On Traffic Prediction in Backbone Networks for Adaptive Proactive Protection

Attila Dobai-Pataky, Balázs Vass, Lehel Csató

***Abstract*—In this paper, we revisit adaptive proactive protection schemes for backbone networks to guarantee high service availability. We aim to predict available – spare – link capacities and to use them as protection bandwidth to meet the service availability requirements of each connection. With a reasonable discretization of time, the advent of highly scalable capacity- and disaster risk-aware routing algorithms enables minimizing the lookahead at 2. In other words, we claim the spare bandwidth has to be predicted for only 2 time slots ahead. While minimizing the lookahead plays a key role in achieving better predictions, foreseeing the network bandwidth usage proves to remain a notoriously difficult task: traffic patterns are highly dynamic and are influenced by factors such as user behavior, daily, weekly, or yearly periodicity, and even unforeseeable external events. After discussing the protection scheme design, this study compares the performance of two time series forecasting methods for the prediction of network traffic in the short term: Autoregressive Integrated Moving Average (ARIMA) models and Temporal Convolutional Networks (TCN). These models are assessed on real traffic data obtained from the Energy Sciences Network (ESnet), a high-speed, international backbone network.**

I. Introduction

In today's cloud era, communication networks are topmost critical infrastructures [1]. The new mission-critical applications, such as telesurgery or stock market prediction, clearly demand high Quality of Service (QoS) of the underlying network infrastructure [2]. Today's networks are operated with approximately four-nines availability, that is, the services are available at least 99.99% of the time, which translates to a maximum of around 53 minutes downtime a year. Four-nines is often considered a reasonable cost-benefit ratio, however, the lack of highly reliable Internet has become a limiting factor for development. Current technology trends point toward the advent of the era of the Internet of Everything (IoE) with an enormous amount of data [3], where e.g., augmented and virtual reality applications not only require high dependability, but are also extremely latency-sensitive. Thus, QoS enhancement to reach and surpass the requirements of ultra-reliable and low-latency communication (URLLC) is a must, where six-nines is a typical required availability [4].

There are two fundamentally distinct strategies to enhance connectivity in backbone networks. *Proactive* methods aim to ready the system for potential failures so that when a failure happens, no internal reconfiguration of the network is necessary. This is accomplished by distributing the same information across multiple routes so that in the event of a failure, only the end node needs to respond. Conversely, *reactive* methods involve altering the network configuration once a failure has been detected. Note that while reactive strategies can conserve a considerable amount of bandwidth compared to proactive ones, they require immediate network reconfiguration after a failure occurs, which can be relatively slow in practice. This makes reactive approaches unsuitable for protecting latency-sensitive data flows. This leaves us with proactive protection schemes. Among these, for achieving the prescribed availabilities, the current best practice is using solutions offering *dedicated protection*, due to their simplicity, robustness, and flexibility.

Attila Dobai-Pataky, Balázs Vass, and Lehel Csató are with the Babeș-Bolyai University of Cluj Napoca, Romania. Balázs Vass is also with the Budapest University of Technology and Economics (BME). Email: attila.dobai@stud.ubbcluj.ro, balazs.vass@ubbcluj.ro, lehel.csato@ubbcluj.ro. Supported by the project "Romanian Hub for Artificial Intelligence - HRIA", Smart Growth, Digitization and Financial Instruments Program, 2021-2027, MySMIS no. 334906.

Co-funded by the European Union

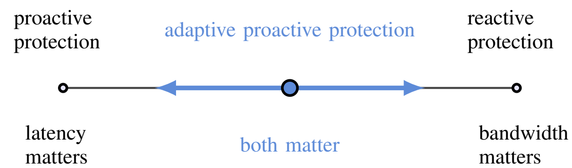


Fig. 1. Proactive protection schemes waste bandwidths, while reactive protection schemes are too slow for latency-sensitive applications. An ideal protection scheme re-introduces adaptiveness to a proactive approach by letting it prepare (react) to *predicted* bandwidth usages instead of peak rates.

Being stuck with a single static dedicated protection scheme, however, would waste a significant amount of bandwidth. Also, it could cause unnecessary bandwidth restrictions on the working paths (WPs). This is because the capacity of the WP is often over-provisioned for peak rates; however, a significant share of the bandwidths is unused most of the time [5]. To put this issue in a slightly different way, the drawback of using a single static dedicated protection scheme is the lack of *adaptability*. We argue that an ideal protection scheme would re-introduce adaptiveness to a proactive approach by letting it prepare (react) to *predicted* bandwidth usages instead of peak rates (see Fig. 1). Here the idea is that the proactive scheme could utilize the predicted spare bandwidth. The drawback of the approach is that if the bandwidth used by the actual traffic demands exceeds the predictions, some (minor) transient bandwidth limitations should be imposed.[1]

[1]Transient bandwidth limitations may appear when applying reactive schemes too. Reactive schemes inevitably lose some packets when failure hits. In the case of ESnet, e.g., the performance impact of losing only .0046% of the packets can cause the throughput to become almost 17 times smaller compared to the loss-free state [6].

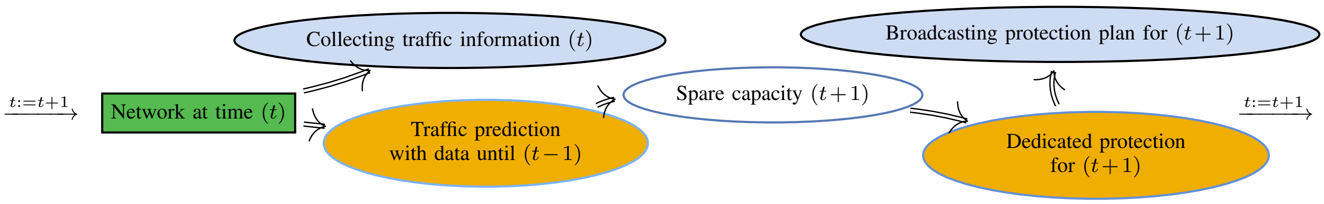


Fig. 2. Scheme of an adaptive proactive protection approach. Time slots could be as short as a few seconds [7].

This issue should be seen as a trade-off between adaptability to changing traffic patterns and occasional bandwidth limitations. No adaptivity, with a basic ability of the proactive protection to enhance availability, and no bandwidth limitations should be considered as the baseline. More opportunistic adaptive schemes utilizing traffic predictions would achieve a higher bandwidth utilization of protection via trading a(n ideally) small timeshare with minor bandwidth limitations. Note that here, the experienced performance of the traffic prediction used plays a key role.

Summarized, the envisioned adaptive proactive protection schemes have two main components:

1) *Prediction:* forecasting the spare bandwidth in an upcoming time period, and
2) *Routing:* computing a best-fit dedicated protection for the period based on the estimated spare bandwidth.

The current study aims to understand the key possibilities and limitations of such approaches by investigating their two key components. To do so, we reiterate on recent article [7], which, to the best of our knowledge, represents the current state-of-the-art among adaptive proactive protection schemes.

Our main contributions are as follows:

- Analyzing the components of an ideal adaptive proactive protection framework, we propose refinements to the state-of-art model [7]. Namely, we argue that with practical discretization of time, a lookahead of as low as 2 can be applied for prediction (instead of the baseline value of 8). We also hint at alternative scalable routing algorithms to be used to compute the dedicated protection routes.
- We evaluate the performance of multiple traffic prediction approaches on real-world data (collected from the ESNet), based on prediction accuracy and resources used. With a lookahead of 2, our best-performing predictions (both among TCNs and ARIMAs) achieve lower average RMSEs than the champion of [7], when traffic increase factored in.
- Concerning a slightly different metric, TCNs are proven to be effective in reducing the *average relative* RMSE of the links.

The rest of the paper is structured as follows: §II presents our model, and background in both prediction and routing. §III presents our test data and methodology for predicting bandwidth demand, §IV discusses our evaluation results, and finally, §V concludes our work and envisions future research directions.

II. Model, Assumptions, and Background

The network is modeled as a directed graph $D=(V,A)$, with each link $a\in A$ having capacity $c_a>0$, and a related series X_t^a $(t\in\mathbb{N})$ describing the bandwidth usage on a at time t. If known in advance, spare bandwidths $Y_t^a:=c_a-X_t^a$ could be used to host backup paths for protected data flows, computed by highly scalable resilient routing algorithms. Unfortunately, future demands are not known. Thus, we create a predicted series of the (slightly underestimated) spare bandwidth $\overline{Y}_t^a:=c_a-\alpha\hat{X}_t^a$, where $\hat{X}_t^a$ is the predicted bandwidth used, and $\alpha\geq 1$ is a parameter close to 1. Here, two issues can arise: 1) Too much underestimation leads to less capacity to be used as a backup resource, and 2) an eventual overestimation may yield bandwidth limitations. We argue $\hat{X}_t^a$ can be predicted when X_{t-2}^a becomes known, as time interval $t-1$ (lasting 30 `[sec]` in our use case) is sufficient to perform all necessary preparations (prediction, routing, updating). This is in contrast to [7], which supposes a lookahead of 8 is necessary, making it a more challenging task.

Hereafter, we will call the setting when $\hat{X}_t^a$ is predicted based on $X_{t-2}^a, X_{t-3}^a, \ldots$ as *short prediction*.

A. Dedicated Protection Approaches

Nowadays, the so-called 1+1 is the most widespread dedicated protection approach. With 1+1, the data can be sent parallel on edge- or node-disjoint working and backup paths (WP and BP), ensuring instantaneous recovery against single link or node failures in a simple manner. Such a path pair can be quickly found, via, e.g., Suurballe's algorithm [8] that computes a WP-BP path pair of minimum total length in $O(|E|+|V|\log|V|)$. Being such a computationally efficient subroutine, Suurballe's algorithm is very useful for preparing the network for single element failures with short predictions.

To exceed a certain availability threshold, however, protection against the simultaneous failure of multiple elements becomes inevitable. Such failures can be caused by natural or man-made disasters (such as earthquakes, electromagnetic pulses, etc.). Such failures are modeled as *Shared Risk Group*s (SRG), which are sets of network elements that are expected to fail simultaneously. For routing purposes, the failure of a node v can be modeled as the failure of all the links incident to v. Thus, it is enough to take into count Shared Risk *Link* Groups (SRLGs), which are simply just SRGs containing only links. To tackle the issue of SRLG failures, the framework of [7] used the GDP-R routing of [9]. Unfortunately, since the problem formulation of [9] is NP-hard, the routing of GDP-R is calculated via the help of an ILP formulation - that can take an infeasibly long time to be optimized, especially if we want to make it work together with short predictions.

Studying SRLG-disjoint path computing problems has a long history [10]. Without going into much detail, [11]–[13] prove the NP-hardness of the problem in various settings. [14]–[16] rely at least partly on ILP/MILP formulations to solve or approximate the weighted version of the SRLG-disjoint paths problem. Under a probabilistic SRLG model, [17] aims finding diverse routes with minimum joint failure probability via an integer non-linear program (INLP). Heuristics were also investigated [18], [19], unfortunately, with issues like possibly non-polynomial runtime or occasional forwarding loops in the presence of disasters.

On the positive side, resulting from the chain of studies [20]–[28], the current state-of-art algorithm published in [29], [30] called *DateLine* is highly scalable (with a near-linear computational complexity in function of $|V|$ in practice), and has a low worst-case time complexity for solving the SRLG-disjoint paths problem supposing the topology is planar and the paths should be node-disjoint. While the DateLine algorithm on its own does not take into account the spare bandwidths of links, being computationally efficient, it can be used for preparing the network for SRLG failures with short predictions. In this sense, it can be seen as the best known counterpart of Suurballe's algorithm for SRLG failures.

In conclusion, we believe the current state of SRLG-disjoint routing algorithms allows replacing the ILP-based GDP-R routing algorithm used in [7] to more scalable efficient algorithms, enabling to choose a lookahead time of as low as 2 for prediction for the network traffic. Note that a lookahead of 1 would not be enough, since then the network should hop to a newly computed protection state in an infinitesimally small time interval. As we will see, a lookahead of 2 is a crucial enabler of more accurate predictions yielding a better overall performance of the protection framework.

B. Traffic Prediction Techniques

There are multiple classical and novel models available for the purpose of time series forecasting. A classical, linear model is the Autoregressive Integrated Moving Average (ARIMA) [31]. The ARIMA is interpretable due to its linear nature and simple structure; it performs well on periodic, linearly dependent data. Given the noisiness and nonlinear behavior of real-world bandwidth usage data, nonlinear models, especially neural networks, are also applied in this domain. Examples include feed-forward neural networks (more specifically, Convolutional Neural Networks – CNNs), such as the

- Temporal Convolutional Networks (TCNs) [32], devised for general-purpose time series forecasting, employing residual connections known from ResNet [33], and
- WaveNet [34], an architecture similar to TCNs, originally used for voice generation, but applicable to other time series prediction problems [35]. Wavenet uses *local* and *global conditioning*, based on metadata provided as input (for example, linguistic features or the speaker's identity). In the case of traffic volume forecasting, such metadata could be the time-of-day or day-of-week.

Recurrent Neural Networks (RNNs) are another family of neural networks suited for processing sequential data [36]. Although RNNs are structurally designed for processing sequential data, they are generally harder to train. (The sequential computation of data inherent in RNNs limits parallelization during training, as opposed to CNNs, which may process a long input sequence as a whole.) Furthermore, TCNs were shown to outperform RNNs (including specialized architectures like LSTMs and GRUs) on multiple sequential tasks [32].

A third, novel family of neural networks, namely Transformer-based architectures have outstanding performance in the domains of natural language processing and speech recognition, yet they are harder to apply to time series forecasting problems, due to their permutation-invariant self-attention mechanism. Study [37] shows that, in many cases, Transformers specialized for time series forecasting have poorer performance than a simple linear model, on various datasets. The original Transformer architecture also has $O(N^2)$ time and memory complexity in terms of input sequence length N, which may be prohibitively expensive in the case of high-volume time series data (whereas the TCN scales linearly). Although some specialized Transformer models reduce this theoretical complexity to $O(N \log N)$, or even $O(N)$, [37] also shows that it is unclear whether this improves the actual inference time and memory cost. A more recent architecture, iTransformer [38], achieves superior performance on multi-variate time series prediction tasks compared to the previous models assessed in [37].

Despite the recent emphasis on deep learning models, new linear models are still being developed; Random Convolutional Kernel Transform (ROCKET) [39], for example, is essentially a convolutional neural network with no hidden layers and a very high number of filters. ROCKET was demonstrated to be on par with deep learning architectures like ResNet in terms of accuracy on certain time series benchmarks.

For our traffic prediction purposes, we assessed a classical linear model, the ARIMA, thanks to its interpretability, and a nonlinear model, the TCN, due to the chaotic nature and high volume of the examined bandwidth usage data.

III. Methodology for Traffic Prediction

A. Test Data

We analyzed bandwidth usage data obtained from the Energy Sciences Network (ESnet), covering the 3-month period of 2024-08-01 to 2024-11-30, marked by the vertical lines in Fig. 3. The resolution of the time steps is 30 secs; this gives approximately 265000 data points per direction per link. The tests were executed on the 141 data links that reported traffic in at least 10% of this time interval. We have analyzed a single data flow direction for each link. (In the context of traffic volume prediction, we may treat each direction as a separate link since we have per-direction bandwidth usage data on them.) The first 78 days of the analyzed interval were used

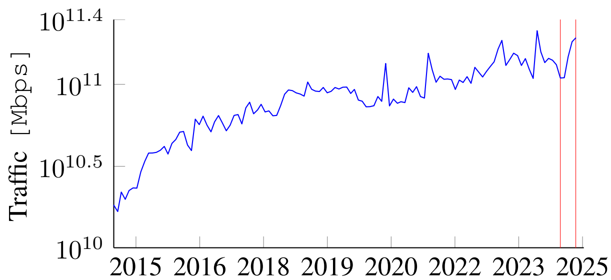


Fig. 3. ESnet average monthly total bandwidth volume over time. The interval between the vertical lines stands for '24-08-01 – '24-11-30.

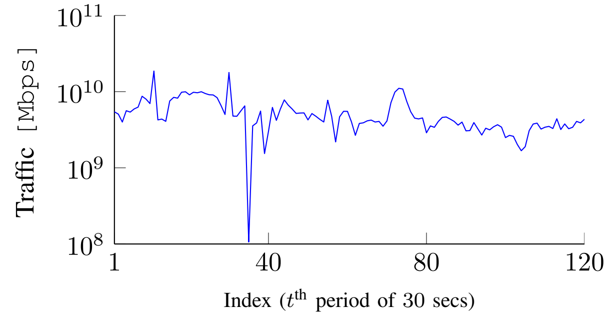


Fig. 4. Traffic on link LASV–LOSA, 2024-08-01 00:00:00-01:00:00

as training data (relevant in the case of TCNs), the last 14 days were used to evaluate the prediction capabilities of the models. This results in a 85%-15% train-test split ratio.

Our goals are to predict the bandwidth usage:

- 4 minutes (8 units) into the future – in order to compare the results with benchmark [7]
- 1 minute (2 units) into the future – since scalable resilient routing algorithms make it possible to determine the routing and distribute the results in such a short timeframe.

For the sake of simplicity and computational efficiency, we make no assumptions about the correlation between traffic volumes on links, meaning all links are treated as separate univariate time series prediction problems. Although each model evaluation happens with the same set of hyperparameters across links, model fitting / training happens separately for each link, since traffic volume, potential periodicity, and other patterns vary widely between links. Fig. 4 shows a one-hour traffic example on a link, with spikes in traffic volume (where the scale is logarithmic), most notably the sudden drop and jump-back of more than an order of magnitude at time step 35. Such spikes make the prediction challenging.

B. *Autoregressive Integrated Moving Average models (ARIMA)*

For a time series X_t $(t \in \mathbb{N})$, ARIMA models predict the differenced series according to equation

$$X'_t = c + \sum_{i=1}^{p} \phi_i X'_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} + \epsilon_t,$$

where ϵ is white noise and the hyperparameters p,d,q are the order of the autoregressive component, the order of differencing, and the order of the moving average component, respectively. Optionally, c denotes a linear trend [31]. To determine the optimal parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ for a given set of hyperparameters in timestep t, we fit the model using the maximum likelihood estimation on data taken from a sliding window $X_{t-l},\ldots,X_t$, with length l. Given the forecast distance f, prediction $\hat{X}_{t+f}$ is the expected value of the f^{th} element in the series generated. Hence, each ARIMA experiment is determined by the choice of hyperparameters (p,d,q), l, and f.

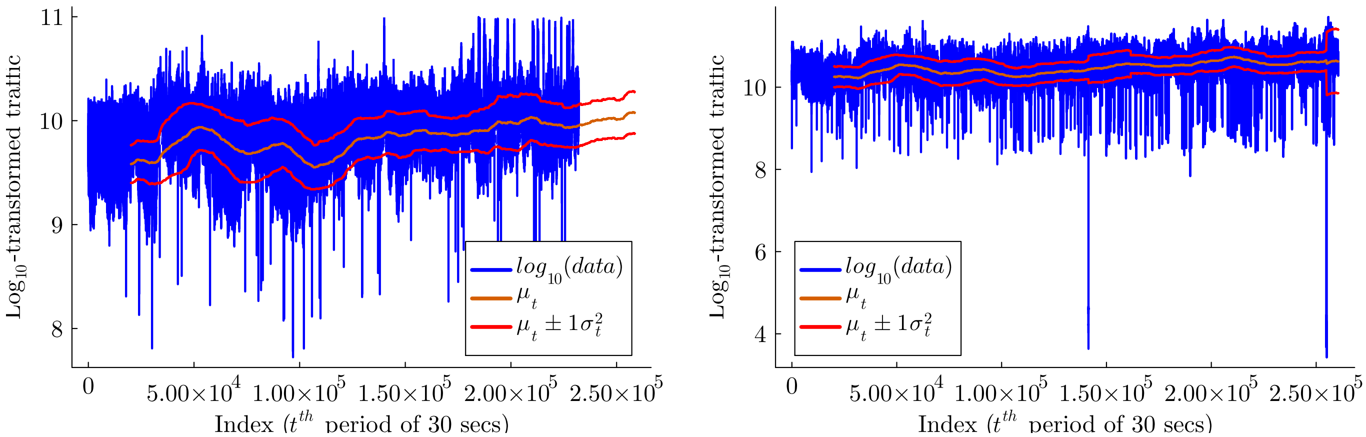
Note that new parameters ($\boldsymbol{\phi}$ and $\boldsymbol{\theta}$) are estimated for each timestep. This is beneficial because the analyzed series are not truly stationary. The optimization is performed using the BFGS method. For the first prediction on a given link, the starting values for the optimization are chosen using the conditional sum-of-squares (CSS) method. Two approaches were used to choose the optimization's starting values for subsequent timesteps: 1) utilizing the CSS method again, and 2) using the values that the previous optimization converged to. The first approach yields a slightly better performance, but has a higher execution time.

C. *Temporal Convolutional Networks (TCN)*

TCNs are convolutional neural networks specifically designed for sequential data processing. Their core component is the *causal convolution*, where the output corresponding to timestep $t+f$ depends strictly on inputs from t and earlier, preventing future information leakage. TCNs employ *residual connections* [33] to mitigate the problem of vanishing gradients. Causal convolutions are *dilated*, in order to expand the receptieve field: the convolutional kernels are applied to input elements spaced d positions apart, where d is the dilation rate. By choosing the dilation rates to be increasing powers of two in consecutive layers, an exponential growth in the size of the receptive field of the model is achieved with a linear increase in the number of parameters [32], [34]. Dropout layers are employed in order to prevent overfitting [32]. Another method applicable to reduce overfitting is the L2 regularization of weights, as used in the case of CNNs for time series forecasting in [35]. In order to detect overfitting, 10% of the training dataset was randomly selected to constitute the validation set.

In our case, TCN instances operate on a much longer range than ARIMA instances: being trained and validated on the first 85% of the data, they use the same parameters during the evaluation on the entire test set. As a consequence, an additional standardization step is required, in order to be able to treat the entire data as a sample from a single distribution. Note that while this standardization proved critical for the performance of TCNs, it's application to ARIMAs showed no improvements in performance.

We observed that, on most links, the marginal distribution of the data on the entire time interval is approximately log-normal, the most significant outliers being the 0-values, comprising 0.77% of the data per link on average. We replaced these with a moving average, in order to be able to



(a) Log-transformed traffic volumes on link CERN513–WASH.

(b) Log-transformed traffic volumes on link NEWY32AOA–WASH.

Fig. 5. Log-transformed traffic volumes on two different links; mean μ_t, and standard deviation σ_t^2 calculated with window size $w = 20160$, i.e., one week.

normalize the data. This leads to more inaccurate predictions corresponding to 0-values, but better performance in general. Occasional outages, although rare, were also represented with 0-values in the dataset.

Furthermore, any sufficiently long interval taken from a link's data also has an approximately log-normal marginal distribution. This, however, does not mean that the data is stationary: the means and variances of the log-transformed data vary significantly in function of the starting point and size of the interval (most notably, the means show a rising trend; for an example, see Fig. 5a.)

These distribution properties are crucial for a meaningful standardization of the data. For each timestep t, X_t (and, just as importantly, $\hat{X}_{t+f}$) is assumed to come from the distribution $\mathcal{N}(\mu_t, \sigma_t^2)$, where μ_t and σ_t^2 are calculated from the running window $X_{t-w}, \ldots, X_t$. The running window length w is considered a hyperparameter of our model. An additional clipping step is performed on the model inputs to lessen the disproportionate influence of outliers on the training process. Here, in contrast to non-sequential supervised learning tasks, outliers could not be simply omitted, as the temporal structure of the data would be compromised in the process.

It is the intuitive option to treat the prediction problem as a regression task, and choose the output of the model to be a single $\hat{X}_{t+f}$ value. To optimize the model parameters, a possible loss function to be minimised is the root mean squared error (RMSE). But, as evidenced in [40], and also applied in [34], the problem can also be treated as a classification task, even though the data values are continuous. In this case, the continuous range of possible values is split evenly into a number of classes. The output of the model is then a categorical probability distribution over these classes, obtained via the softmax function. The loss function in this case is the cross-entropy between the predicted and observed distributions for X_{t+f} [36]. The advantage of such a model is that it can better approximate values from multimodal distributions, whereas regression works best on unimodals. As presented by §IV, regression-based and classification models had no significant differences in performance in our particular case.

IV. Evaluation

For conducting our evaluations, we implemented the prediction module in the Julia programming language and made it available on GitLab.[2] The ARIMA tests were executed parallelly on a machine with an Intel Core(TM) i7-3770K CPU @ 3.50GHz and 32 GB RAM. The TCN tests were executed on an Nvidia GeForce RTX 3070 GPU with 8 GB RAM.

[2]Implementation available: https://gitlab.com/dobaipatakyattila/qoserm-tsa

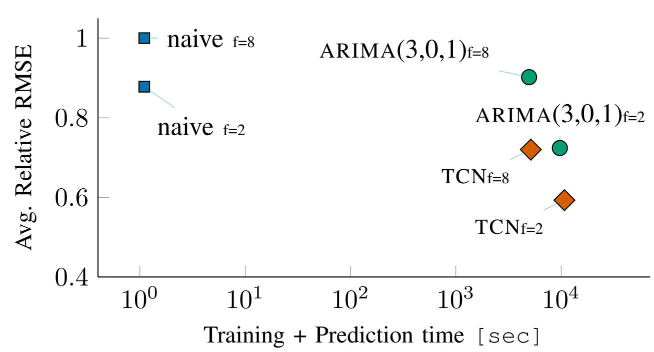


Fig. 6. Average of per-link relative model losses vs absolute runtimes

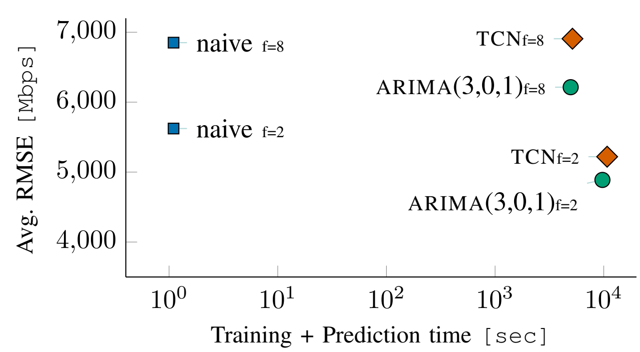


Fig. 7. Average of absolute model losses vs runtimes

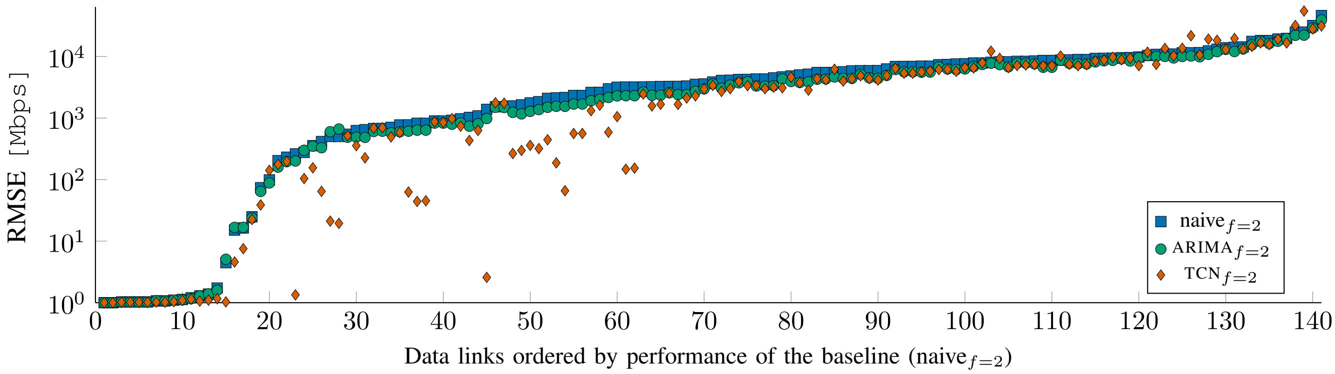


Fig. 8. Model performance evaluation on a per-link basis, with a forecast distance of 2 units.

The baseline for our forecasting accuracy are the naïve lag-based models $\hat{X}_{t+f} = X_t$, for $f = 2$ and $f = 8$, having the average RMSEs of 5623.62 and 6851.56 MBPS per link, respectively. Training+prediction times and the averages of per-link relative losses compared to the lag-based model with $f = 8$ are depicted in Fig. 6. Taking the average loss reduction on the links, the best evaluated ARIMA model for $f = 8$ decreased loss by 10%, while the TCN achieved a 39% decrease. As expected, reducing f to 2 proved highly beneficial: the simple lag-based model's loss decreased by 13%, while the ARIMA and the TCN achieved 38% and 68% lower losses on average than the $f = 8$ lag-based model, respectively.

The best-performing ARIMA model inspected with the parameters $(3,0,1)$ and sliding window size 1200 predicted with an average RMSE of 4888.51 MBPS in a distance of 2 units, and 6214.62 MBPS in a distance of 8. On the same network, for the year 2019, [7] reported an RMSE of 4451 MBPS in a distance of 8, with their best-among-arimas ARIMA$(3,1,2)$. These values may be compared when we factor in the 97% traffic volume increase experienced by the network in the last 5 years (based on the average monthly traffic of 2019 and the average monthly traffic for 2024–11 reported by ESnet). The growth-adjusted RMSE value for [7] would be $4451 \cdot 1.97 = 8768.47$, although this simplistic calculation can not take into account the possible change of patterns, variance, or network expansion (the addition of new links).

Turning to TCNs, the best-performing TCN model inspected consisted of 4 convolutional layers, each with 40 channels and size 3 kernels. Increasing the number of convolutional layers beyond a certain threshold did not improve performance. After every second convolutional layer, 20% dropout layers were used. Residual connections bridged every two layers. A dilation factor of 2 was used. The best lookback window (the size of the input of the TCN) was 3000 timesteps long (corresponding to 25 hours), in order to capture daily periodicities, a compromise between model performance and memory size and training time. A small sized fully connected layer (50 parameters, swish activation function) between the layer collecting the convolutional outputs and the final output layer significantly improved model performance. With respect to the last layer, we evaluated single output (regression), and 20, 64 output (classification) models. Classification models had close to identical performance to regression models, with a minor degradation attributable to quantization. Notably, classification models exhibited a higher tendency to overfit, demonstrating their capacity to approximate arbitrary data distributions. The running window length w, used in data standardization, was fixed at 20160 timesteps (equivalent to 1 week). Input values outside the $\mu_t \pm 3\sigma_t^2$ limit were considered outliers and clipped. The hyperparameter w could be further tuned on a per-link basis: preliminary experiments showed that on certain links chosing w to be 40000 brought up to 90% performance improvements. We chose not to factor in these improvements in the final evaluation, in order to adhere to our primary objective of using the same hyperparameters across all links for the assessment of a given model.

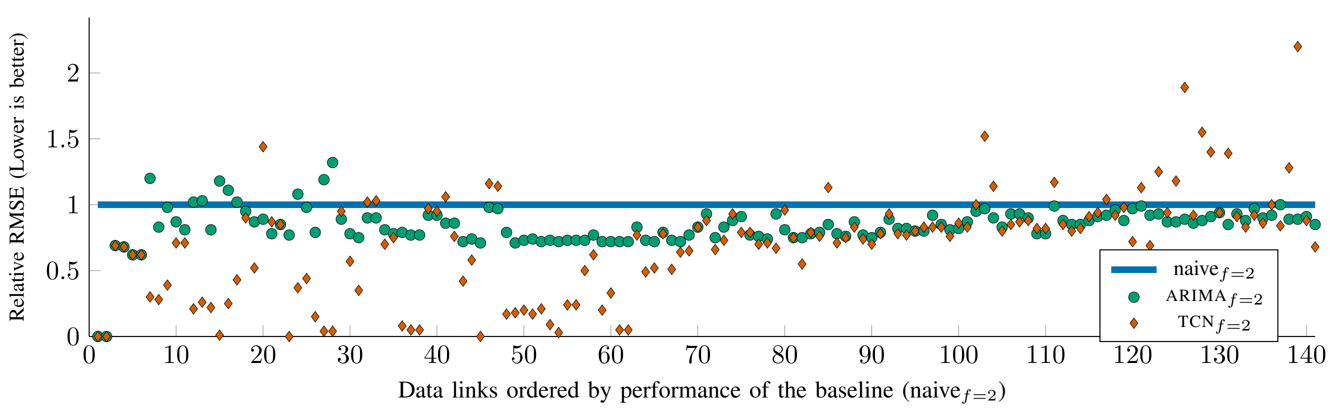


Fig. 9. Model performance compared to the baseline on a per-link basis, with a forecast distance of 2 units.

For forecast distance $f = 2$, this model produced an average RMSE of 5220.72 MBPS. Even though this single average value is worse than that of the ARIMA model, the TCN outperformed the ARIMA in the majority of cases (93 out of the 141). The worse average RMSE is due to severe failure of the TCN in the few extreme cases where the behavior of the test data changes suddenly and differs enormously from the train data. (As an example, see Fig. 5b. Note the sudden increase in variance after $t = 2.5 \times 10^5$). In such cases, the ARIMA performs better, due to it's shorter lookback window and independence of values past that window. Inspecting the per-link loss decrease however, TCNs have a 22% lower loss than ARIMAs on average. See Fig. 9 for a per-link evaluation of model performances relative to the baseline and Fig. 8 for the same evaluation in absolute terms. For forecast distance $f = 8$, the TCN had an average RMSE of 6907.83 MBPS. The best-performing nonlinear model in [7] had a RMSE of cca. 2800 MBPS (the simple growth-adjusted value would be cca. 5516 MBPS). Our TCN models, for $f = 8$, fall short of this value, primarily due to the previously mentioned cases where extreme shifts occured in test data behavior. Such shifts greatly influence the RMSE value due to the relatively short test dataset (compared to [7]).

V. Conclusions and Future Work

Based on our simulation results, we conclude that minimizing the prediction distance should be a design priority for the family of adaptive protection schemes described in the paper. The performance of our TCN-based model could be further improved by adaptively selecting the optimal window length for input standardization, as using a fixed global value w appears to be a key cause of bad predictions in extreme cases. A potential solution is locally conditioning the model on multiple transformed versions of the time series, standardized with different w values. The model's generalization capabilities could also be improved by training on a longer time interval. With this said, as part of our future work, we will also try to achieve better prediction performance by leveraging the possible correlations between data links, arising from the graph structure of the data. Concerning the big picture, our plans also include implementing a full-blown adaptive proactive protection scheme.

References

[1] A Review of Critical Infrastructure Domains in Europe. SPEAR, www.spear2020.eu/News/Details?id=120. Accessed: 2025.

[2] Z. Wang, *Internet QoS: Architectures and Mechanisms for Quality of Service*. San Francisco, CA, USA: Morgan Kaufmann Publ. Inc., 2001.

[3] F. Tariq *et al.*, "A speculative study on 6g," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 118–125, 2020.

[4] D. Feng *et al.*, "Ultra-reliable and low-latency communications: applications, opportunities and challenges," *Science China Inf. Sci.*, 2021.

[5] Y. Huang and R. Guerin, "Does over-provisioning become more or less efficient as networks grow larger?" in *13TH IEEE International Conference on Network Protocols (ICNP'05)*, 2005, pp. 11 pp.–235.

[6] On packet loss in ESNet. fasterdata.es.net/network-tuning/tcp-issues-explained/packet-loss/. Accessed: 2025.

[7] F. Mogyorósi *et al.*, "Adaptive protection of scientific backbone networks using machine learning," *IEEE TNSM*, vol. 18, no. 1, 2021.

[8] J. W. Suurballe and R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, no. 2, pp. 325–336, 1984.

[9] P. Babarczi *et al.*, "Instantaneous recovery of unicast connections in transport networks: Routing versus coding," *Computer Networks*, vol. 82, pp. 68–80, 2015.

[10] P. Datta *et al.*, "Sub-graph routing : A novel fault-tolerant architecture for shared-risk link group failures in wdm optical networks," in *Design of Reliable Communication Networks (DRCN)*, 2003, pp. 296–303.

[11] J.-Q. Hu, "Diverse routing in optical mesh networks," *IEEE Trans. Communications*, vol. 51, pp. 489–494, 2003.

[12] J.-C. Bermond *et al.*, "SRLG-diverse routing with the star property," in *Design of Reliable Communication Networks (DRCN)*. IEEE, 2013.

[13] X. Luo and B. Wang, "Diverse routing in WDM optical networks with shared risk link group (SLRG) failures," in *IEEE DRCN*, 2005.

[14] D. Xu *et al.*, "SRLG-diverse routing of multiple circuits in a heterogeneous optical transport network," in *IEEE DRCN*, 2011.

[15] T. Gomes *et al.*, "Two heuristics for calculating a shared risk link group disjoint set of paths of min-sum cost," *Journal of Network and Systems Management*, vol. 23, no. 4, pp. 1067–1103, 2015.

[16] A. de Sousa *et al.*, "Determination of the minimum cost pair of *D*-geodiverse paths," in *IEEE DRCN*, Munich, Germany, March 8-10 2017.

[17] H.-W. Lee *et al.*, "Diverse routing in networks with probabilistic failures," *IEEE/ACM Trans. Netw.*, vol. 18, no. 6, pp. 1895–1907, 2010.

[18] K. Xie *et al.*, "Divide and conquer for fast SRLG disjoint routing," in *IEEE/IFIP DSN*, 2018, pp. 622–633.

[19] B. Yang *et al.*, "Keep forwarding: Towards k-link failure resilient routing," in *IEEE INFOCOM*. IEEE, 2014, pp. 1617–1625.

[20] D. Bienstock, "Some generalized max-flow min-cut problems in the plane," *Mathematics of Operations Research*, vol. 16, no. 2, 1991.

[21] C. Mcdiarmid *et al.*, "Induced circuits in planar graphs," *Journal of Combinatorial Theory, Series B*, vol. 60, no. 2, pp. 169 – 176, 1994.

[22] C. McDiarmid *et al.*, "Non-interfering network flows," in *Algorithm Theory—SWAT'92: Third Scandinavian Workshop on Algorithm Theory Helsinki, Finland, July 8–10, 1992 Proceedings 3*. Springer, 1992.

[23] S. Neumayer *et al.*, "Assessing the vulnerability of the fiber infrastructure to disasters," *IEEE/ACM Trans. Netw.*, pp. 1610–1623, 2011.

[24] ——, "Geographic max-flow and min-cut under a circular disk failure model," in *IEEE INFOCOM*, 2012, pp. 2736–2740.

[25] Y. Kobayashi and K. Otsuki, "Max-flow min-cut theorem and faster algorithms in a circular disk failure model," in *IEEE INFOCOM*, 2014.

[26] K. Otsuki *et al.*, "Improved max-flow min-cut algorithms in a circular disk failure model with application to a road network," *European Journal of Operational Research*, vol. 248, no. 2, pp. 396–403, 2016.

[27] B. Vass *et al.*, "Polynomial-time algorithm for the regional SRLG-disjoint paths problem," in *Proc. IEEE INFOCOM*, May 2022.

[28] ——, "A whirling dervish: Polynomial-time algorithm for the regional srlg-disjoint paths problem," *IEEE/ACM Trans. on Netw.*, 2023.

[29] E. Bérczi-Kovács *et al.*, "Efficient algorithm for Region-Disjoint survivable routing in backbone networks," in *IEEE INFOCOM*, 2024.

[30] ——, "DateLine: Efficient Algorithm for Computing Region Disjoint Paths in Backbone Networks," *IEEE JSAC*, pp. 1–14, Jan. 2025.

[31] R. Hyndman, *Forecasting: principles and practice*. OTexts, 2018.

[32] S. Bai *et al.*, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," arXiv, 2018.

[33] K. He *et al.*, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[34] A. van den Oord *et al.*, "Wavenet: A generative model for raw audio," in *Arxiv*, 2016.

[35] A. Borovykh *et al.*, "Conditional time series forecasting with convolutional neural networks," arXiv, 2018.

[36] I. Goodfellow *et al.*, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[37] A. Zeng *et al.*, "Are transformers effective for time series forecasting?" in *AAAI'23/IAAI'23/EAAI'23*. AAAI Press, 2023.

[38] Y. Liu *et al.*, "itransformer: Inverted transformers are effective for time series forecasting," in *The Twelfth International Conference on Learning Representations*, 2024.

[39] A. Dempster *et al.*, "Rocket: exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, p. 1454–1495, Jul. 2020.

[40] A. Van Den Oord *et al.*, "Pixel recurrent neural networks," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016.