Adaptive Protection of Scientific Backbone Networks Using Machine Learning

Ferenc Mogyorósi, *Member, IEEE*, Alija Pašić, Richard Cziva, Péter Revisnyei, Zsolt Kenesi, and János Tapolcai

***Abstract*—In this article, we propose a new protection scheme for backbone networks to guarantee high service availability. The presented scheme does not require any reconfiguration immediately after the failure (i.e., it is proactive). At the same time, it does not require any reserved backup network resources either. To achieve these seemingly contradictory goals, we utilize the recent advancements in Machine Learning (ML) to implement a network intelligence that periodically re-allocates the unused capacity as protection bandwidth to meet the service availability requirements of each connection. Our goal is achieved by two components (1) predicting the traffic for the next period on each link, and (2) intelligently selecting the best fit dedicated protection scheme for the next period depending on the estimated unused (spare) bandwidth and the previous service availability violations. Note that re-allocating protection bandwidth affects neither the operational connections nor the current best practice of operators to over-provision network bandwidth to support elephant flows. Finally, we provide a case study on the real traffic from Energy Sciences Network (ESnet), a high-speed, international scientific backbone network. The key benefit of our framework is that adaptively utilizing the over-provisioned bandwidth for spare capacity is sufficient to improve the availability from three-nines to five-nines (in ESnet for the 30 examined connections). The drawback is negligible bandwidth limitations; the user perceives a minor and very temporal bandwidth limitation in less than 0.1% of the time.**

***Index Terms*—Quality of Service (QoS), availability, traffic prediction, energy sciences network, service level agreement (SLA), deep learning.**

Manuscript received May 1, 2020; revised September 21, 2020 and December 21, 2020; accepted December 23, 2020. Date of publication January 11, 2021; date of current version March 11, 2021. The research leading to these results was partially supported by the High Speed Networks Laboratory (HSNLab). Projects no. 123957, 129589, 124171, 134604 and 128062 have been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the FK_17, KH_18, K_17, FK_20 and K_18 funding schemes respectively. The research reported in this paper and carried out at the BME was supported by the "TKP2020, Institutional Excellence Program" of the National Research Development and Innovation Office in the field of Artificial Intelligence (BME IE-MI-SC TKP2020). This research has been has been partially sponsored by NSF grant No. 2018754. This article is based on work from COST Action CA15127 ("Resilient communication services protecting end-user applications from disaster-based failures" – RECODIS), supported by COST (European Cooperation in Science and Technology). This manuscript has been authored by an author at Lawrence Berkeley National Laboratory under Contract No. DE-AC02-05CH11231 with the U.S. Department of Energy. The associate editor coordinating the review of this article and approving it for publication was A. Dhamdhere. *(Corresponding author: Ferenc Mogyorósi.)*

Ferenc Mogyorósi, Alija Pašić, Péter Revisnyei, and János Tapolcai are with the MTA-BME Future Internet Research Group, Department of Telecommunication and Media Informatics, Faculty of Electrical Engineering and Informatics (VIK), Budapest University of Technology and Economics (BME), 1117 Budapest, Hungary (e-mail: mogyorosi@tmit.bme.hu; pasic@tmit.bme.hu; revisnyei@tmit.bme.hu; tapolcai@tmit.bme.hu).

Richard Cziva is the with Energy Sciences Network, Lawrence Berkeley National Laboratory, Berkeley, CA 94702 USA (e-mail: rcziva@lbl.gov).

Zsolt Kenesi is with the Ericsson Research, Research Area Artificial Intelligence, Ericsson Hungary Ltd., 1117 Budapest, Hungary (e-mail: zsolt.kenesi@ericsson.com).

Digital Object Identifier 10.1109/TNSM.2021.3050964

I. Introduction

IN TODAY'S connected era, communication networks are considered among the topmost critical infrastructures. The new mission-critical applications such as telesurgery or stock market clearly demand a higher Quality of Service (QoS) of the underlying network infrastructure. Hence stricter Service Level Agreements (SLAs) are expected to satisfy the requirements of such critical communication services on which not only governments but also people rely more and more.

The SLA is a formal contract between a service provider and a subscriber that defines the QoSs, i.e., defines the so-called Service Level Specifications (SLSs) detailing the technical specifications [1], [2]. The primary metrics usually associated with QoS are packet loss, packet delay, guaranteed throughput, and port availability [3]. Nonetheless, back in 2002, the notion of "service availability" was introduced, which measures the fraction of time the service can be provided to the customer [3].

There are two fundamentally different ways to improve connection availability in transport networks. In proactive approaches, the goal is to prepare for failures so that when failures occur, no reconfiguration is required inside the network. This is achieved by sending the same data along multiple paths so that in case of failure, only the destination node has to take action. At the same time, in reactive approaches, the network is reconfigured once a failure occurs. Reactive strategies can save a significant amount of bandwidth compared to proactive; however, they need network reconfiguration immediately after the failure, which can be slow in practice. Note that devices send an alarm if they perceive any unordinary behavior. Network devices are so much interconnected that a single failure causes sending alarm messages from many devices, called alarm storm. It results in long queues in the signaling channels; the network will be in a temporal "shock", making recovery much slower than expected. As a result, the simplest proactive approach, *dedicated protection*, is currently the de-facto solutions [4] for achieving the given availability target due to their simplicity, robustness, and flexibility.

Reactive approaches can save bandwidth because they are adaptive compared to the proactive ones, i.e., they provide

recovery configurations depending on the failure. Motivated by the above, this article proposes an adaptive protection scheme; however, it is not adaptive to the failure, as we want to avoid reconfiguring the network under a shock. Instead, we propose a protection scheme that is adaptive to the current network state, such as traffic load and the probability of violating a SLS. Its core is an intelligent module that periodically re-allocates the unused capacity as protection bandwidth to maintain the service availability requirements of each connection defined in the corresponding SLS. Note that in ESnet[1] the capacity is over-provisioned for peak rates; however, a significant amount is unused most of the time, see also Fig. 7. The idea is to utilize this unused capacity adaptively for a proactive protection scheme, with the drawback that the user occasionally perceives temporal bandwidth limitations. Note that re-allocating protection bandwidth does not affect operational connections.

The first question we are facing is, how often should the scheme adaptively reallocate the protection bandwidth. To avoid frequent configuration of the switches, performing a reconfiguration in every second is practically the shortest period. Having prolonged periods has a drawback that because of a sudden traffic peak, the user perceives bandwidth limitations. There are many reasons for the aggregated traffic rate fluctuations on the links. It can be (1) due to its normal stochastic behavior after aggregation, (2) due to traffic shifts within the network caused by popular contents, (3) a result of a network reconfiguration for example for performance optimization by traffic engineering [6], or (4) simply a start of an "elephant flow". As several measurement-based studies have revealed, flow statistics exhibit strong heavy-tail behaviors in various networks (including the Internet) [7]. This characteristic is often referred to as the elephant and mice phenomenon (a.k.a. the vital few and trivial many rule); i.e., most flows (mice flows) only have a small number of packets and a significant part of them do not last over a couple of seconds, while a very few flows (elephant flows) have a large number of packets and last significantly longer [8].

We are focusing only on peaks caused by (2)-(4) when adaptively tuning the protection bandwidth. These are all much slower and can be adapted with periods of 10 seconds or even 10 minutes.

The feasibility of the proposed approach depends on how precisely we are able to predict the unused (spare) capacity of the links. Backbone traffic combines traffic from multiple downstream operators and peers, resulting in fluctuating aggregate traffic. Nevertheless, the real-time traffic prediction in operational backbone networks gained significant interest since the breakthrough in Machine Learning. Our paper is a case study using the dataset publicly available for the Energy Sciences Network (ESnet). ESnet is a multi-100Gbit/s international science network, connecting research laboratories and distributing scientific datasets from experiments such as the Large Hadron Collider (LHC) at CERN. In average 39% percent of the traffic of the Energy Sciences Network[2] is by regular Internet users, while the rest has different characteristics. Science traffic has different characteristics compared to traffic at commercial Internet providers: flows are typically larger by size and speed (up to PB transferred consuming up to 70-80 Gbit/s individual flow throughput), and in general, they are machine to machine (e.g., science instrument to data storage) data transfers instead of traffic created by humans, i.e., data distribution jobs. Note that elephant flows, despite being relatively less, are those accounting for most of the traffic load. According to our measurements[3] only 0.77% of the flows have a traffic higher than 10^8 bytes, but they are responsible for 66% of the total traffic. In Section V, we will show that predicting the elephant behavior is sufficient to obtain fair estimates of the load in the future and allows us to allocate protection bandwidth to improve the availability by two-nines.

The main contributions of the paper are the following:

- We propose a novel protection framework that does not require any reconfiguration immediately after the failure, and it does not require any reserved backup network resources either. This is achieved by predicting the traffic for the next period on each link, and intelligently selecting the best fit dedicated protection scheme for the next period depending on the estimated unused (spare) bandwidth, and the estimated probability of violating the required service availability according to the SLS.
- We present how to use the estimated spare capacity to guarantee that all the availability requirements are satisfied. We provide a set of dedicated protection solutions for multiple trade-offs in the bandwidth and service availability from three to six nines (see Fig. 1).
- We evaluate the state-of-the-art traffic prediction approaches based on prediction accuracy and training time. In addition, we introduce a custom loss function to prefer overestimation in order to avoid bandwidth limitations. We used a real-life case study where the scheme adaptively utilized the over-provisioned bandwidth to improve the availability from three-nines to five-nines, such that the user perceives a minor and very temporal bandwidth limitation in less than 0.1% of the time.

The rest of this article is organized as follows: In Section II the background of the work is presented. In Section III, a brief overview of the subject of the case study is presented (ESnet). In Section IV, the model of our novel framework is presented, and finally, the experimental results in Section V.

II. Background and Related Works

A. Traffic Prediction Techniques

Most of the recent studies on traffic prediction are based on the following linear and ML techniques:

1) Autoregressive Integrated Moving Average (ARIMA) [9]: is one of the most well-known models in statistics. The ARIMA uses a linear equation to forecast a stationary time series, in which the predictors consist of lags of the dependent variable and the forecast errors. An ARIMA(p, d, q) model has three parameters: p is the number of autoregressive terms

[1]ESnet commonly applies a 5× overprovisioning over the average utilization when purchasing new links, as suggested in [5].

[2]See the real-time data at https://my.es.net/traffic-volume, The average is taken over the last five years. It was 38,8% in August, 2020.

[3]Data taken from a 15 minutes period of live production traffic on Sept 15, 2020 of ESnet.

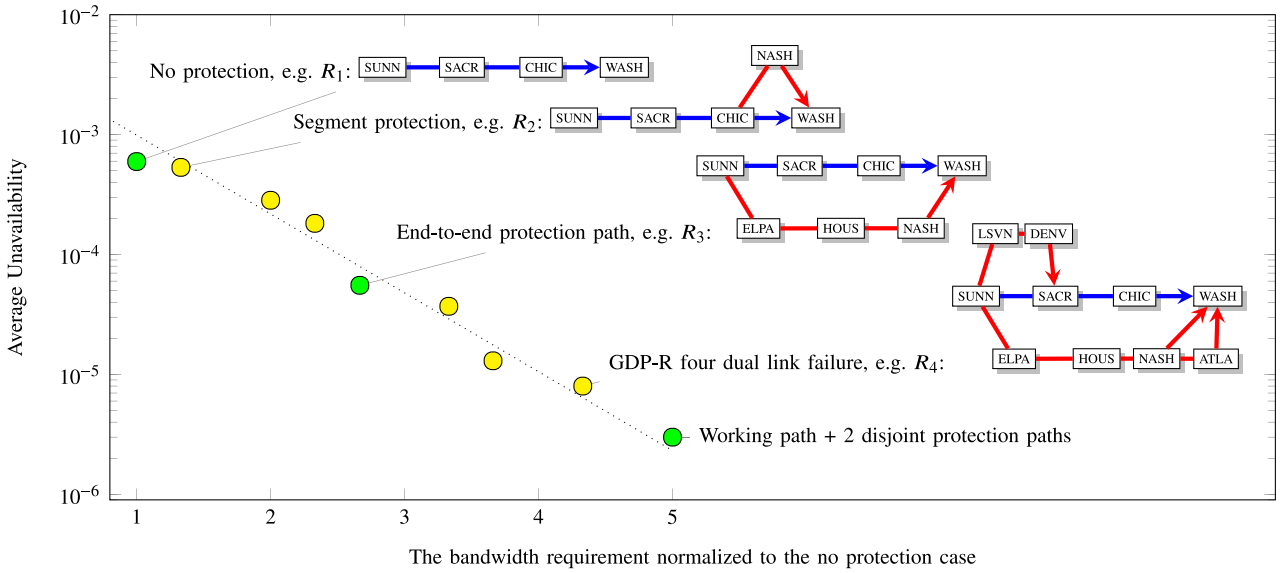

Fig. 1. The availability and bandwidth allocation trade-off represented on the logarithmic scale. Note that the dots represent binned (aggregated) values. We have also added examples of dedicated protection approaches in red between Sunnyvale (California) and Washington D.C. in ESnet's U.S. terrestrial backbone.

(i.e., the number of lag observations included in the model), d is the degree of difference, i.e., the number of nonseasonal differences necessary for stationarity, and q is the number of lagged forecast errors, i.e., the size of the moving average window. Some well known special cases are the ARIMA(1, 0, 0) first-order autoregressive model, ARIMA(0, 1, 0) random walk, and ARIMA(0, 1, 1) exponential smoothing model.

2) Long Short-Term Memory (LSTM) [10]: neural networks are an improvement of the recurrent neural network (RNN) [11] to avoid the vanishing or exploding gradient problem. LSTM introduces new gates (such as input and forget gates), which allow better control over the gradient flow and enable better preservation of long-range dependencies. The neural network utilizes a gated cell too, where the cell decides whether or not to store or drop the information. The purpose of these gates is to have a long term memory to learn from experiences that have a very long time lags in between. LSTM keeps the gradient steep enough, and therefore the training time is relatively short; nonetheless, it yields accurate results for most of the problems. Since the LSTM neural network was built to deal with time series, it is the most used neural network for traffic prediction.

3) Convolutional Neural Network (CNN): was developed with the idea of local connectivity. The CNN consists of a sequence of convolutional layers, which are connected only to local regions in the input. The idea is to apply a sliding filter over the input, then for each point compute the dot product (i.e., convolution) between the input and filter [12]. This structure allows the model to learn filters that can recognize specific patterns in the input data. Forecasting time series with CNNs is still rare, as these types of neural networks are much more commonly applied in classification problems. In [13], an undecimated convolutional network was proposed based on the undecimated wavelet transform. In [12], the authors present a method for conditional time series forecasting based on an adaptation of a deep convolutional WaveNet architecture. The proposed neural network contains stacks of dilated convolutions that allow it to access a broad range of history when forecasting.

Of course, other methods can be used for traffic prediction, too. For example, recent advances in machine learning techniques show that Reinforcement Learning (RL) [14], Tree-RNN based solutions [15], Echo-state Restricted Boltzmann Machines (ERBM) [16] or Spatio-Temporal Graph Convolutional Networks (STGCN) [17] could be a viable option. Nonetheless, in most studies, LSTM still performs very well.

B. Performance of Traffic Prediction Approaches

In [18], [19], [20], and [21] several linear approaches such as Autoregressive Integrated Moving Average (ARIMA), Autoregressive Autoregressive (ARAR), and Holt-Winters (HW) were compared with the non-linear approach, i.e., neural networks. The experimental results showed that in most configurations, the non-linear models outperformed the linear models. In [18] the best non-linear model (FFN MRL - Feedforward Neural Network with Multi-Resolution Learning) performed 87% better than the best linear model but in [20] the FFN (Feedforward Neural Network) achieved a 2% higher average error than the ARIMA. In [22], the performance of several deep learning prediction approaches were investigated in the GÉANT network. Namely, the performance of various Recurrent Neural Networks (RNN), i.e., Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Identity Recurrent Unit (IRNN) was evaluated on real network data. It was shown that the LSTM performs well compared to other RNN, FFN (Feedforward neural network), and classical methods. Besides, it was highlighted that the performance of both GRU and IRNN techniques is comparable to LSTM although the LSTM network achieved $20-40\%$ lower MSE. In [23], the authors use the Adaboost training method to improve the

TABLE I
SUMMARY OF THE RESULTS FROM THE RELATED WORKS. EACH ROW CORRESPONDS TO AN ARTICLE. THE NUMBERS ARE THE ERRORS RELATIVE TO EACH OTHER (IN THE ROW) IN PERCENTAGE [%]. LESS MEANS BETTER

	HW	ARIMA	FFN	LSTM	FFN MRL	AdaBoost LSTM	LSTM DNN
[18]	69	100	36	-	13	-	-
[19]	-	98	100	-	-	-	-
[20]	-	100	94	-	-	-	-
[21]	100	80	75	-	-	-	-
[22]	-	-	100	46	-	-	-
[23]	-	87	-	100	-	88	-
[24]	-	-	-	100	-	-	58

training of the LSTM network and forecast Internet traffic. The utilization of Adaboost improved the LSTM's performance but the classical (ARIMA) model still outperformed it by 1%. In [24], a model of a neural network was proposed, which can be used to combine Long Short Term Memory networks (LSTM) with Deep Neural Networks (DNN). It was shown that the LSTM-DNN model outperforms the LSTM model in certain cases but it requires large and high granularity datasets. The summary of the described results can be seen in Table I. Each row represents the results of one article. Since the articles used several error functions the results in the table are standardized. In each row 100%, corresponds to the highest error.

C. Dedicated Protection Approaches

In today's transport networks, the so-called $1 + 1$ is the most widespread dedicated protection approach. With $1 + 1$, the data can be sent parallel on disjoint primary and backup paths, providing instantaneous recovery against single link failures in a simple manner. This approach is most commonly implemented using network overlays configured with multiple label switched paths taking different physical routes.

In [25], General Dedicated Protection (GDP) was introduced, which enables protecting arbitrary failure patterns listed in the *Shared Risk Link Group* (SRLG) list ($\mathcal{F}$). SRLG consists of a set of links that are considered to share a common resource (e.g., links sharing a duct, a cable or fiber). The GDP with Routing (GDP-R) generalizes the rigid SRLG-disjoint path structure of $1 + 1$ to routing along an arbitrary directed acyclic graph (DAG) between the source and target nodes (see Fig. 1 as an example). Implementation wise it allows three different node operations: simple transfer nodes, splitter nodes that split the data stream into multiple substreams, and merger nodes that can merge multiple data stream into a single one. Thus a GDP-R connection can be dedicated segment protection or even a densely meshed subgraph between the source and destination node depending on the amount of available bandwidth and availability requirements (see Fig. 1).

More specifically GDP-R protects all protectable[4] failures $f \in \mathcal{F}$ by calculating a minimum cost path in every failure graph obtained by removing the failed edges of f and adds it to the solution. Hence, it provides an extremely high connection availability and instantaneous recovery even in sparse networks for the price of increased (however still moderate) bandwidth consumption. GDP-R minimizes the total bandwidth cost and provides the optimal solution for the non-bifurcated scenario (called GDP-R), i.e., when the entire user data is sent along with all links of the connection as in $1 + 1$. It was shown that finding an optimal GDP-R solution in terms of bandwidth cost is NP-complete [25].

[4]We call a failure $f \in \mathcal{F}$ *protectable* if the network topology remains $s - t$ connected after removing the links in f.

III. CASE STUDY - THE ESNET NETWORK

Energy Sciences Network (ESnet), the world's fastest dedicated science network, is the primary provider of international network connectivity for world-leading research laboratories and international experiments (such as the Large Hadron Collider (LHC), Coupled Model Intercomparison Project (CMIP), and International Thermonuclear Experimental Reactor (ITER)) and the single largest supporter of basic research in the physical sciences in the U.S. ESnet, operated from the Lawrence Berkeley National Laboratory, manages a multi-100Gbit/s optical network in the U.S. and the EU with over 500Gbit/s transatlantic capacity interconnecting them. It carries over 100PB of science data each month, with some transfers reaching 300 Gbit/s. Similar to many backbone networks, ESnet utilizes MPLS [26] for Layer 2.5 to implement bandwidth utilization, traffic-engineering, and resiliency. It deploys On-Demand Secure Circuits and Advance Reservation System (OSCARS) to enable on-demand provisioning of guaranteed bandwidth secure circuits within ESnet.

A. ESnet Characteristics

Science mission networks, including ESnet, have different characteristics compared to most commercial Internet Service Providers. The main differences are the followings:

1) Participants With Dedicated QoS: Science overlay networks, such as the LHC Open Network Environment (LHCONE) network dedicated to the LHC experiments at CERN, are well-bounded and defined communities. As such, these networks are only available to known participants where site coordinators and funding agencies have agreed to specific policies, including dedicated bandwidth between data storage and production facilities. This simplifies the separation and identification of network traffic, making traffic analysis and engineering tailored for a particular site and science experiment.

Dedicated QoS has paramount importance for many science experiments, where data can not be stored at the data acquisition facilities. As a concrete example, some telescopes are deployed at remote locations with no data storage facility. For these experiments, data is being streamed to remote storage facilities requiring a certain bandwidth while they are running.

2) Users and Usage Trends: While commercial Internet providers transfer the traffic of individual users, science networks in addition transfer machine to machine data transfers, i.e., data distribution jobs. This means that seasonality trends of human connections, such as evening peak times, can not be easily observed in science network traffic. This is also explained by the fact that science networks transfer data between continents, with different timezones (9 hours difference between two points of the network). In particular, ESnet transfers most of its transatlantic data from CERN in Switzerland to Fermilab and Brookhaven National Laboratory in the U.S., a distance of around 7000 km, while the majority of commodity traffic travels only about 280 km to the nearest CDN [27].

3) Growth of Traffic: Science data traffic carried by ESnet is exponentially increasing 36.6% each year. This increase is higher than the growth of commodity IP traffic globally, which is estimated at 26%, according to Cisco's latest forecasts and trends for 2022 [28]. Scientific network traffic is opportunistic and has a lot of uncertainty due to its nature. While usually only compressed, aggregated, filtered data is being transferred, when a scientific discovery is suspected, researchers can pull terabytes of raw data over the network – a format they only look at opportunistically.

4) Type of Traffic: While Web browsing and streaming media [29] dominate commercial Internet traffic, science networks generally move data using TCP-based file transfer protocols, such as GridFTP [30], or XRootD [31] that do not tolerate congestion and the resulting packet loss well. Science data flows tend to use larger packet sizes (average of 1500 byte packets, up to 9000 bytes jumbo frames), while commodity traffic uses 500-byte packets on average [32].

The network contains various link types: for example, backbone core links, access links, and cloud peering links. The ratio of "human" traffic (traffic of individual users) and "machine" traffic (machine to machine data transfers) is different on each link; some of them transmit mostly human traffic while some of them transmit almost purely machine traffic.

ESnet manages two virtual networks – an IP network to carry day-to-day traffic, including e-mails, video conferencing, etc., and a circuit-oriented Science Data Network to haul massive scientific datasets. Part of the scientific data is transmitted through a multi-domain layer 3 Virtual Private Network called LHCONE (LHC Open Network Environment), and the rest through reserve bandwidth channels by OSCARS. In August 2020, during the whole month, 34PB (47%) of the traffic was routed in LHCONE, 28PB (38%) was normal Internet traffic, and 10PB (14%) was OSCARS.

B. Generalization of Our Solution to Other Networks

While we present our solution on a specific network of ESnet, the solution is fully generalized to other network providers. The following points detail some of the technical aspects to take into consideration when applying this method on other networks.

1) Network Statistics: To retrieve network statics, we are using the widely deployed SNMP protocol to collect all interface counters from all production routers, every 30 seconds. SNMP protocol was introduced in the 1980s and is still the de-facto standard for interface statistics to this day. Newer interface statistic collection methods currently replacing SNMP in some networks (e.g., streaming network telemetry) can also be used, as they provide better quality data than SNMP. In general, the finer the resolution of the network telemetry, the better the forecasting will become.

2) Network Control: QoS can be controlled in different ways. ESnet uses the industry-standard bandwidth reserved MPLS circuits managed by OSCARS, ESnet's On-Demand Secure Circuits and Advance Reservation System. In particular, Label Switch Paths (LSPs) are set up - primary and backup - with bandwidth reservations that can be dynamically re-configured using open APIs. Other QoS enforcement solutions can be specific to the network management technology (such as SD-WAN, Segment Routing), vendor-specific (e.g., Cisco Per-Session QoS), or fully SDN-controlled [33]. It is important to note that a network without any bandwidth reservation scheme would not be able to utilize our solution. This could be the case for some virtual or overlay operators only manage L2 or L3 connectivity between PoPs, without control to protected bandwidth or the physical infrastructure.

3) Network Topology: For a protection scheme to work, multiple paths are required between sources and destinations in the network. Our case study uses a continental-scale terrestrial network that is constructed around a high-speed, fault-tolerant core ring that provides at least two connections between any PoPs. Large customers are connected with at least two (up to six) access links to two physically distanced routers. This ring-based, multi-homed network topology is used at science networks (e.g., JANET in the U.K.) as well as commercial telco operators (e.g., AT&T in the U.S.) allowing our protection scheme to be used in a wide variety of networks. While most DC topologies with multiple paths between hosts (e.g., clos topology) can also apply our method, single-path topologies (e.g., hierarchical tree) or topologies with partial single-path would not benefit from any bandwidth protection scheme.

4) Compute Requirements: Our solution uses a single VM with an Intel Xeon CPU, an NVIDIA Telsa K80 GPU, and 16GB of memory to run our network modeling and predictions which we define as minimum requirements. This VM is available from Google Cloud for anyone and allows scaling up with larger GPUs and more RAM.

IV. The Model

In this section, we present the concept of our framework. The overall architecture is visualized in Fig. 2. As shown, our framework relies on real-time data input from a network controller (i.e., an SDN or WAN controller) by pulling network state from them and configuring protection bandwidth in regular intervals. The input for our solution is a Network State

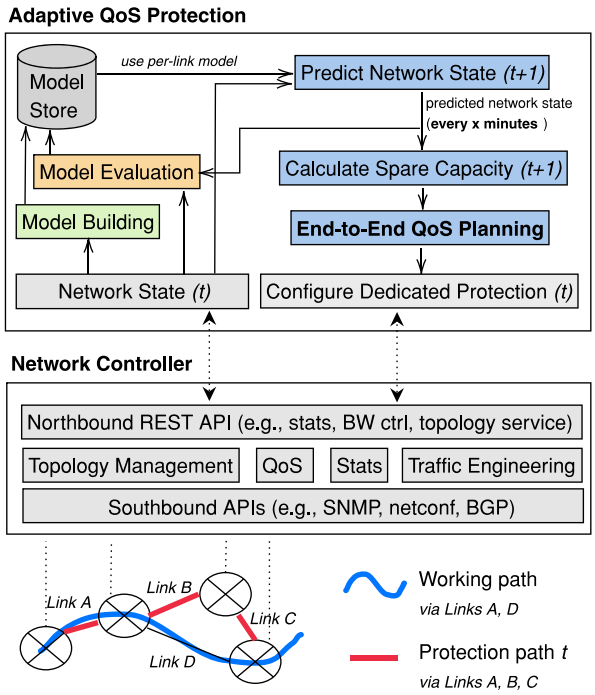

Fig. 2. High-level concept of QoS enhancement with the help of ML. The QoS level is adjusted every x minutes.

at time t representing the traffic load in the network and the availability of the connections. Utilizing custom deep learning methods (see Section IV-A), we extract patterns from the historical traffic information and predict the traffic load for the next period. According to that, we estimate the spare capacity for $t + 1$, i.e., for the following period, and adjust the dedicated protection scheme, i.e., the QoS level for each connection (see Section IV-B). The length of the period is adjusted according to the forecastability of the traffic and the rerouting process itself. In other words, we have to take into account that it is more difficult to predict the traffic pattern further in the future if the forecastability is low and that the rerouting process can be time-consuming.

A. Traffic Prediction

The traffic prediction is pivotal for enhancing the QoS. At the initial stage, several time-series prediction techniques were implemented and evaluated such as ARIMA, SARIMA, RNN, and LSTM. We assessed different neural networks, like Feedforward Neural Network, Recurrent Neural Network, and Convolutional Neural Network. In Figure 3 we show the trade-off between the RMSE of the predicted traffic (in Mbps) and the training time achieved by each network type. A vast number of ARIMA and SARIMA models with different p, d, and q parameters were compared to find the classical traffic prediction method which achieves the lowest RMSE. The best obtained classical method was an ARIMA with parameters $p = 3$, $d = 1$, and $q = 2$, which achieved an RMSE (Root Mean Square Error) of 4451. The Feedforward Neural Network performed poorly compared to other neural networks

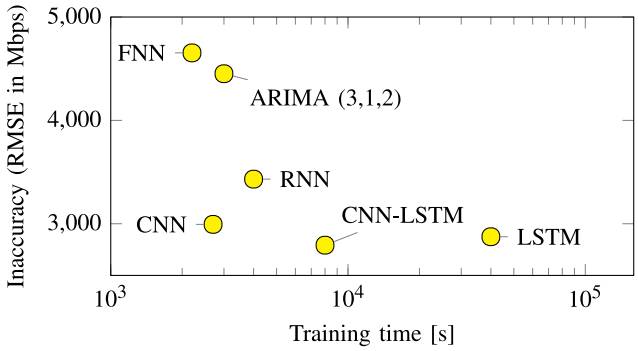

Fig. 3. Comparison of the prediction methods in terms of precision (RMSE, Root Mean Square Error) and training time (of a time series of one year).

since the method itself was not designed for time-series analysis. Although it achieved similar RMSE to the ARIMA method while having almost the same training time too. The RNNs performed significantly better than the ARIMA models, however, the LSTM networks outperformed each of them at the achieved RMSE at a cost of the larger training time. Convolutional Neural Networks had a clear edge at training time, although, they performed surprisingly well at extracting features from time-series. This is the reason why hybrid neural networks were implemented and evaluated. These hybrid networks consist of Convolutional Neural Network (CNN) linked together with an LSTM network. For each network, several hyperparameter setups were evaluated, among others the number of layers varied between 1 and 5, the number of neurons varied between 16 and 128. For the convolutional networks the number of filters varied between 50 and 200, the width of a filter between 3 and 11, and the stride between 1 and 3.

During this initial evaluation, we attained the two best performing network types, i.e., the LSTM and the CNN-LSTM hybrid network (as it can be seen in Figure 3) which are further discussed in detail in the rest of the paper.

1) The Best Models: The convolutional layers were used to extract features from the traffic, which was utilized by the LSTM layers. All models were trained and evaluated on real ESnet traffic data. For each neural network type, around 100 models were trained to optimize the hyperparameters.

The structure of best pure LSTM network can be seen in Table II:

- It contains 3 LSTM layers with 64 neurons per layer. Note that lower complexity was not sufficient to capture real trends in the traffic. With increased layer and neuron numbers, the neural network was more demanding to train nevertheless yielded similar results.
- For the activation function, we used the Rectified Linear Unit (ReLU) non-linearities since ReLU is highly successful for most applications and is proven to be faster to train than the standard sigmoid units.
- To overcome overfitting, dropout and weight constraints were utilized. Note that the dropout is randomly dropping units (along with their connections) from the neural network during training [34].

TABLE II
THE STRUCTURE OF THE BEST LSTM NETWORK

Layer type	Output shape	Param #
LSTM	(None, 64, 64)	16896
Dropout (0.2)	(None, 64, 64)	0
LSTM	(None, 64, 64)	33024
Dropout (0.2)	(None, 64, 64)	0
LSTM	(None, 64, 64)	33024
Dense	(None, 8)	520
Total params		83,464

TABLE III
THE STRUCTURE OF THE BEST CNN-LSTM HYBRID NETWORK

Layer type	Output shape	Param #
Conv1D	(None, 58, 100)	800
Conv1D	(None, 52, 100)	70100
MaxPooling1D	(None, 26, 100)	0
Conv1D	(None, 22, 100)	50100
Conv1D	(None, 18, 100)	50100
MaxPooling1D	(None, 9, 100)	0
Flatten	(None, 900)	0
RepeatVector	(None, 8, 900)	0
LSTM	(None, 8, 64)	247040
Dropout (0.2)	(None, 8, 64)	0
TimeDistributed	(None, 8, 32)	2080
TimeDistributed	(None, 8, 1)	33
Total params		420,253

The structure of the best CNN-LSTM hybrid network can be seen in Table III:

- Several one dimensional convolutional and pooling layers are utilized. These are responsible for the feature extraction. The size of the filters is 7 in the first two and 5 in the second two convolutional layers. Note that the key difference between 1D, 2D, and 3D layers is the dimensionality of the input data and how the feature detector (or filter) slides across the data.
- After the CNN layers, an LSTM layer (with 64 neurons and ReLU activation function) and 2 fully-connected layers were added to perform the prediction based on the extracted features.

Although the number of hyperparameters and parameters of the hybrid neural network are very high given we have to adjust among others the number of layers, number of filters, length of filters, number of fully-connected and LSTM layers (see Table III), the training time of the network is much less than the LSTM network as a result of the convolutional layers.

The models were trained utilizing several different loss functions, e.g., our custom loss function and MSE. The performance of the neural networks in different scenarios is demonstrated in Section V-C.

2) Custom Loss Function: Since we aspire to use the spare capacity of the network to provide higher QoS (i.e., high availability) to given applications, it is more adverse to underestimate the traffic than to overestimate the same. Hence a custom loss function was implemented to punish more severely the underestimation of the traffic. The loss function that fulfills our requirements is a modified version of mean absolute error. We punish the overestimation of the traffic with an absolute error, and the underestimation with an additional component,

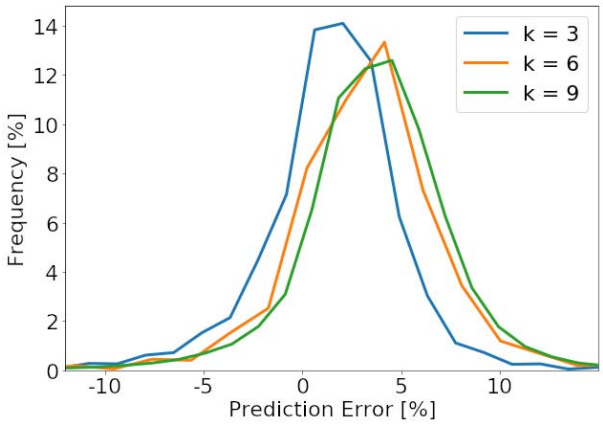

Fig. 4. The error distribution produced by neural networks trained with different k parameter of our loss function in our numerical evaluation.

formally:

$$\mathbf{\Delta}(\mathbf{y}, \hat{\mathbf{y}}) = \hat{\mathbf{y}} - \mathbf{y} \tag{1}$$

$$\mathbf{\Delta}_{-} = \frac{\mathbf{\Delta} - |\mathbf{\Delta}|}{2} \tag{2}$$

$$c(\mathbf{y}, \hat{\mathbf{y}}) = \overline{\mathbf{\Delta} + k\mathbf{\Delta}_{-}} \tag{3}$$

where $\mathbf{y}$ and $\hat{\mathbf{y}}$ are the real and predicted traffic vectors, respectively. $\mathbf{\Delta}$ is the difference of the predicted and real traffic vectors. $\mathbf{\Delta}_{-}$ is a filtered version of $\mathbf{\Delta}$ which contains only the negative values, and the positives are replaced with zeroes. Every operation is elementwise except Eq. (3), where we take the mean of the vector and the resulted scalar is the loss. Parameter k is used to fine-tune the loss function to a certain level of overestimation in the training set. Our objective was to create a loss function that overestimates the traffic more than 90% of the time (in the training set) while holding the overestimation on an acceptable level. The error distribution for different k values can be observed in Fig. 4. It is apparent that by increasing k, the distribution shifts slowly towards overestimation. Our goal, to overestimate 90% of the time in the training set, was achieved with $k = 9$, i.e., when the underestimation is punished 9 times more severely than the overestimation. The performance of the different neural networks is discussed in Section V.

B. QoS Differentiation Using Dedicated Protection

Fig. 2 shows the work cycle of the proposed framework. It can be observed that after we predict the traffic, i.e., the network state for the next period, we can obtain the spare capacity available for each connection in the following period (e.g., for the following $x = 4$ minutes in our example). According to the spare capacity estimation and the QoS requirements (i.e., SLA), we can adjust the dedicated protection scheme (enhancing the availability of the connection). The dedicated protection schemes have to be pre-calculated to minimize the delay of the routing adjustment.

Let us assume that our connection C is unprotected, i.e., the data is sent solely along the Working Path (WP). According to Fig. 1 in this scenario, the availability is approximately 3 nines. However, if we are able to accurately predict the spare

capacity, we are able to enhance the availability to four (if three times the WPs capacity is available in the network) or even six nines (if approximately five times the WPs capacity is available in the network) depending on the network load.

Meaning if we estimate that along the pre-calculated backup routes with the spare capacity requirements $B_1, B_2, \ldots, B_n$ $S_1, S_2, \ldots, S_n$ spare capacity is available, we can adjust the dedicated protection approach to any of the pre-calculated schemes, where $B_n \leq S_n$. We emphasize that if due to the traffic, the real spare capacity is below the estimated value, we assume the worst-case, i.e., all backup routes are preempted, and only the WP remains allocated, resulting in a decrease in the availability.

When utilizing traditional dedicated protecting approaches the granularity for the adjustment is very sparse, i.e., we can either use a single WP, the $1+1$ dedicated protection approach, or if there are 3 disjoint paths available between the source and destination node - which is rarely the case - even the $1+1+1$ dedicated protection approach (i.e., the green dots in Fig. 1). Nonetheless, if GDP-R is utilized, a much more fine-grained adjustment is possible. Note that in Fig. 1 the blue dots are aggregated (binned) values.

In Section V, we will demonstrate the availability improvement possible with this fine-tuned framework. We will show the improvement through an example, however, our framework is flexible and can be adjusted according to various requirements.

V. Experiments Setup and Results

This section presents the major aspects of the experimental setup. Note that, the most fine-grained granularity data was 30 sec in our evaluation, and we predict the next 8 data points, that is the next 4 minutes. We assume if the resolution of the traffic measurements is better (e.g., 100ms), there is a higher correlation between the adjacent data points; thus, much higher precision is expected. Nevertheless, we will show that even at this sampling rate, sufficient prediction of the bandwidth is achieved for our application.

We will demonstrate the availability improvement possible with this fine-tuned framework through an example in ESnet where the traffic is forecasted for the next $x = 4$ minutes, the overestimation is set to 90% on the training set, and only the protection routes demonstrated in Fig. 1 are utilized. In particular, the performance of an LSTM and Hybrid neural networks is discussed in detail. However, note that our framework is flexible and can be adjusted according to various requirements.

A. Dataset and Evaluation Process

ESnet interconnects 60 sites (nodes) with 75 high capacity links. Most interface's traffic is available from the beginning of 2016 with a resolution of 30 seconds. Due to a high number of planned outages and maintenance, the data required extensive preprocessing. The lone missing values were interpolated; however, by lengthy outages (caused e.g., by fibre cable repair), the dataset had to be divided. Fortunately, less than 0.1% of the dataset was missing and the outages could be used in the simulation as failures. In the next step, the values were rescaled between 0 and 1 and were used to create the input and output sequences for the neural networks. The input sequences were 64, and the output sequences 8 timesteps long. The length of the output sequence was set according to the length of the time period (i.e., $x = 4$ minutes) and the length of one time step (i.e., 30 seconds).

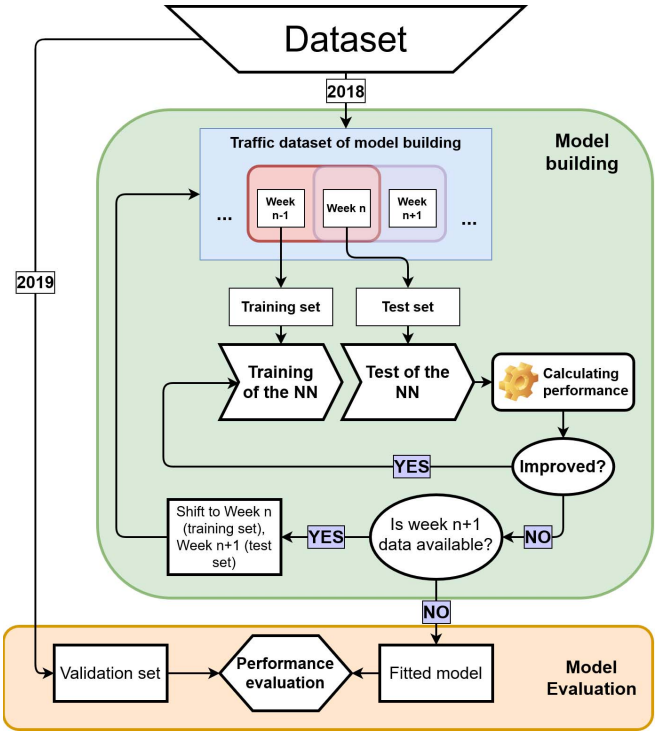

Fig. 5. Illustration of the fine-tuning and performance evaluation process.

The model training and evaluation process is displayed in Figure 5. First, the dataset is split into a validation set (2019 data) and a dataset for model building, i.e., training and test set (2018 data). The traffic dataset for model building is further divided into weekly portions. In each training iteration (several epochs), the sequences of weekly traffic were fed into the neural network, for training week n $-$ 1, and for testing the next weeks' traffic, i.e., week n. The neural network was trained on the given week n $-$ 1 until its performance stopped improving on the test set, i.e., week n. In other words, early stopping was performed to avoid overfitting. When the training was finished for the given input, i.e., week n $-$ 1, the process continued with the next training/test set, i.e., week n as the training set and week n + 1 for the test set. In the end, we obtain a fine-tuned model for which the performance is evaluated on the validation set.

B. Traffic Forecastability in ESnet

The ability to accurately determine the a priori forecastability of a time series is essential as it guides the potential for accurate forecasts. Measures of entropy, such as sample entropy [35], provide an assessment of the regularity or similarity within a time series. A more significant value of sample

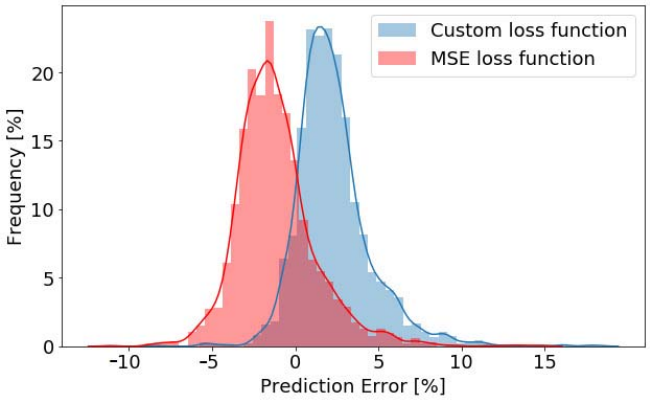

Fig. 6. The distribution of the prediction error in case of the MSE and our Custom loss function ($k = 9$) on the traffic of SACR-CHIC link (2019.08.18. from 13:30 to 19:30).

entropy conveys more disorder, randomness, and system complexity. The measured sample entropies in ESnet ranged from 0.6 to 0.9 for the different links, which indicates that the forecastability of these time series is rather low. In other words, in ESnet, traffic prediction is quite a challenge because it lacks some of the usual periodicity of Internet traffic. Nonetheless, we present neural networks that can capture the information of the historical traffic data and predict the next network state even in ESnet.

Note that our adaptive framework can be utilized in any backbone network, where our approaches can serve as an initial solution and can be fine-tuned according to the specific traffic characteristics to achieve the desired result. Other backbone networks may require adjustments in the structure of the neural network, the asymmetric loss function, and the spare capacity calculation to improve the traffic prediction's performance and avoid traffic congestion. For example, with higher underestimation punishment the overestimations will be larger, resulting in fewer congestions but more unnecessary roll-backs. In our case, the asymmetric loss function with $k = 9$ provides a good balance between the two, resulting in a high average availability improvement for the protected connections.

C. Traffic Prediction Performance

First, we present a loss and training time analysis, and next the performance of the best neural network, i.e., the hybrid neural network is discussed in detail.

1) Loss and Training Time Analysis: As it was emphasized in Section IV-A, the underestimation of the traffic could cause traffic congestion, which should be avoided. This can be observed in Fig. 6, where the frequency of the prediction errors is demonstrated. We can witness the merits of the asymmetric loss function ($k = 9$). The predicted traffic is overestimated in most states; nevertheless, it is close to real traffic. We see that our custom loss function is shifted and skewed to the overestimation. When using the MSE as the loss function, we overestimate the traffic only 23.6% of the time, and the average Root Mean Square Error (RMSE) is 2.7%; however, when using our custom loss function we overestimate the traffic 90.6% of the time and the RMSE is 3.4%. Note that unsurprisingly that the RMSE is higher when using the custom loss function; however, the difference is moderate.

TABLE IV
COMPARISON OF THE TWO TYPES OF NEURAL NETWORKS BASED ON THE REACHED CUSTOM LOSS ($k = 9$) AND TRAINING TIME

	LSTM	CNN-LSTM
Custom loss on the validation set	0.0314	0.0285
Training time of 1 batch	~ 6 ms	~ 0.3 ms

Table IV shows a comparison of the two investigated neural networks based on the reached custom loss ($k = 9$) and training time (The networks were trained on 160,000 batches in both cases). It can be observed that the hybrid CNN-LSTM neural network outperformed the pure LSTM network based on the achieved loss and the training time, too. The training of the hybrid network was approximately 20 times faster than the training of the LSTM network. The loss of the validation set was about 10 % better in the case of the hybrid network. Considering the hybrid network was better in both comparisons, we selected it for the traffic prediction task.

2) Performance of the Hybrid Neural Network: Fig. 7 shows the performance of the hybrid neural network with our custom loss function (denoted as H_1). The blue line is the real traffic between Sacramento and Chicago on the 18th of May of 2019. The green line is the predicted traffic for each period. The red function is the spare capacity estimation for each period calculated from the predicted traffic with a 10% overestimation buffer. We can see that the prediction follows the trends of the traffic well while moderately overestimating the traffic most of the time.

In Table V two hybrid neural networks are compared, the one trained with our custom loss function (H_1) and the other with MSE (denoted with H_2). Two traffic scenarios are taken into account: Traffic A and B, where A denotes the 'In' and B the 'Out' traffic on the link CERN-WASH in the year 2019. Both networks were trained on a training set created from the first 6 months traffic of Traffic A. As expected H_2, i.e., the one trained with MSE achieves much better MSE scores on the train (0.00138 vs. 0.00205) and validation (0.00103 vs. 0.00134) set of Traffic A and B while the H_1 (trained with our custom loss function) achieves better Custom loss scores (0.02850 vs. 0.03092). Although the H_2 can accomplish better MSE scores and not much worse custom loss scores than the one trained with our custom loss, it's not capable to accurately predict the critical high traffic, i.e., congestion periods (see Section V-D) which are crucial for the QoS planner.

D. QoS Enhancement

1) Minimal Example: To demonstrate the benefits of our framework, we present a simple example of ESnet. To make it easily comprehensible, we will exclusively use the routing instances presented in Fig. 1 (even though the GDP-R is able to provide additional alternative, i.e., a more fine-grained scenario) to assess the possible availability enhancement for the SUNN-WASH connection on the 18th of August 2019. We assume that the connection request between SUNN-WASH is

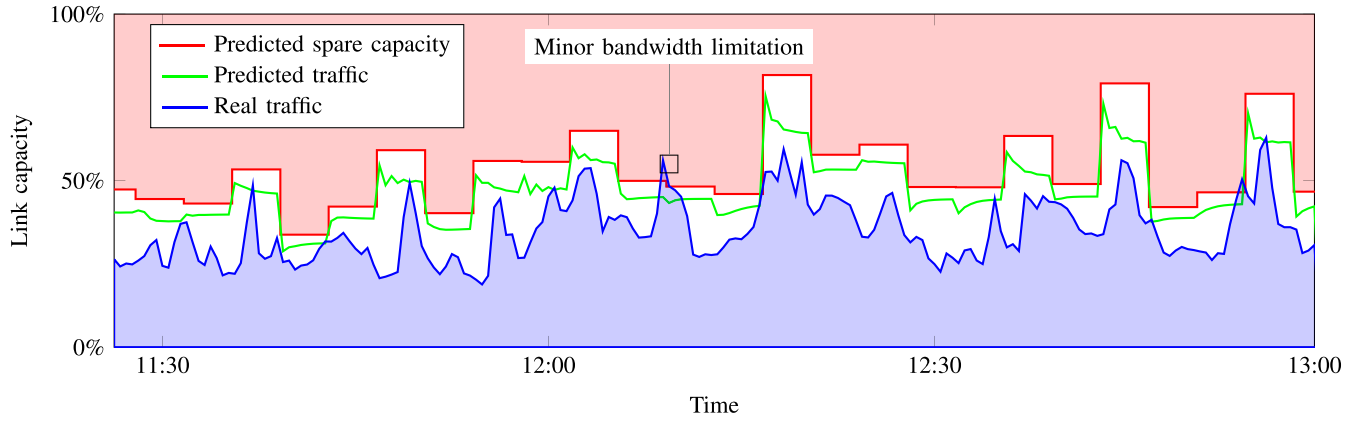

Fig. 7. Traffic prediction ($k = 9$) for the SACR-CHIC link with H_1 (2019.05.18. from 11:30 to 13:00). The predicted spare capacity is the link capacity minus 110% of the predicted traffic.

9Gbps. We have four levels of protection R_1 with availability 0.999306, R_2 with availability 0.999468, R_3 with availability 0.999990, and R_4 availability ≥ 0.999999. Note that the links SUNN-LSVN and LSVN-DENV are only 10G links; hence the most advanced routing R_4 is strongly dependent on the traffic of these 10G links. With our framework, we estimate the traffic on each link of the network (with the custom loss function with $k = 9$) and select the proper dedicated protection scheme in each network state. If the traffic is underestimated and the spare capacity is below the required, we assume the worst-case scenario and roll-back to the single WP, i.e., R_1. Note that a more sophisticated roll-black algorithm could be easily implemented. However, we intend to demonstrate that the availability enhancement is significant even in the worst-case. Our results reveal that we can utilize the most advanced routing (R_4) in 97.2% of the time and the R_3 in 1.9% of the time. We only roll-back unnecessarily in 0.4972% of the time to R_3 and 0.4972% of the time to R_1 due to underestimation of the traffic. With our novel framework, the availability can be enhanced from 0.999306 to 0.999992, i.e., from three nines to five-nines in this particular case. Of course, the availability enhancement strongly depends on the traffic load; nonetheless, our framework can maximize the utilization of the spare capacity in the network.

2) Performance Evaluation for Traffic of 2019: The QoS enhancement performance for other connections was in a similar range. To simulate a real-world environment, a lengthy time-period (an entire year - 2019) was selected from the historical data of ESnet and 30 randomly selected connection requests had to be routed consecutively assuring the highest possible availability (by utilizing the advanced protection routing GDP-R). We assumed that each of our new connection requires 9 Gbps of bandwidth, in order to load the network properly. For each connection request, numerous different protection routes were calculated with GDP-R to provide the highest possible availability for each traffic and failure scenario. The neural networks were trained on the traffic data of all links of the year 2018 (almost 80 million data points) and then fine-tuned on the traffic of each specified link of the connection. As a result, each link had a unique neural network predicting its traffic. In other backbone networks, much less data is sufficient if the traffic is generated by humans. The computation time of one prediction period is 3 seconds (using an Intel Xeon 2.3 GHz CPU) but it can be significantly decreased utilizing parallel computing. The merits of our custom loss function are not reflected in the metrics scores themselves (see Table V) however in the ability to predict congested periods, which are crucial for the QoS enhancement. In Table VI the two best performing hybrid neural networks are compared, i.e., the already introduced H_1 and H_2 (trained with our custom loss function and with MSE, respectively) based on the ability to forecast congestion. A congestion period is defined as a scenario when Backup Paths (resources) have to be discarded (i.e., on a 100G link it means that the traffic exceeding 91 Gbps, since in this case, we can not route our new 9 Gbps traffic request). The performance is analyzed for 30 randomly selected connections. The two example connections (Connection 1 & 2) are Connection 1, the connection between Sunnyvale and Washington, and Connection 2 the one between Sacramento and Chicago.

In Table VI the performance results for congestion prediction are presented. The True positive denotes that the neural network successfully forecasted the congestion, and some of the Backup resources have been adjusted in order to prevent the congestion and the roll-back to the single WP. Note that since we avoid roll-back to the single WP (i.e., we avoid the worst-case scenario), a much more efficient protection routing providing a higher QoS level can be successfully implemented.

The False negative denotes that according to the prediction, no roll-back was necessary, however since the traffic caused congestion, the dedicated protection had to be dropped (i.e., a roll-back to the single WP was necessary). The False positive means that the traffic was overestimated and an unnecessary roll-back was performed to the highest assumed available QoS level according to the framework, i.e., we roll-back to a lower availability protection routing, however not to the single WP.

The results show that the network trained with our custom loss function, i.e., H_1 is able to predict congested periods more reliable than H_2, i.e., the one trained with MSE. In particular, the first two rows of the summary column show the ratio of the successfully forecasted and missed congested periods.

TABLE V
THE EVALUATION OF THE HYBRID NEURAL NETWORKS TRAINED WITH DIFFERENT LOSS FUNCTIONS. ONE WAS TRAINED WITH MSE AND ONE WITH OUR CUSTOM LOSS FUNCTION

Loss function for training	Traffic A (CERN-WASH)				Traffic B (WASH-CERN)			
	Train set		Validation set		Train set		Validation set	
	MSE	Custom	MSE	Custom	MSE	Custom	MSE	Custom
Custom (H_1)	0.00205	0.03011	0.00134	0.02850	0.00178	0.03700	0.00103	0.02136
MSE (H_2)	0.00138	0.03538	0.00103	0.03092	0.00151	0.03691	0.00063	0.02710

TABLE VI
COMPARISON ABOUT THE FORECASTING ABILITY OF CONGESTED PERIODS WHEN SOME OF THE DEDICATED PROTECTION SHOULD BE DROPPED TO AVOID TRAFFIC CONGESTION

	Connection 1 (SUNN-WASH)		Connection 2 (SACR-CHIC)		Summary (30 connections)	
	H_1 (Custom)	H_2 (MSE)	H_1 (Custom)	H_2 (MSE)	H_1 (Custom)	H_2 (MSE)
True positive	99	5	142	13	81% (±1.6%)	7% (±1.4%)
False negative	7	101	39	169	19%	93%
False positive	160	21	416	53	72% (±1.1%)	78% (±3.4%)

While our solution, i.e., H_1 predicted ∼81% of these periods, H_2 missed almost each such period (93%, i.e., only 7% are predicted correctly). In addition note that H_1 outperforms H_2 in the ratio of false positive too (72% to 78%, respectively).

It may appear that using an overestimation buffer for the MSE, i.e., utilizing H_2 and adding a given percentage X% in addition to overestimate the traffic can make the custom loss function obsolete (i.e., H_1), however, this is not the case. Our analysis shows that in order to yield the same prediction accuracy for the congested periods, i.e., the number of True positives as H_1 the overestimation for H_2 has to be in the range of 60% to 110% depending on the links.

Thanks to our custom loss function and the spare capacity calculation, the bandwidth limitations are rare. Note that we define "a bandwidth limitation" very conservatively as the state of the network where the real traffic exceeds our prediction, i.e., the over-provisioned (+10%) predicted traffic. The length of the bandwidth limitation is determined to be the time that remained from that prediction period (i.e., minimum 30 seconds, maximum 4 minutes). In this worst-case scenario, the limitations occur less than 0.1% of the time and the average underestimation (without the +10% over-provisioning) is 13%.

In practice, not every traffic underestimation will cause a real bandwidth limitation and even the real bandwidth limitations will be much shorter than we assumed. A real bandwidth limitation is caused only when the real traffic and the used spare capacity for the protection require more bandwidth than the total link capacity. In this case, the user will perceive bandwidth limitation until the network recognizes the congestion and discards the protection paths. In our demonstrative example, the protection required 9 Gbps, which means that the traffic has to exceed 91 Gbps on a 100 Gbps link for congestion.

Of course, the overestimation of the traffic by H_2 increases the number of unnecessary roll-backs (False positives) too. Our simulation results show that by achieving the same number of True positive predictions we produce 75% more unnecessary roll-backs (i.e., False positives) by H_2 which can be avoided by utilizing H_1. Note that by applying such a huge overestimation buffer (60% to 110%) the RMSE of H_2 deteriorates significantly.

In summary, we can state that H_1 (the network trained with our custom loss function) has the ability to predict the congested periods with accuracy over

$$\frac{TP + TN}{TP + TN + FP + FN} = 99.9\%$$

and with a precision of

$$\frac{TP}{TP + FP} = 28\%,$$

and hence in most of the time, a uniquely high QoS level can be provided for the examined connections in ESnet, i.e., with our novel framework, the average availability of the 30 connections can be enhanced from 0.999296 to 0.999997. The performance of the framework is subject to the network topology and traffic conditions.

VI. CONCLUSION AND FUTURE WORK

In this article, we propose a novel framework to enhance the availability with the help of Machine Learning (ML). We present a new protection scheme that does not require any reconfiguration immediately after the failure, and it does not require any reserved backup network resources either. This is achieved by predicting the traffic for the next period on each link, and intelligently selecting the best fit dedicated protection scheme for the next period depending on the estimated unused bandwidth, and the estimated probability of violating the required service availability according to the SLS. Note that re-allocating protection bandwidth does not affect operational connections, and a significant amount of bandwidth is unused most of the time because the capacity is over-provisioned for peak rates and elephant flows. The proposed scheme utilizes this unused capacity adaptively for a proactive protection scheme. The drawback is that the user occasionally perceives temporal bandwidth limitations, and the approach is less effective during peak hours. The performance is demonstrated through a case study of ESnet. We illustrated that with a fine-tuned neural network for traffic prediction (with a custom loss function) and advanced dedicated protection approaches, we could significantly improve the availability (in ESnet for the 30 examined connections) even in the worst-case scenario,

from three-nines to five-nines, and the users perceive only a minor and very temporal bandwidth limitation in less than 0.1% of the time.

In the future, we aim to extend our work by implementing and testing additional advanced machine learning techniques like Reinforcement Learning, Tree-RNN based solutions, or Spatio-Temporal Graph Convolutional Networks (STGCN).

ACKNOWLEDGMENT

The U.S. Government retains, and the publisher, by accepting the Acknowledgment for publication, acknowledges, that the U.S. Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for U.S. Government purposes.

DISCLAIMER

REFERENCES

[1] W. Fawaz, B. Daheb, O. Audouin, M. Du-Pond, and G. Pujolle, "Service level agreement and provisioning in optical networks," *IEEE Commun. Mag.*, vol. 42, no. 1, pp. 36–43, Jan. 2004.

[2] J. Gozdecki, A. Jajszczyk, and R. Stankiewicz, "Quality of service terminology in IP networks," *IEEE Commun. Mag.*, vol. 41, no. 3, pp. 153–159, Mar. 2003.

[3] G. Iannaccone, C.-N. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures in an IP backbone," in *Proc. 2nd ACM SIGCOMM Workshop Internet Meas.*, 2002, pp. 237–242.

[4] W. Kellerer *et al.*, "How to measure network flexibility? A proposal for evaluating softwarized networks," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 186–192, Oct. 2018.

[5] Y. Huang and R. Guérin, "Does over-provisioning become more or less efficient as networks grow larger?" in *Proc. 13th IEEE Int. Conf. Netw. Protocols (ICNP)*, 2005, pp. 225–235.

[6] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of internet traffic engineering," Internet Eng. Task Force, Fremont, CA, USA, RFC 3272, May 2002.

[7] F. Hernandez-Campos, J. S. Marron, G. Samorodnitsky, and F. D. Smith, "Variable heavy tails in Internet traffic," *Perform. Eval.*, vol. 58, nos. 2–3, pp. 261–284, 2004.

[8] T. Mori, M. Uchida, R. Kawahara, J. Pan, and S. Goto, "Identifying elephant flows through periodically sampled packets," in *Proc. 4th ACM SIGCOMM Conf. Internet Meas.*, 2004, pp. 115–120.

[9] J. Wu and S. Wei, *Time Series Analysis*, vol. 20. Changsha, China: Hunan Sci. Technol. Press, Jan. 1989, p. 2018. [Online]. Available: http://www2.geog.ucl.ac.uk/mdisney/teaching/GEOGG121/time_series/GEOGG121_5_TimeSeries_ Wu.pdf

[10] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. 11th Annu. Conf. Int. Speech Commun. Assoc.*, 2010, pp. 1045–1048.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] A. Borovykh, S. Bohte, and C. W. Oosterlee, "Conditional time series forecasting with convolutional neural networks," 2017. [Online]. Available: arXiv:1703.04691.

[13] R. Mittelman, "Time-series modeling with undecimated fully convolutional neural networks," 2015. [Online]. Available: arXiv:1508.00317.

[14] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT press, 2018.

[15] Y. Shen, S. Tan, A. Sordoni, and A. Courville, "Ordered neurons: Integrating tree structures into recurrent neural networks," 2018. [Online]. Available: arXiv:1810.09536.

[16] X. Sun *et al.*, "Enhanced echo-state restricted boltzmann machines for network traffic prediction," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1287–1297, Feb. 2020.

[17] L. Zhao *et al.*, "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[18] M. Barabas, G. Boanea, A. B. Rus, V. Dobrota, and J. Domingo-Pascual, "Evaluation of network traffic prediction based on neural networks with multi-task learning and multiresolution decomposition," in *Proc. IEEE 7th Int. Conf. Intell. Comput. Commun. Process.*, 2011, pp. 95–102.

[19] G. Mao, "Real-time network traffic prediction based on a multiscale decomposition," in *Proc. Int. Conf. Netw.*, 2005, pp. 492–499.

[20] H. Feng and Y. Shu, "Study on network traffic prediction techniques," in *Proc. Int. Conf. Wireless Commun. Netw. Mobile Comput.*, vol. 2, 2005, pp. 1041–1044.

[21] P. Cortez, M. Rio, M. Rocha, and P. N. M. de Sousa, "Internet traffic forecasting using neural networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2006, pp. 2635–2642.

[22] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Applying deep learning approaches for network traffic prediction," in *Proc. Int. Conf. Adv. Comput. Commun. Informat. (ICACCI)*, 2017, pp. 2353–2358.

[23] G. Bian, J. Liu, and W. Lin, "Internet traffic forecasting using boosting LSTM method," in *Proc. CSAE*, 2017, pp. 468–478.

[24] Q. Zhuo, Q. Li, H. Yan, and Y. Qi, "Long short-term memory neural network for network traffic prediction," in *Proc. 12th Int. Conf. Intell. Syst. Knowl. Eng. (ISKE)*, 2017, pp. 1–6.

[25] P. Babarczi, A. Pasic, J. Tapolcai, F. Németh, and B. Ladóczki, "Instantaneous recovery of unicast connections in transport networks: Routing versus coding," *Elsevier Comput. Netw.*, vol. 82, pp. 68–80, May 2015.

[26] T. Lehman *et al.*, "Multilayer networks: An architecture framework," *IEEE Commun. Mag.*, vol. 49, no. 5, pp. 122–130, May 2011.

[27] M. Calder, A. Flavel, E. Katz-Bassett, R. Mahajan, and J. Padhye, "Analyzing the performance of an anycast CDN," in *Proc. Internet Meas. Conf.*, 2015, pp. 531–537.

[28] "Cisco visual networking index: Forecast and trends, 2017–2022," San Jose, CA, USA, Cisco, White Paper,2018.

[29] N. Brownlee and K. C. Claffy, "Understanding Internet traffic streams: Dragonflies and tortoises," *IEEE Commun. Mag.*, vol. 40, no. 10, pp. 110–117, Oct. 2002.

[30] W. Allcock, J. Bresnahan, R. Kettimuthu, and M. Link, "The globus striped GridFTP framework and server," in *Proc. ACM/IEEE Conf. Supercomput.*, 2005, p. 54.

[31] A. Dorigo, P. Elmer, F. Furano, and A. Hanushevsky, "XROOTD—A highly scalable architecture for data access," *WSEAS Trans. Comput.*, vol. 1, no. 4, pp. 348–353, 2005.

[32] R. Sinha, C. Papadopoulos, and J. Heidemann, "Internet packet size distributions: Some observations," Dept. Inf. Sci. Inst., Univ. Southern California, Los Angeles, CA, USA, Rep. ISI-TR-2007-643, 2007.

[33] M. Karakus and A. Durresi, "Quality of service (QOS) in software defined networking (SDN): A survey," *J. Netw. Comput. Appl.*, vol. 80, pp. 200–218, Feb. 2017.

[34] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014.

[35] J. S. Richman, D. E. Lake, and J. R. Moorman, "Sample entropy," *Methods Enzymol.*, vol. 384, pp. 172–184, Apr. 2004.

Ferenc Mogyorósi (Member, IEEE) received the M.Sc. degree (*summa cum laude*) in electrical engineering from the Budapest University of Technology and Economics (BME), Hungary, in 2020, where he is currently pursuing the Ph.D. degree with the High-Speed Networks Laboratory, Department of Telecommunications and Media Informatics, Doctoral School of Electrical Engineering. His research interests focus on survivability in optical backbone networks, artificial intelligence, and mobile positioning.

Alija Pašić received the M.Sc. and Ph.D. degrees (*summa cum laude*) in electrical engineering from the Budapest University of Technology and Economics (BME), Hungary, in 2013 and 2019, respectively, where he is currently an Assistant Professor with the High-Speed Networks Laboratory, Department of Telecommunications and Media Informatics. His research interests focus on survivability in optical backbone networks, network coding, machine learning, artificial intelligence, and mobile positioning.

Richard Cziva received the B.Sc. degree in computer engineering from the Budapest University of Technology and Economics, Hungary, in 2013, and the Ph.D. degree in computing science from the University of Glasgow, U.K., in 2018. He is currently a Software Engineer with Energy Sciences Network, Lawrence Berkeley National Laboratory. His current research and development focuses on network data analysis using machine learning and building systems for fine-grained, and high-speed streaming network telemetry.

Péter Revisnyei received the B.Sc. degree in biology from the Eötvös Loránd University (ELTE), Hungary, in 2018, and the M.Sc. degree in biomedical engineering from the Budapest University of Technology and Economics (BME), Hungary, in 2020. His research interests focus on sport analytics and machine learning.

Zsolt Kenesi received the M.Sc. degree in electrical engineering from the Budapest University of Technology and Economics (BME), Hungary, in 1999. He works with Ericsson Hungary Ltd., Budapest. He works with Ericsson Research as a Section Manager, he lead various innovation projects in the field of network management systems, services, 4G and 5G system, and artificial intelligence.

János Tapolcai received the M.Sc. degree in technical informatics and the Ph.D. degree in computer science from the Budapest University of Technology and Economics (BME), Budapest, in 2000 and 2005, respectively, and the D.Sc. degree in engineering science from the Hungarian Academy of Sciences (MTA) in 2013. He is currently a Full Professor with the High-Speed Networks Laboratory, Department of Telecommunications and Media Informatics, BME. He has authored over 150 scientific publications. He was a recipient of several best paper awards, including ICC'06, DRCN'11, HPSR'15, and NaNa'16. He is a Winner of the MTA Lendület Program and the Google Faculty Award in 2012, and the Microsoft Azure Research Award in 2018. He is a TPC member of leading conferences, since 2012 IEEE INFOCOM, and the General Chair of ACM SIGCOMM in 2018.