bhat, A. Rizk, M. Zink, R. Steinmetz, SABR: Network-assisted content distribution for Qoe-driven ABR video streaming, ACM Trans. Multimedia Comput. Commun. Appl. (TOMM) 14 (2s) (2018) 1–25.
[38] C. Rattaro, P. Belzarena, Throughput prediction in wireless networks using statistical learning, in: LAWDN-Latin-American Workshop on Dynamic Networks, 2010, pp. 4–p.
[39] A. Mestres, E. Alarcón, Y. Ji, A. Cabellos-Aparicio, Understanding the modeling of computer network delays using neural networks, in: Proc. ACM Big-DAMA Workshop, 2018, pp. 46–52.
[40] K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, A. Cabellos-Aparicio, RouteNet: Leveraging Graph Neural Networks for Network Modeling and Optimization in SDN, IEEE J. Sel. Areas Commun. 38 (10) (2020) 2260–2270.
[41] M. Ferriol-Galmés, J. Suárez-Varela, J. Paillissé, X. Shi, S. Xiao, X. Cheng, P. Barlet-Ros, A. Cabellos-Aparicio, Building a digital twin for network optimization using graph neural networks, Comput. Netw. 217 (2022).
[42] B. Jaeger, M. Helm, L. Schwegmann, G. Carle, Modeling TCP performance using graph neural networks, in: Proc. ACM Workshop on Graph Neural Networking, GNNet, 2022, pp. 18–23.

Nehal Baganal-Krishna completed his M.Sc. in Communication and Information Technology in 2021 at Ulm University, Germany. Subsequently, he embarked on his Ph.D. journey in 2021 as a member of the Networks and Communications Systems Group at the University of Duisburg-Essen, Germany. His doctoral research revolves around exploring Network Applications for CCAM on Programmable Hardware, specifically focusing on P4-programmable switches and FPGAs.

Ralf Lübben received the Doctoral degree in computer engineering from Leibniz University Hannover, Hanover, Germany, in 2014. From 2014 to 2018, he was a research engineer with the Robert Bosch GmbH, Gerlingen, Germany, Since 2018, he has been a professor of computer networks with the Flensburg University of Applied Sciences, Flensburg, Germany.

Dr. Eirini Liotou, holds a PhD degree from the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens since 2017. She obtained the MSc in Informatics and Telecommunications from the National and Kapodistrian University of Athens and the MSc in Communications and Signal Processing from the Imperial College of London. She received the Diploma in Electrical and Computer Engineering from the National Technical University of Athens in 2006. She has worked as a Software Engineer in Siemens AG and as a Senior Software Engineer in Siemens Enterprise Communications in the R&D department for the SME business sector. She has participated in various EU projects, national projects and COST actions. Her main research interests include Quality of Service and Quality of Experience management, adaptive video streaming, and SDN/NFV in mobile cellular networks. Since September 2021, she is working as a Project Manager / Senior Researcher for EU Research Projects at the I-SENSE/ICCS group.

Konstantinos V. Katsaros, Ph.D., is leading the Intelligent Networks & Services Team at I-SENSE / ICCS NTUA (Greece). His R&I agenda includes management & orchestration (M&O) of Federated Learning workloads, incl. drift management aspects, the support of MLOps and Distributed ML-as-a-Service in B5G/6G networks, Extreme Edge Computing and Predictive QoS in Automotive Environments. Previously, he held a Senior Researcher position at Intracom S.A. Telecom Solutions (Greece), and Research Associate positions at University College London (UK) and Telecom ParisTech (France). Dr. Katsaros has experience in the areas of network function virtualization (NFV), multi-access edge computing (MEC) and software-defined networking (SDN) for 5G networks. He also further holds extensive experience in information-centric networking, IoT interoperability, smart grids, inter-domain carrier services and multicast/broadcast service provisioning over cellular networks. Dr. Katsaros holds a BSc (2003) in Informatics, a MSc (2005) in Computer Science and a PhD (2010) in Computer Science, from AUEB (Greece). He has more than 80 publications and talks in numerous academic conferences, workshops and peer reviewed journals, with more than 3300 citations.

Amr Rizk received the doctoral degree (Dr.-Ing.) from the Leibniz Universität Hannover, Germany, in 2013. After that he held postdoctoral positions at University of Warwick, UMass Amherst and the TU Darmstadt, Germany. From 2019 to 2021 he was an assistant professor at Ulm University, Germany. Since 2021 he is a professor at the department for computer science at the University of Duisburg–Essen, Germany. He is interested in performance evaluation of communication systems, stochastic models of networked systems and their applications to communication systems.