Data-Driven End-to-End Delay Violation Probability Prediction with Extreme Value Mixture Models

Seyed Samie Mostafavi, György Dán, James Gross
School of Electrical Engineering and Computer Science
KTH, Royal Institute of Technology
Stockholm, Sweden
{ssmos,gyuri,jamesgr}@kth.se

ABSTRACT

With the advent of edge computing, there is increasing interest in wireless latency-critical services. Such applications require the end-to-end delay of the network infrastructure (communication and computation) to be less than a target delay with a certain probability, e.g., 10^{-2}-10^{-5}. To deal with this guarantee level, the first step is to predict the transient delay violation probability (DVP) of the packets traversing the network. The guarantee level puts a threshold on the tail of the end-to-end delay distribution; thus, it makes data-driven DVP prediction a challenging task. We propose to use the extreme value mixture model in the mixture density network (MDN) method for this task. We implemented it in a multi-hop queuing-theoretic system to predict the DVP of each packet from the network state variables. This work is a first step toward utilizing the DVP predictions, possibly in the resource allocation scheme or queuing discipline. Numerically, we show that our proposed approach outperforms state-of-the-art Gaussian mixture model-based predictors by orders of magnitude, in particular for scenarios with guarantee levels above 10^{-2}.

KEYWORDS

edge computing, delay violation probability, time sensitive networks, extreme value mixture models

ACM Reference Format:
Seyed Samie Mostafavi, György Dán, James Gross. 2021. Data-Driven End-to-End Delay Violation Probability Prediction with Extreme Value Mixture Models. In *The Sixth ACM/IEEE Symposium on Edge Computing (SEC '21), December 14–17, 2021, San Jose, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3453142.3493506

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SEC '21, December 14–17, 2021, San Jose, CA, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8390-5/21/12…$15.00
https://doi.org/10.1145/3453142.3493506

1 INTRODUCTION

Latency-critical applications are emerging as important use cases for edge computing systems. Examples of such applications include cyber-physical systems (CPS) and Human-in-the-loop (HITL) applications, where the end-nodes require timely response from the edge server for smooth operation. Such *closed-loop* applications with real-time requirements have traditionally been either implemented with embedded computers, i.e., eliminating the need for communication, or communication was performed over wired networks with dedicated capacity. With the advent of edge computing, there is increasing interest for deploying networked closed-loop applications over wireless links as well. For instance, wireless time-sensitive networks (TSN) are emerging based on IEEE 802.11ax, and 5G cellular mobile networks and edge computing are expected to enable reliable, delay bounded, high-bandwidth industrial communications. Hence, it is expected that the need for wireless closed-loop services will grow substantially over the next years [3, 4]. For latency-critical applications, the end-to-end delay of the network must be guaranteed not to violate a target delay with a certain probability. For instance, there are HITL applications that typically have delay targets around 100 ms with the guarantee levels in the range of $10^{-2} - 10^{-3}$ [11]. Importantly, it is the tail of the end-to-end delay distribution which is critical for the guarantee level and not the average delay.

The probability that a packet will not successfully traverse the closed-loop within a certain delay bound or delay violation probability (DVP) is a critical metric in this context. In order to deal with the guarantee level efficiently, the transient DVP must be accurately predicted based on the instantaneous state of the network. Then, the edge infrastructure (wireless network and compute node) could utilize DVP predictors to deal with the delay bound and attune their resource allocation scheme or queue management policy. In this work we study how to estimate the transient DVP using a model which is accurate for the tail as well as the average latency.

1.1 Related Works

The problem of delay prediction in communication networks has been addressed by several previous studies. Previous studies typically focus either on analytical approaches or exploit data-driven methods. For example, authors in [2], attempt to characterize the DVP by finding probabilistic bounds on the end-to-end delay of a multi-hop network. They obtain the bounds using stochastic network calculus. However, the bounds are only applicable to i.i.d service processes, i.e., service times cannot be correlated .

Among the data-driven approaches, there are a few works that chose to implement and evaluate an end-to-end latency predictors on a real network. One of them is [5], where the authors use a histogram as the conditional distribution and rely on system variables such as position, time, radio channel, and received signal power for latency estimation. In [9], a Gaussian mixture model with mixture density network (MDN) is used to predict service metrics such as response time or frame rate from a set of infrastructure measurements in a cloud environment.

Another common data-driven approach is to design the predictors in a queuing theoretic context [6, 12]. Authors in [6] propose to predict the expected waiting time in a single multi-server queue from the queuing delay history, and compare their approach to common estimators based on the queue length. Authors in [12] introduce a framework to predict waiting times in service queues. They consider various predictors, including delay-history-based predictors and snapshot predictors. Contrary to ours, these data-driven works focus on average delay prediction and they do not consider the distribution of the delay. A closely related queuing-theoretic data-driven work is [7], where the authors used MDNs to predict the distribution of the end-to-end delay given the network's state, defined as the tandem queues' backlogs. The predictor in [7] is built using Gaussian mixture models (GMM), which works well for the body of the distribution, but is not good at predicting the tail probabilities, which is the focus of our work.

1.2 Contributions

The main contribution of this work lies in proposing a data-driven predictor that accurately and efficiently predict the DVP of packets for real-time closed-loop applications. Our proposed predictors do not assume i.i.d. service processes, make no assumption about the arrival process, and are applicable to networks with general stationary service time processes. Numerically, we show that our proposed approach outperforms state-of-the-art predictors substantially, in particular for scenarios with limited training data. A key feature of the proposed approach that ensures accurate tail prediction is to leverage extreme value theory models. To the best of our knowledge, ours is the first data-driven transient DVP predictor that is designed and evaluated extensively for tail probability prediction.

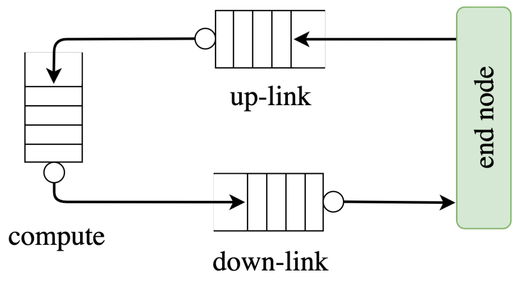


Figure 1: Illustration of the considered queuing network model for closed-loop applications.

2 SYSTEM MODEL AND PROBLEM STATEMENT

We consider the end-to-end latency of *tasks* sent over a network from an end node to a compute node and back, as illustrated in Figure 1. The end node interfaces with the controlled plant (sensor/actuator) or a human user (augmented reality, for instance). The compute node provides computational capacity. The end-to-end latency of a task consists of the following. The end node first transmits a data packet (e.g. the sensor data) to the compute node (e.g. control unit) to be processed. After receiving the packet, the compute node executes a computation task on the data and sends a response packet (e.g. the actuation data) to the end node. Once the packet is received by the end node, it is processed instantaneously without any further delay.

We model the considered closed loop system as a tandem queuing network with three queues, as shown in Figure 1. Each queue has a single server, an infinite buffer, and serves tasks in first-in, first-out (FIFO) order. Tasks are generated at the end node and queued for the up-link transmission. Once they are successfully received at the compute node, they are buffered again until the compute node becomes available and can process the corresponding task. The resulting feedback is again first queued upon availability of the down-link resources, and then transmitted back to the end node.

Let us denote by T_n the generation time of task n, and by Y_n its sojourn time or end-to-end delay in the queuing network. The completion time of task n is then $T_n + Y_n$. Similarly, we denote by W_n^i the queueing time of task n in queue $i \in 1, 2, 3$, by S_n^i its service time in queue i, and by $Y_n^i = W_n^i + S_n^i$ the resulting sojourn time. Clearly, $Y_n = Y_n^1 + Y_n^2 + Y_n^3$. We denote by $F_{S^i}(s)$ the distribution of the service time S^i, and assume that the service process is stationary. The distributions and their high-order characteristics are considered to be unknown.

Problem Statement: We are interested in predicting the DVP of task n, i.e., the probability that the sojourn time of task n will exceed an end-to-end latency target τ, given the state of the network at time t, i.e.,

$$\varphi_{n,t} = \mathbb{P}\left[Y_n > \tau \mid Y_n \geq \delta_{n,t}, X_t\right] , \quad (1)$$

where $\delta_{n,t} = t - T_n$ denotes the time that task n has already spent traversing the network path, and X_t is the state of the network at time t.

3 APPROACH

At time instant t we define the state of the network path by the vector Q_t with domain $\mathcal{Q} \subset \mathbb{Z}_+^3$ which denotes the number of tasks present in the queues, and the vector M_t with domain $\mathcal{M} \subset \mathbb{R}_+^3$ which denotes the time in service of the tasks being served at time t. For task n at time instant t, we can write

$$\varphi_{n,t} = \mathbb{P}\left[Y_n > \tau \mid Y_n \geq \delta_{n,t}, M_t, Q_t\right] . \quad (2)$$

3.1 Conditional Density Estimation with Mixture Density Networks

The above task of estimating the transient delay violation probabilities of the tasks in the network path falls into the domain of conditional density estimation (CDE). We define the CDE problem as estimating the probability density of end-to-end delay Y given the network state X, which is a vector of predictive variables (or set of variables) with domains $\mathcal{Y} \subset \mathbb{R}_+$ and $\mathcal{X} \subset \mathbb{R}^{d_x}$ in the form $\mathbb{P}[Y \mid X = \boldsymbol{x}]$. Using a collection of i.i.d samples $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_N, y_N)\}$, the goal is to obtain a function that maps the values of the predictive variable X into the space of probability densities over the possible values of Y. Formally, the goal of CDE is to obtain an estimate $\hat{p}(Y \mid X = \boldsymbol{x})$ such that $\hat{p}(Y \mid X = \boldsymbol{x}) \approx \mathbb{P}[Y \mid X = \boldsymbol{x}]$.

In parametric density estimation, we fit a parametric model such as a GMM to the set of data samples. This parametric density function is described by a finite-dimensional parameter θ [1]. For example, θ could be a vector of the weights, locations, and variances of the GMM density function. To map the values of X into the space of the probability density parameter, it is common to obtain the parameters θ through another function h_ω in the form $\theta_t = h_\omega(\boldsymbol{x}_t)$. MDN is a well-known method that uses a fully connected neural network as h_ω to control the parameters of the conditional density estimate $\hat{p}_\theta$ [1]. In MDN, maximum likelihood estimation (MLE) is used for estimating the parameters ω, i.e., ω is chosen so that the conditional likelihood of the samples $\mathcal{D}$ is

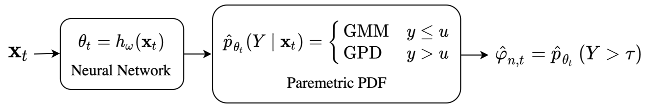

$\mathbf{x}_t$
$\theta_t = h_\omega(\mathbf{x}_t)$
Neural Network
$\hat{p}_{\theta_t}(Y \mid \mathbf{x}_t) = \begin{cases} \text{GMM} & y \leq u \\ \text{GPD} & y > u \end{cases}$
Paremetric PDF
$\hat{\varphi}_{n,t} = \hat{p}_{\theta_t}(Y > \tau)$

Figure 2: Block diagram of the predictor, showing inputs, components, and outputs.

maximized. This is equivalent to minimizing the Kullback-Leibler divergence (KL-divergence) between the empirical data and the parametric density $\hat{p}_\theta$ [1].

Let $X_t = \{M_t, Q_t\}$ with dimension $d_x = 6$ represent the network state. Based on Equation 2, we can estimate the transient DVP of task n at time t in the form

$$\hat{\varphi}_{n,t} = \hat{p}_\theta\left(Y > \tau \mid X_t\right) . \quad (3)$$

The next step is to establish the parametric distribution $\hat{p}_\theta$ to be used at the core of the CDE problem. In the following we show how to to leverage the models introduced by Extreme value theory (EVT) for the tail estimation in addition to the Gaussians for this task.

3.2 Transient Probability Estimation by Extreme Value Mixture Models

Many parametric density distributions with different characteristics have been proposed to be used for CDE problems. Mixture density networks, normalizing flow networks, and kernel density networks are the most common ones. In these methods, the Gaussian density function is commonly used to form a weighted mixture (MDNs) or is transformed through a series of invertible mappings (normalizing flows). These approaches have been extensively studied and were shown to be accurate for normal use cases where the main point of concern is how well the parametric density distribution fits the body of the empirical distribution [8].

Nonetheless, our use case requires high accuracy at the tail of the fitted probability density, which can make the application of GMM-based solutions problematic. The output of the GMM-based solutions is a density function whose tail probability decreases exponentially fast. If the empirical distribution is not light tailed then GMM needs lots of samples together with an expansion of centers. Otherwise, it leads to significant probability error for low probability events, e.g., in the order of 10^{-5}. To the best of our knowledge, there are no established methods that can efficiently deal with the prediction of such extreme events in the context of a CDE problem. Our proposal is to leverage the generalized Pareto distribution (GPD) in the parametric distribution $\hat{p}_\theta$ such that the focus is not only on the bulk of the distribution, but also on the tail.

The GPD was introduced in extreme value theory to estimate the tail of univariate distributions. In the Peaks over

Thresholds (PoT) method, the GPD is fit to the exceedances of a random variable above a suitable threshold. This is motivated by Pickand´s theorem, which states that for most random variables, the exceedances converge to a GPD as the threshold tends to the right endpoint of the support of the underlying distribution. Letting u be the tail threshold, ξ the tail index, and β the scale parameter, the GPD located above u is given in the form

$$g(y|\beta,\xi,u)=\begin{cases}\frac{1}{\beta}\left(1+\frac{\xi}{\beta}(y-u)\right)^{-1/\xi-1} & \xi\neq 0\\ \frac{1}{\beta}e^{\frac{y-u}{\beta}} & \xi=0,\end{cases} \tag{4}$$

where $y \geq u$, $\xi \geq 0$, and $u \leq y \leq u-\beta/\xi$ when $\xi < 0$.

The tail index ξ of a distribution characterizes the heaviness of the tail of the distribution. When $\xi > 0$, the distribution is heavy-tailed and the tail decreases at a subexponential rate, like for the Pareto, the α-stable or the Student-t distributions. When $\xi = 0$, the distribution is light-tailed, that is, the tail decreases exponentially. For example the Gaussian, the exponential and the log-normal distributions are light-tailed. Fitting a GPD to the exceedances provides an estimator of the tail index of the underlying distribution [10].

It is clear that the GPD has favorable properties for characterizing the tail of the delay distribution. At the same time, the GMM is vital for capturing the bulk of the delay distribution; the MDN system enables us to estimate the conditional probability density from a predictive variable. We introduce the following parametric mixture function that encapsulates the GPD tail model in combination with the GMM to be fused with the neural network of a MDN in the form:

$$\hat{p}(y|\theta=h_{\omega}(\boldsymbol{x}))=\begin{cases}f(y|\phi) & y\leq u\\ [1-F(u|\phi)]g(y|\beta,\xi,u) & y>u,\end{cases} \tag{5}$$

where $f(y|\phi)$ and $F(y|\phi)$ denote the GMM's probability density function (PDF) and cumulative density function (CDF) respectively, $g(y|\beta,\xi,u)$ denotes the GPD's PDF, u is the tail threshold, and θ is the collection of parameters ϕ, β, ξ, and u.

In this new MDN scheme, tail parameters of the distribution (threshold, tail index, and scale) are being estimated by the neural network in addition to the bulk distribution parameters. Essentially, fitting a GPD to the empirical data for estimating a density function tail index and tail threshold accurately requires a lot of samples. In our method, the neural network explicitly learns the relation between the seen states and the tail behaviour of the delay distribution and generalizes it to the unseen states. Therefore, as shown in the numerical results, this novel method requires much less empirical samples to achieve the same level of tail probability estimation accuracy compared to the state-of-the-art GMM-based MDNs.

Other extreme value mixture models have been introduced in several works to automate the tail threshold estimation [10]. They encase the usual GPD tail model in combination with other kinds of bulk distributions or apply a continuity (or higher order) constraint at the threshold. To the best of our knowledge, non of them proposed or studied an extreme value mixture model to be fused with a neural network of an MDN system.

4 NUMERICAL RESULTS

In this section, we provide the numerical performance evaluation for our proposed extreme mixture model (EMM)-based DVP predictor. In the following subsection all the steps for benchmarking the predictors, performance metrics, and key variables of the experiments are introduced[1].

4.1 Methodology

We evaluate the proposed predictor using an event-based, continous-time 3-hop tandem queuing system implemented in the MATLAB Simulink framework. Three DVP predictors are implemented before each queue as shown in figure 3, and utilize only Q_t as for the vector of predicting variables. Tasks arrive at the first hop periodically with the rate $\lambda = 0.9$. The three queues have the same service time distribution with service rates $\mu^1 = \mu^2 = \mu^3 = 1$. The service times are i.i.d drawn from a heavy-tail Gamma distribution obtained as follows. We spliced a GPD to a Gamma distribution at its 0.8 quantile to have a parametric tail. The parameters that we used are $\theta_{\text{Gamma}} = 0.2$, $k_{\text{Gamma}} = 5$, and $\xi_{\text{GPD}} = 0.2$ for all service processes.

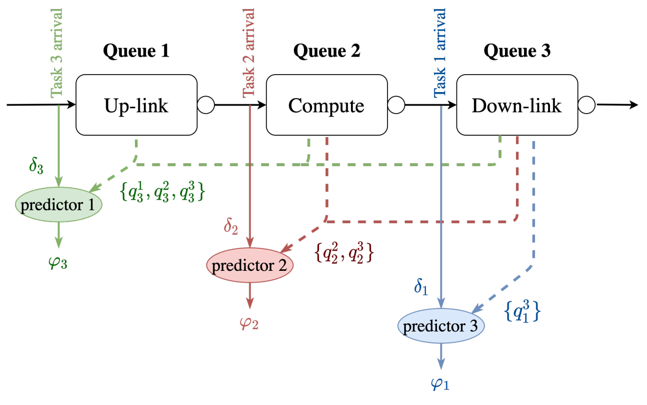


Figure 3: Structure of the queuing network model with DVP predictors implemented for the evaluation.

Predictors: The predictions are implemented using Tensorflow, seperated from the MATLAB Simulink queue simulation environment. Our EMM-based predictor is an extension

[1]The reproducible experiments: https://github.com/samiemostafavi/data-driven-dvp-prediction

to the MDN implementation of [8]. It has 2 Gaussian components in all of the experiments while GMM predictors have 3 Gaussian components. The neural network of the predictors has 2 hidden layers of size 16 by default. The output layer of the neural network governs the parameters of the mixture density function. Each parameter type uses a different activation function. For instance, the mixture weights of a GMM with 3 centers are controlled by the softmax activation function applied to 3 output layer tensors.

Training: To train the predictors we start by creating the training dataset. In order to have N training samples, we run the queuing simulation for the timespan of $(N/\lambda) \times 1.1$. A 10% more duration is added to have all of the training samples from the queues in steady state. Then, the initial 10% samples will be removed and we save the remaining N completed tasks. Finally, Adam weight optimizer trains the neural networks with the likelihood-based loss function. There is no regularization, weight decay, or droupout used in the trainings.

Evaluation: The implemented predictors estimate DVPs conditioned on the queue backlogs $Q = \boldsymbol{q}$ and for any target delay τ. We can obtain the true DVP by counting the number of delay violations in a set of conditioned records and dividing it by the size of the set. Assume index i to denote the sojourn times of N tasks that have seen network state $\boldsymbol{q}$. The true DVP could be written as

$$\mathbb{P}\left[Y_n > \tau \mid Q = \boldsymbol{q}\right] = \lim_{N\to\infty} \frac{\sum_{i=1}^{N} \mathfrak{I}\left[y_i \geq \tau\right]}{N}, \tag{6}$$

where $\mathfrak{I}[\cdot]$ is the indicator function equalling 1 if the argument is true and 0 otherwise. The size of the evaluation dataset must be large enough to have a sufficient precision for the ground truth. For instance, to obtain the ground truth for a state that forms 1% of the evaluation dataset, we use at least 10^9 samples. Hence, we will have 10^7 conditional samples which is enough to cover the minimum target delay quantile which is 10^{-5}.

We evaluate the predictors for the most common states of the ground truth (15,30, and 60 states for each predictor respectively) and we refer to them as *evaluation states* in the results. As for the target delays, we measure 10^{-2} to 10^{-5} quantiles of the remaining sojourn time conditioned on the evaluation states and use them as the *evaluation target delays*. Hence, we have a set of target delays for each evaluation state as the ground truth. The logarithmic error a for the state $\boldsymbol{q}$ and target delay τ is given by

$$a(\boldsymbol{q}, \tau) = \left|\log(\mathbb{P}\left[Y > \tau \mid Q = \boldsymbol{q}\right]) - \log(\hat{p}_\theta\left(Y > \tau \mid Q = \boldsymbol{q}\right))\right|. \tag{7}$$

Let B denote the number of evaluation states and assume indexes j and l to denote evaluation states and evaluation target delay quantiles respectively. We calculate the error for every evaluation state and its target delays. By averaging it over the states, we obtain one error value for each evaluation quantile in the form

$$e_l = \frac{\sum_{j=1}^{B} a(\boldsymbol{q}_j, \tau_{j,l})}{B}. \tag{8}$$

It is evident that if the number of training samples is small compared to the number of samples needed for evaluation qunatiles, the predictor's performance vary depending on the distribution of the training samples. Therefore, in order to show these variations, we train 20 predictors with 20 different training datasets instead of one. Furthermore in the benchmarks, the average, minimum, and maximum of them per quantile is shown.

4.2 Results

In this section, we show the results of the evaluation schemes. These schemes are defined to compare the proposed method to the state-of-the art. Specifically, the novel EMM-based predictors are compared against the predictors only with GMMs as their parametric distribution.

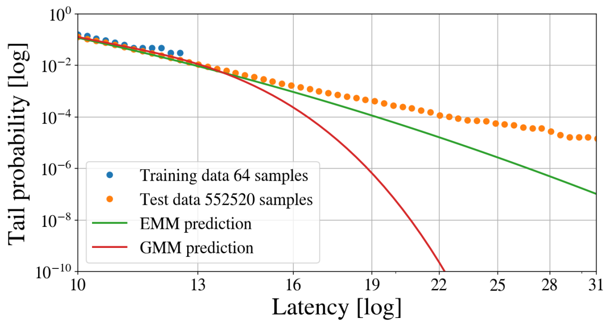
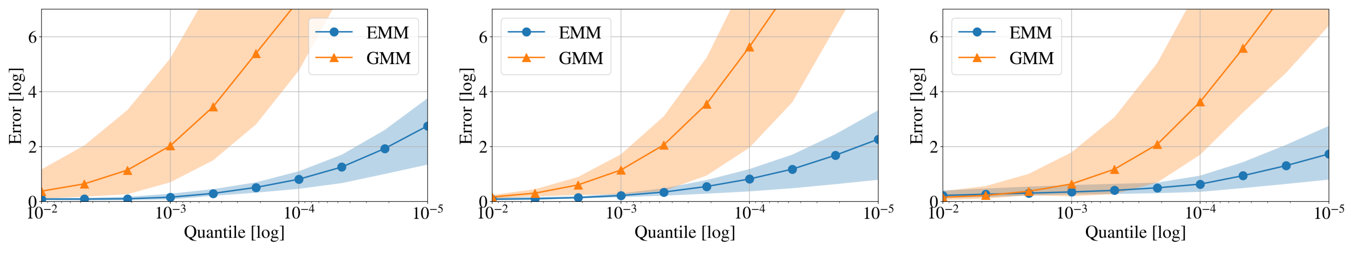


Figure 4: Sojourn time tail probability estimation of predictor-1 (three-hop) for the state $\boldsymbol{q} = \{1, 4, 2\}$

We start by sojourn time tail probability or DVP estimation of predictor-1 (three-hop) for the network state $\boldsymbol{q} = \{1, 4, 2\}$. The predictor is trained with 10k training samples which is the default for the rest of experiments. DVPs at the times that tasks enter the queues are estimated in 2 different predictors: EMM-based and GMM-based. Figure 4 depicts that the GMM error starts to grow where there is no training data (tail probability is less than 10^{-2}) and the predictions start to fall exponentially fast as expected. However, EMM predictions closely follow the ground truth until 10^{-4}. This shows the ability of EMM to exercise the limited training samples for the tail estimation.

Next, let us study the accuracy of predictors with the introduced evaluation metric e over all of the evaluation states as shown in Figure 5. We observe that the GMM error curves are higher than EMM in all cases and the difference increases for larger quantiles. However, it becomes less bold when the



(a) Predictor-1 (three-hop predictor) (b) Predictor-2 (two-hop predictor) (c) Predictor-3 (one-hop predictor)

Figure 5: DVP prediction errors for networks with different number of hops, all trained with 10k samples.

number of hops reduce. Since the first predictors (referred to as a three-hop predictor) have to deal with three dimensional features and a delay distribution with a heavier tail, they are less accurate compared to the one-hop predictors. It should be noted that there is improvement for both predictor types as we move from three-hop to one-hop, however, it is subtle for EMM.

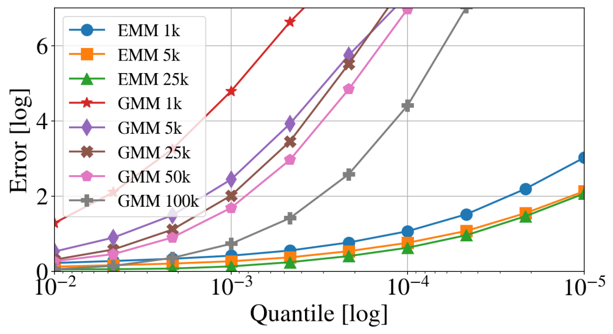


Figure 6: DVP prediction errors of three-hop predictors trained with different number of samples.

In this part, we study the impact of the number of training samples on the performance of predictors. By providing more training samples, the maximum likelihood algorithm finds more of the rare samples (outliers). The optimizer will try to cover them with the parametric distribution to get a lower loss value. Therefore, we expect less error at the tail region for the predictors that were trained with more samples. The error curves shown in Figure 6 approves our conclusion for the predictors of the same kind. However, EMM predictors' average errors are lower than GMM by orders of magnitude in the log scale. Despite the huge gap between EMM and GMM, the improvement of GMM predictors is significant when we provide more samples.

Finally, we investigate if the gap between our proposed EMM predictor could be vanished by elaborating the GMM. Using more Gaussian components (in other words, centers) in the mixture model distribution, should improve probability prediction accuracy at the tail. Because, the optimizer can move the spare Gaussian centers towards the tail and cover more area from that region. The downside is that the predictor will become more complex and it will need more samples and epochs for the training. Figure 7 shows 3 GMM

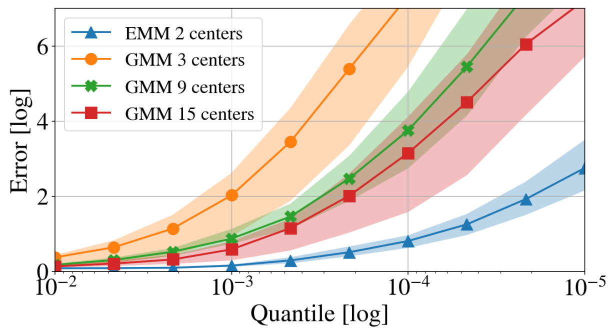


Figure 7: Average violation prediction errors of three-hup predictors equipped with different number of Gaussian components. GMM-based predictors with 9, and 15 centers use more complex neural networks with hidden layers of size 32, and 48 respectively.

predictors with different number of Gaussian components. As expected, more centers mean less average error for the GMM. However, it does not outperform EMM-based predictors that use GPD in addition to 2 Gaussian components.

5 CONCLUSIONS

We have studied the problem of predicting the transient DVP of latency-critical tasks traversing through a three-hop edge network modelled by a tandem queuing system. As the end-to-end delay guarantee levels put a threshold on the tail of the delay distribution, we introduced a novel DVP predictor that is not only accurate for the average delays, but also for the tail using extreme value theory. We have studied state-of-the-art GMM-based predictors and have demonstrated their poor performance for the tail delay predictions in the range 10^{-2}-10^{-5}. We fused an extreme mixture model with the state-of-the-art MDN method to deal with extreme events prediction. A key aspect of our proposed data-driven predictor is that its neural network explicitly learns the tail behaviour of the training data and can generalize that to the unseen network conditions. Therefore, as we observed in extensive simulations, it requires much less training samples compared to the commonly used GMM-based MDN method. Our approach is applicable to networks with general stationary service processes and does not assume i.i.d. service processes. We believe that key features of the proposed predictor make it useful in the design and optimization of wireless network infrastructures for latency-critical applications. Future work includes applying our approach to real networks and an in-depth study of DVP prediction in networks with non-stationary service processes, cross-traffic, and multi-server queues.

6 ACKNOWLEDGMENT

This research has been partially funded by the VINNOVA Competence Center for Trustworthy Edge Computing Systems and Applications (TECoSA) at KTH Royal Institute of Technology.

REFERENCES

[1] Bishop, C. Mixture density networks. Workingpaper, Aston University, 1994.

[2] Champati, J. P., Al-Zubaidy, H., and Gross, J. Transient analysis for multihop wireless networks under static routing. *IEEE/ACM Transactions on Networking 28*, 2 (2020), 722–735.

[3] Chen, H., Abbas, R., Cheng, P., Shirvanimoghaddam, M., Hardjawana, W., Bao, W., Li, Y., and Vucetic, B. Ultra-reliable low latency cellular networks: Use cases, challenges and approaches. *IEEE Communications Magazine 56*, 12 (2018), 119–125.

[4] Elbamby, M. S., Perfecto, C., Liu, C.-F., Park, J., Samarakoon, S., Chen, X., and Bennis, M. Wireless edge computing with latency and reliability guarantees. *Proceedings of the IEEE 107*, 8 (2019), 1717–1737.

[5] Flinta, C., Yan, W., and Johnsson, A. Predicting round-trip time distributions in IoT systems using histogram estimators. In *IEEE/IFIP Network Operations and Management Symposium (NOMS)* (2020), IEEE, pp. 1–9.

[6] Ibrahim, R., and Whitt, W. Real-time delay estimation based on delay history. *Manufacturing & Service Operations Management 11*, 3 (2009), 397–415.

[7] Raeis, M., Tizghadam, A., and Leon-Garcia, A. Predicting distributions of waiting times in customer service systems using mixture density networks. *International Conference on Network and Service Management (CNSM)* (2019), 1–6.

[8] Rothfuss, J., Ferreira, F., Walther, S., and Ulrich, M. Conditional density estimation with neural networks: Best practices and benchmarks. *arXiv:1903.00954* (2019).

[9] Samani, F. S., and Stadler, R. Predicting distributions of service metrics using neural networks. In *2018 14th International Conference on Network and Service Management (CNSM)* (2018), pp. 45–53.

[10] Scarrott, C., and MacDonald, A. A review of extreme value threshold estimation and uncertainty quantification. *REVSTAT–Statistical Journal 10*, 1 (2012), 33–60.

[11] Schulz, P., Matthe, M., Klessig, H., Simsek, M., Fettweis, G., Ansari, J., Ashraf, S. A., Almeroth, B., Voigt, J., Riedel, I., Puschmann, A., Mitschele-Thiel, A., Muller, M., Elste, T., and Windisch, M. Latency critical IoT applications in 5G: Perspective on the design of radio interface and network architecture. *IEEE Communications Magazine 55*, 2 (2017), 70–78.

[12] Senderovich, A., Weidlich, M., Gal, A., and Mandelbaum, A. Queue mining – predicting delays in service processes. In *Advanced Information Systems Engineering* (Cham, 2014), Springer International Publishing, pp. 42–57.