Dynamic Learning-based Link Restoration in Traffic Engineering with Archie

Wenlong Ding, Hong Xu
The Chinese University of Hong Kong

***Abstract*—Fiber cuts reduce network capacity and take a long time to fix in optical wide-area networks. It is important to select the best restoration plan that minimizes throughput loss by reconfiguring wavelengths on remaining healthy fibers for affected IP links. Recent work studies optimal restoration plan or ticket selection problem in traffic engineering (TE) in a one-shot setting of only one TE interval (5 minutes). Since fiber repair often takes hours, in this work, we extend to consider restoration ticket selection with traffic dynamics over multiple intervals.**

To balance restoration performance with reconfiguration overhead, we perform dynamic ticket selection every T time steps. We propose an end-to-end learning approach to solve this T-step ticket selection problem as a classification task, combining traffic trend extraction and ticket selection in the same learning model. It uses convolution LSTM network to extract temporal and spatial features from past demand matrices to determine the ticket most likely to perform well T steps down the road, without predicting future traffic or solving any TE optimization. Trace-driven simulation shows that our new TE system, Archie, reduces over 25% throughput loss and is over 3500x faster than conventional demand prediction approach, which requires solving TE many times.

I. Introduction

Fiber cuts are serious failures in optical wide-area networks. When a fiber cut occurs, all IP links on this fiber stop carrying traffic, which causes throughput loss. The IP links can be recovered by reconfiguring to use remaining healthy fibers with Reconfigurable Optical Add Drop Multiplexers (ROADMs) [1, 9, 44]. With different fiber paths and wavelength assignments, many possible restoration plans exist for a given fiber cut scenario even in a medium-scale network [44]. Each plan essentially results in a different topology that partially restores the IP links to varying capacities. Therefore, given the traffic demand, the network operators need to determine the best restoration plan that maximizes the recovered throughput, or equivalently, minimizes the total throughput loss.

To tackle this problem, state-of-the-art solution Arrow [44] abstracts a restoration plan as a ticket. When a fiber fails, given the traffic demand, the traffic engineering (TE) system solves the TE for a specific ticket, calculates the total throughput realized, and identifies the best ticket by enumerating all candidate tickets in the same way. While this approach has been proven effective in production networks [44], we explore two new aspects that have not been examined before.

First, Arrow [44] only considers ticket selection for a single time step, i.e. one TE interval. In practice, a fiber cut usually takes hours to fix [44], and traffic exhibits significant dynamics over this period as we show with empirical evidence in §II-C. Thus, we set out to explore the problem in the long run, where dynamic ticket selection is necessary to cope with the dynamic traffic. A natural solution here is to simply re-select the ticket in every TE time step, i.e. 5 minutes [12, 15, 24]. This *frequent restoration* scheme does not work well because it overlooks the reconfiguration overhead at the optical layer. Optical wavelength reconfiguration takes at least $O(10)$ seconds with the latest hardware on a 4-node topology [44], while older components and larger topologies can take minutes [8, 18, 39]. During the reconfiguration process, the corresponding IP links are completely down [29, 35, 44], creating excessive throughput loss for the already-attenuated network. Thus, to rein in the reconfiguration overhead with frequent restoration, we propose to perform ticket selection periodically every T time steps.

Dynamic T-step ticket selection naturally motivates us to consider another new aspect: Since now a restoration ticket is to be used for T time steps, how can we select a good ticket that works well down the road? Clearly we need to depart from Arrow's one-shot myopic approach which only considers instantaneous demand, and take into account the future demand time series. Intuitively, one can predict the demand for each of the future T steps, and apply Arrow's ticket selection method using a series of T demand matrices. However, this *demand prediction* based approach entails solving the TE optimization problem TZ times in total for Z candidate restoration tickets across T steps, which results in non-negligible time overhead for the 5-minute TE budget. As we will demonstrate in §III-A, for a medium-scale topology with 16 nodes and 47 IP links, this method takes over 50 seconds to select the best ticket for future 10 time steps with 35 candidate tickets. Further, it relies on accurate prediction of traffic demand, which is challenging in itself as we try to predict a more distant future.

We take an end-to-end learning approach to design a fast and accurate ticket selection method for a T-step horizon. We view the problem as a classification task, and train a neural network with traffic history and the corresponding candidate tickets for a particular fiber cut scenario. The neural network learns which ticket is more likely to lead to the best TE performance over the next T steps, without explicitly predicting the demand or exactly solving TE. We train one model for one possible failure scenario offline, and deploy them in the TE system for online inference. Our approach enjoys faster decision time since it requires only one pass of the neural network instead

This work is supported in part by funding from the Research Grants Council of Hong Kong (GRF 11209520, CRF C7004-22G) and from CUHK (4937007, 4937008, 5501329, 5501517).

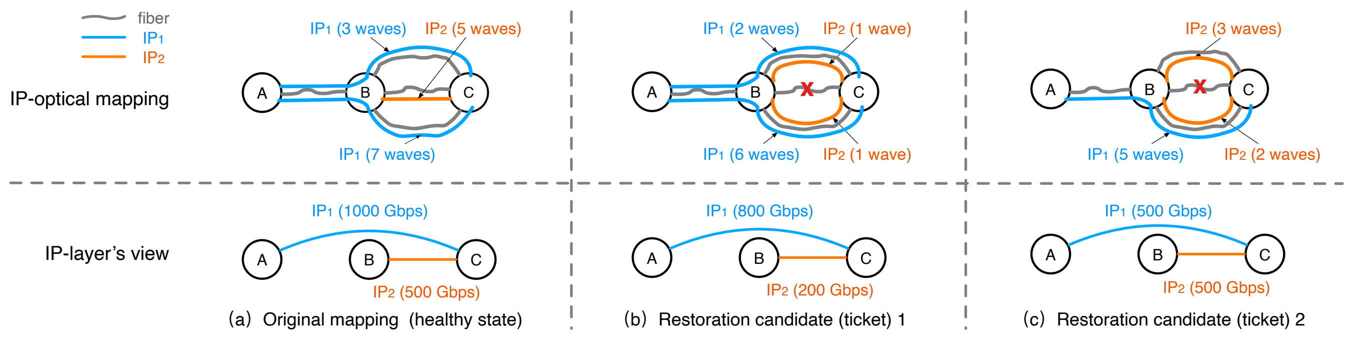

Fig. 1: IP-optical mapping and restoration tickets in optical network. (a) depicts the routing of IP links on fiber paths and wavelengths assignment on each fiber in a healthy state, with its IP-layer's view when a single wave carries 100Gbps. (b) and (c) show different partial restoration plans (tickets) to restore IP_2 when middle fiber from B to C is cut, along with their IP-layer's views, which is achieved by rerouting IP_2 on the unaffected upper and bottom fibers from B to C and reconfiguring their wavelengths. (This figure refers to Fig.7 in Arrow [44].)

of solving optimization programs many times. It is also more robust to inherent prediction errors as its end-to-end training is targeted at selecting one correct ticket (i.e. a classification label) instead of predicting all future demand.

We make several novel contributions in this paper.

- We design Archie, a TE system with dynamic restoration ticket selection to adapt to traffic dynamics during fiber cut period. Archie takes an end-to-end learning approach to periodically solve the ticket selection problem for a T-step horizon. The learning model utilizes convolutional LSTM [33] to extract temporal and spatial features of demand time series, and uses them to determine the best ticket without explicitly predicting demand or solving any TE. When a fiber cut is detected, Archie immediately handles it with the corresponding pre-trained model.
- We prototype Archie and conduct trace-driven simulation to extensively evaluate its performance. The results show that on average Archie can reduce throughput loss by 27.1% and improve decision time by 3598x compared to the strawman solution using demand prediction. It further reduces overall throughput loss by 64.7% and 59.6% compared to frequent and static restoration in the long run, respectively, which represent two extremes of the design space. Archie's performance robustness and scalability to larger topologies and traffic volumes make it suitable for use in production networks.
- We conduct analyses to understand why and how Archie's neural network helps here. We find that, spatially, Archie learns to focus on the critical flows that significantly affect overall throughput during restoration; temporally, it learns to be more attentive to traffic spikes. These features set Archie apart from the demand prediction approach which treats all flows equally and is agnostic to traffic spikes, and may offer valuable insights for better future designs of dynamic restoration ticket selection.

II. Background and Motivation

In this section, we first introduce background on restoration tickets in traffic engineering (TE), and then discuss the motivation to consider forward restoration ticket selection for TE in the long run.

A. Traffic Engineering (TE)

TE is one of the most well-studied problems in the networking community. It allocates traffic of source-destination pairs to available paths in order to optimize overall performance, mostly represented by minimizing maximum link utilization or maximizing total throughput (with or without fairness constraints) [4, 6, 24, 25, 34, 41, 44]. TE is done on the IP layer and is agnostic to optical layer details.

In our work, we mainly focus on maximizing total throughput similar to Arrow [44]. We model the IP-layer topology as a graph $G = (V, E)$, where V represents the set of routers and E the set of links. A link e has capacity C_e. We call a source-destination pair a *flow*, and the instantaneous traffic demand of flow f is D_f. For each flow, there are K candidate paths that can be used for routing. We simply use K shortest paths as the candidate paths in our work following most prior work [24, 25, 41, 44]. We use a binary variable P_e^k to encode the path-link relationship, where it equals 1 if path k traverses through link e and otherwise 0. We compute TE solution $\{w_f^k\}$ which represents the fraction of traffic on flow f assigned to path k, which can be formulated as follows:

$$\max \sum_f \sum_k w_f^k D_f \tag{1}$$

$$\text{s.t.} \sum_k w_f^k \leq 1, \forall f, \tag{2}$$

$$w_f^k \geq 0, \forall f, k, \tag{3}$$

$$\sum_f \sum_k P_e^k w_f^k D_f \leq C_e, \forall e. \tag{4}$$

Constraint (2) ensures that each flow's total routed traffic does not exceed its demand. Constraint (3) is the traffic non-negativity constraint, and (4) is the link capacity constraint.

B. Restoration Tickets in Optical Networks

IP-optical mapping. In optical wide-area networks, traffic is routed on IP links, and each link is composed of optical physical paths and occupies certain wavelengths on the fibers of these paths [37, 40, 44]. Fig. 1(a) shows how IP-optical mapping works. The network has two links, where IP_1 is routed on fiber AB with 10 wavelengths, as well as on the upper and bottom fibers of BC with 3 and 7 wavelengths, respectively,

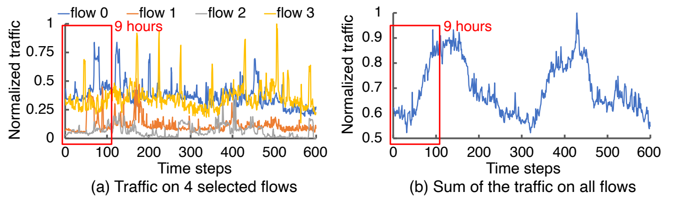

Fig. 2: Illustration of traffic dynamics in Abilene topology from SNDlib [31]. The topology has 12 nodes and 15 edges (fibers), and we present its traffic in 600 continuous time steps with a granularity of 5 minutes. (a) shows demand of four representative flows, and (b) shows the total demand across topology. Traffic data is normalized by dividing the maximum value in each sub-figure.

while IP_2 uses BC's middle fiber with 5 wavelengths. Suppose each wavelength carries 100Gbps; IP_1 and IP_2 have 1000Gbps and 500Gbps capacity in IP-layer's view, respectively.

Partial restoration and restoration ticket. When a fiber cut occurs, IP links that use this fiber lose all capacity. We can reconfigure remaining fibers and their wavelengths to partially restore these links [7, 14, 17, 21]. As shown in Fig. 1, when the middle fiber of BC is cut, IP_2 also fails immediately. We can reconfigure it to use the other two healthy fibers from B to C as shown in Fig. 1(b) and (c). This partial restoration is common in practice [44]. Notice that there are many feasible partial restoration plans with different wavelength assignments, i.e. restoration tickets, which result in different IP-level capacities.

To determine which restoration ticket is better, one has to consider the traffic demand. If the traffic demand is 700Gbps for IP_1 and 100Gbps for IP_2, ticket 1 (Fig. 1(b)) achieves a total throughput of 800Gbps but ticket 2 has 600Gbps only. Instead, if the demand is 300Gbps and 500Gbps for IP_1 and IP_2, ticket 2 delivers 800Gbps throughput and is better. Thus, it is necessary to store various restoration tickets offline for each possible fiber cut scenario so the network control plane can select the best tickets according to dynamic demands.

C. Ticket Selection for the Long Run

A recent work, Arrow [44], is the first to explore the restoration ticket selection problem in TE. It focuses on the simple one-shot scenario and uses traffic demand of one time step in its design and evaluation. However, the reality is much more complicated. Fiber cuts take a long time to physically repair: It has been reported that more than half of fiber cuts last longer than nine hours to repair for Facebook's WANs [44]. This prompts a deeper inquiry into ticket selection for the long run, which is clearly more challenging. First, traffic demand fluctuates during the fiber cut duration, making the one-shot approach in Arrow sub-optimal in the long run. As shown in Fig. 2, the total traffic demand of the entire WAN and demand of each flow (i.e. source-destination pair) in production networks show great dynamics. We quantify the traffic dynamics using the peak-to-trough ratio, which represents the ratio between the highest and lowest traffic demands. Over nine hours as in Fig. 2, the peak-to-trough ratios of four randomly selected flows are 2.60, 9.80, 6.55 and 3.24, respectively; and it stands at 1.79 for all flows. We also demonstrate experimentally in §IV-C that adopting this *static restoration* plan which utilizes the same ticket for the whole fiber cut duration results in a significant loss in TE objective.

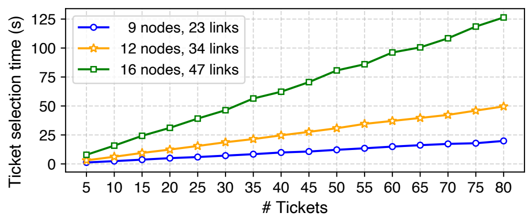

Fig. 3: Selection time (seconds) of a 10-step ($T = 10$) ticket in Demand prediction. Results for different topologies and candidate ticket numbers are shown. We solve TE optimization problem following the formulation in §II-A using the hardware environment in §IV.

One may be tempted to consider another extreme design that selects a new restoration ticket at each TE interval with instantaneous traffic demand, which we refer to as *frequent restoration*. TE is performed at 5-minute intervals in practice [4, 25, 34]. Yet, a new restoration ticket entails reconfiguring wavelengths at optical layer, which takes at least $O(10)$ seconds with the latest hardware on a simple 4-node topology, while older optical components and larger topologies take minutes [8, 18, 39, 44]. IP links under optical reconfiguration cannot carry any traffic [29, 35, 44]. This implies that frequent reconfiguration of optical network causes significant and frequent throughput loss to upper-layer workloads, which is clearly undesirable.[1]

Therefore, we propose to re-select restoration ticket every T time steps to better cope with traffic dynamic and reconfiguration overhead. T time steps (e.g., $T = 10$) is a short period compared to the whole fiber cut duration, during which traffic dynamics do not wildly change. The reconfiguration overhead can be greatly reduced as we reduce the frequency of ticket selection. Considering ticket selection for multiple time steps is a fundamentally new problem different from the one-shot selection [44], since we have to select one restoration ticket that works well for T steps with T different sets of future traffic demands and their TE plans that are both unknown to us at decision-making time.

III. Design

We present Archie's design now. We first sketch a strawman solution motivated by previous work and analyze its shortfalls. We then present Archie's end-to-end learning approach and explain the detailed design based on Convolutional LSTM.

A. Strawman Solution with Demand Prediction

An intuitive solution to our problem is to just predict the traffic demands for each of the future T time steps down the road, which extends what we do now for TE that predicts only 1-step ahead in production [24, 25]. That is, for a restoration ticket, at each step, we use the predicted demand matrix and the ticket to solve the TE problem and obtain the network throughput. Summing up the throughput across T steps, we have the total throughput corresponding to one ticket, and select the best ticket that maximizes this total throughput.

[1]A new TE plan also entails reconfiguration at IP routers, but IP links and old routing rules still work until they are de-activated right before new rules are applied [27, 43]. Thus congestion and throughput loss are much milder.

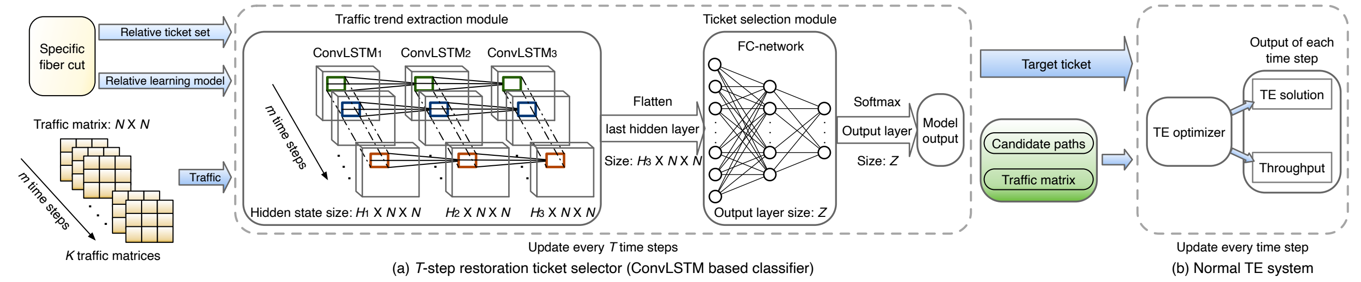

Fig. 4: Overview of Archie with T-step restoration ticket selection. Archie has different pre-trained models for different fiber cut scenarios. A fiber cut triggers Archie to use the corresponding model (ticket selector), and its output ticket to deal with current failure.

This solution has several important drawbacks. First, it requires solving TE TZ times where Z is the number of candidate tickets. Though a single instance of TE typically does not take long to solve (usually a linear program problem), scaling it by a factor TZ makes it much more significant. For example in Fig. 3, when T is set to 10, for a topology with 16 nodes and 47 IP links, it takes over 50 seconds to select the ticket given 35 candidates (recall one TE interval is 5 minutes). Larger topologies also greatly prolong the solving time as shown in the same figure. In addition, to better accommodate various demand matrices we need as many candidate tickets as possible, aggravating the time pressure.

Second, prediction entails errors even with state-of-the-art methods [5, 11, 16, 26, 28], and errors manifest as one tries to predict further away in the time horizon. The adverse effect may not be an issue for one-shot TE; as we move to consider T time steps the prediction error can actually cause salient performance drop with sub-optimal ticket selection results as we will demonstrate in §IV-B.

B. *Archie's Approach: End-to-End Learning*

We therefore take a very different approach in Archie. Instead of explicitly predicting traffic time series and solving TE, we view the ticket selection problem as a classification task, and adopt a learning approach that uses a neural network to learn which restoration ticket leads to the best performance over next T time steps given history traffic. The neural network is composed of two modules: The first is tasked to extract the features of traffic demands that are critical to ticket selection, and the second is responsible for making a selection decision. Essentially, by combining these two modules into one model with supervised learning, Archie learns what traffic features are important, how these features would look like, and then determines the potential best ticket given this information, without solving the problem explicitly with tainted input.

An analogy to better understand our idea and why it can work is the classical CNN models like AlexNet [23] and ResNet [10]. To do image classification, it is not necessary to recognize for example each body part of an animal and its exact position accurately. Instead, it is more effective and robust to just learn the relationship between features and the classification, so the model knows that "fluffy ears" and "long muzzle" can tell a dog apart from a cat. This is what Archie sets out to do. We perform in-depth analysis in §V to see what features Archie actually learns and validate our idea.

Archie's learning approach has two advantages over demand prediction based approach. First, it is much faster in decision time, requiring only one pass of the neural network instead of solving TZ optimization programs. The running time also scales more gracefully with network topology. Second, Archie is more robust to the inherent prediction inaccuracy. It takes an end-to-end approach that enables the neural network to learn how to cope with traffic uncertainties as long as the final output (ticket selection) is correct, whereas the demand prediction based approach only works when each flow's demand at each time step can be accurately predicted. For this reason, we observe that Archie is able to deliver better TE performance.

C. *Detailed Design: ConvLSTM Based Classifier*

Fig. 4 shows the overall design of Archie. As discussed in §III-B, it has two neural modules. The first uses Convolutional LSTM (ConvLSTM) [33] cells to extract both temporal and spatial traffic trends from the past traffic matrices. The output of ConvLSTM's last hidden layer goes into a fully connected neural network (FC-network) for ticket selection. These two modules are concatenated in one model and trained together.

Why ConvLSTM? ConvLSTM replaces the fully connected layers in conventional LSTM cells with convolutional layers as Fig. 4 depicts. It can explore more complex time series features compared to statistical models such as ARIMA [30] due to deep neural network's ability to model high-dimensional data. Its use of convolutional layers further allows it to explore spatial information in the data, which is ignored by conventional LSTM that models traffic matrix prediction as a multivariate time series prediction problem. ConvLSTM has shown wide success in real-world traffic matrix prediction [11, 16, 26, 28].

Training of ConvLSTM based classifier. We train one specific ConvLSTM based model for a given fiber cut scenario of the network. At time step t, we input traffic demand matrices of past m time steps $(TM_{t-m+1}, TM_{t-m+2}, \cdots, TM_t)$ into the model. The default value of m is 40 in this work. Each matrix is $N \times N$ where N is the node number in the topology. A typical ConvLSTM requires 3D input (channel, length, and width) [33], and we set channel number to 1 to enable $1 \times N \times N$ model input. Archie uses three convolutional layers in ConvLSTM as shown in Fig. 4. Layer i has a hidden state size of $H_i \times N \times N$, where H_i is a configurable parameter representing hidden dimensions. We flatten and feed hidden state of the last time step in the last convolutional layer into FC network. The output layer size of FC-network is Z, representing Z restoration tickets in current fiber cut scenario. We use Cross-Entropy loss with the true labels for training, and select the ticket with maximal probability when inferring target ticket. The label tickets are the best ticket that maximizes TE objective in next T steps which are found by brute-force search using actual demand matrices of next T steps.

D. Archie's Deployment and Workflow

We demonstrate Archie's use in practical network settings. The system has two stages: offline preparation and online TE.
Offline preparation. Following Arrow [44], we first collect restoration tickets for each possible fiber cut case. We prepare a distinct ConvLSTM based classifier for each possible failure scenario, using its corresponding ticket set for label generation and model training. Subsequently, these trained models are deployed in the system for inference only.
Online TE. Upon detection of a fiber cut, the centralized network controller identifies the affected fiber and triggers Archie to address the issue using the corresponding learning model and ticket set (as shown in Fig. 4). The online workflow of Archie is as follows: At the beginning of each T-step interval, the learned model takes traffic matrices of the past m steps as input and selects a ticket. After the ticket is configured, it does not change for this T-step interval. At each TE step, the TE optimizer mentioned in §II-A updates the TE solution using the candidate paths and instantaneous traffic demand.

IV. Evaluation

We evaluate Archie on the following aspects:

- How does Archie's ConvLSTM model perform against other ticket selection methods in one T-step time window? (§IV-B)
- How does Archie's dynamic restoration design work in the long run compared to other design choices including static restoration and frequent restoration? (§IV-C)
- How does the number of candidate restoration tickets affect the performance? (§IV-D)

The TE problems are solved with Gurobi Optimizer [2], which has been widely used in existing TE systems [25, 44]. It runs on a server with 4 Intel Xeon Platinum 8268 24-Core processors and 128GB RAM. The learning models are implemented with Pytorch [3] and trained on an Nvidia GeForce RTX 3090 GPU. The default settings of the models are as follows: for ConvLSTM, we use 3 convolutional layers with the hidden state dimensions of $H_1 = 8, H_2 = 4, H_3 = 4$ and kernel size of 3×3 for all layers. For FC-network, we use 2 fully connected layers (not including input layer) with ReLU activation in each layer; the first layer has 128 neurons and the output layer has Z neurons. For TE formulation, we use $K = 4$ shortest paths for each flow.

A. Experiment Setup

Topologies, traffic traces, and restoration tickets. We evaluate Archie on four topologies as shown in Table I. Abilene is a real-world backbone network topology commonly used in existing work [8, 24, 25], and we obtain its 3000 continuous

Topology	# Nodes	# Fibers	# IP links	# Traffic matrices (Train + Test)
Arpanet	9	10	23	2880+120
Abilene	12	15	34	
Airtel	16	26	47	
GRnet	37	42	101	

TABLE I: Network topologies used in evaluations.

traffic matrices in 5-minute intervals from SNDLib [31]. Arpanet, Airtel, and GRnet are three real-world optical topologies obtained from Internet Topology Zoo [19]. We generate realistic traffic matrices using the gravity model in YATES [24] based on the traffic distribution in Abilene with the same 5-minute granularity. There are many versions of Arpanet in Internet Zoo Topology; we use the version with 9 nodes.

Following Arrow [44], we generate IP-layer topologies from the optical ones following the distributions of the number of IP links per fiber and the number of wavelengths per IP link in Facebook. We use traffic matrices of the first 2880 time steps for model training, and the rest for testing. Note that 120-step testing data corresponds to 10 hours which is similar to the duration to fix a fiber cut in practice [44]. Since multiple fiber failures are extremely rare in practice [32], we only consider single fiber failures randomly generated across the optical topology in our evaluation. The process of generating restoration tickets is as follows: 1) We identify four shortest healthy fiber paths with the same endpoints as the failed fiber, and these fiber paths are used to restore the failed IP links. 2) We generate Z candidate tickets by enumerating the possible combinations of wavelength reconfiguration on the four identified restoration paths. We set the default value of Z to 30 unless otherwise specified.
Schemes compared. We compare Archie against the following T-step restoration ticket selection schemes:

- *One-shot myopic*: It selects the ticket that maximizes the throughput of the instantaneous demand matrix at the start of current T-step window, and uses it for next T steps.
- *Demand prediction*: The strawman solution described in §III-A that predicts demand for each of the T steps, and uses the TE results with predicted traffic to select the best ticket. We use ConvLSTM with the same training dataset and model settings as Archie to predict future traffic to ensure a fair comparison.
- *Optimal*: It uses the actual traffic of next T steps to select the best ticket that maximizes total throughput. This serves as the upper bound for performance comparison.

Lastly, we apply *demand scaling* on the traffic traces to fully evaluate Archie under different traffic loads.

B. Performance of ConvLSTM-based Ticket Selection

In this section, we focus on various ticket selection methods under one 10-step time window, by evaluating overall *throughput loss* which is defined as the fraction of total throughput loss relative to total demand. We do not consider reconfiguration overhead which is the same across the board. Its effect is considered in §IV-C when we evaluate the impact of T.

We observe that the overall performance of Archie is the best among all comparison schemes as shown in Fig. 5. Archie

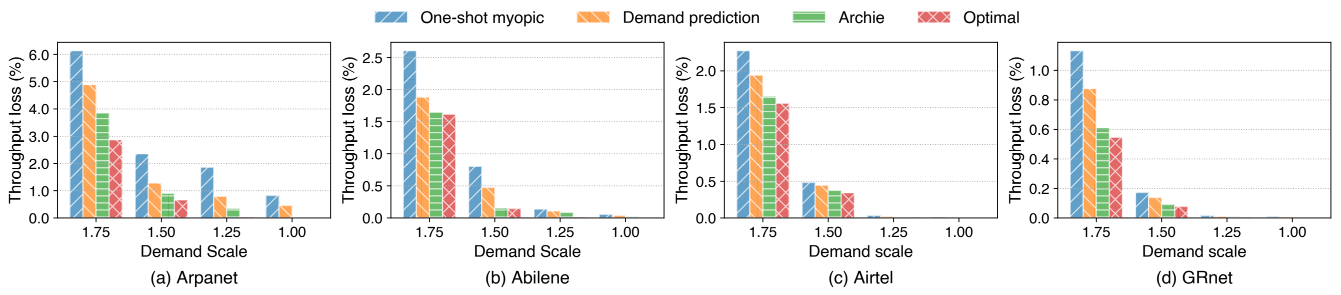

Fig. 5: Comparison of throughput loss for different restoration ticket selection methods in T-step window under various demand scales. (T is fixed to be 10).

Topology	T	Ticket selection time (seconds)		
		One-shot myopic	Demand prediction	Archie
Arpanet	5	0.729	3.644	0.01875
	10	0.721	7.102	0.01900
	15	0.732	10.982	0.01894
Abilene	5	1.845	9.221	0.02544
	10	1.852	18.001	0.02501
	15	1.847	27.705	0.02539
Airtel	5	4.746	23.712	0.03011
	10	4.745	48.003	0.02996
	15	4.751	71.265	0.03008
GRnet	5	51.974	259.182	0.04355
	10	52.123	522.313	0.04441
	15	51.988	779.320	0.04483

TABLE II: Single ticket selection time for different schemes. We show the average values on the testing set with 120 traffic matrices, involving an average of 24 runs for $T = 5$, and 12 and 8 runs for $T = 10$ and $T = 15$, respectively.

reduces the throughput loss by 48.5% and 27.1%, compared to One-shot myopic and Demand prediction, respectively, averaged over all topologies and demand scales. Archie is able to offer close-to-optimal performance with additional throughput loss ranging from 0% to 0.98% at most.

Second, evaluation results reveal that Archie has a more pronounced impact at a larger demand scale compared to other methods. In terms of additional throughput loss relative to Archie averaged over four topologies, when demand scales are 1.25, 1.5, and 1.75, One-shot myopic suffers from 4.95x, 2.59x, and 1.81x more additional loss compared with the loss when demand scale is 1.0, and Demand prediction suffers from 3.82x, 1.72x, and 1.01x more. For a specific example in Arpanet, One-shot myopic has 2.28% additional loss when scale is 1.75, compared to 0.83% at original demand. When demand is small, ticket selection plays a minor role since there are many tickets that can give good or even optimal throughput performance. As demand scales up, good restoration tickets become more difficult to identify and Archie is more capable of finding them compared to other schemes.

Table II further shows the running time comparison of different ticket selection schemes with different values of T. Archie greatly reduces the running time: on average across all topologies and settings, its speedup relative to One-shot myopic and Demand prediction stands at 362x and 3598x, respectively. Notice that Archie almost always takes less than 45ms to select a ticket. Second, as T increases from 5 to 15, Demand prediction's running time quickly escalates because of more runs of TE optimizer, but Archie's time remains steady by design. Third, Archie's running time has a slower growth speed as topology size increases compared to other methods. For example, in the largest GRnet with $T = 10$, Archie's running time is only 2.3x that of the smallest Arpanet, whereas One-shot myopic and Demand prediction take 72.3x and 73.5x more time, respectively.

C. *Benefit of Dynamic Link Restoration*

To validate the design choice of dynamically updating the link restoration plan, here we compare Archie against the two extreme design choices discussed in §II-C: The first is *static restoration* that selects one ticket and deploys it throughout the whole fiber cut duration, which in our traces corresponds to setting T to 120, the length of the test dataset. The second is to re-select the ticket at each TE step, i.e. $T = 1$, and we refer to this as *frequent restoration*. We obtain the cumulative throughput over 120 steps, and model the restoration overhead by setting the reconfiguration time at the optical layer to be $\frac{1}{30}$ time of one step (10 seconds). During this period, no traffic can be delivered on the IP links traversing these failed optical fibers. Our setting is in line with prior work [44] which shows that reconfiguration time is $O(10)$ seconds.

We observe from Fig. 6 that the extreme methods lead to more throughput loss than Archie with other settings of T (5, 10, 15). On average, Archie under the best setting of T can reduce throughput loss by 64.7% and 59.6% compared to frequent and static restoration, respectively. The poor performance of the two extreme methods justifies our choice to use a moderate T to practically reap the benefit of dynamic restoration without excessive overheads from frequent reconfiguration and traffic dynamics.

Second, the throughput loss of static restoration grows faster than frequent restoration with demand scaling. When demand scale is 1, frequent and static restoration has an average throughput loss of 0.94% and 0.28%, respectively. Frequent restoration has a higher loss at this scale because of reconfiguration overhead. However, when demand scales are 1.25, 1.5, and 1.75, frequent restoration's throughput loss only increases by 1.18x, 1.62x, and 3.24x, respectively; while for static restoration, it increases by 2.30x, 4.85x, and 12.32x, respectively. As demand scale becomes larger, static restoration will incur more throughput loss than frequent restoration.

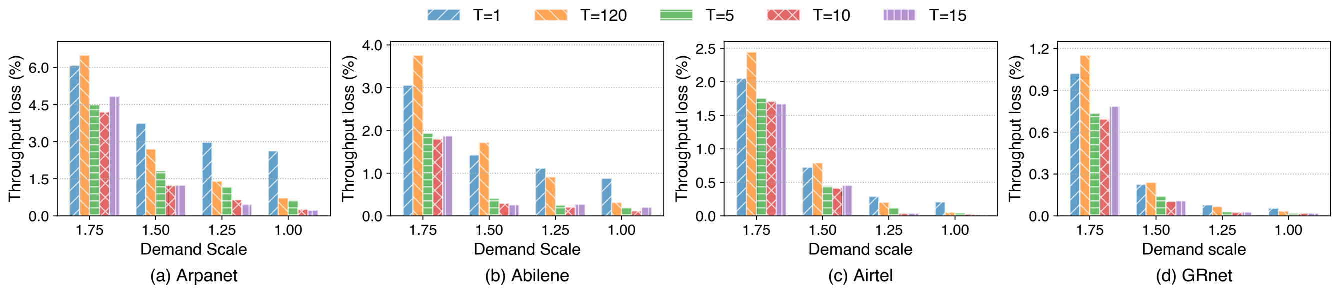

Fig. 6: Comparison of throughput loss in the long run (120 time steps) considering reconfiguration overhead. The restoration methods evaluated include static restoration ($T = 120$) and frequent restoration ($T = 1$) along with other settings of T using Archie ($T = 5, 10, 15$).

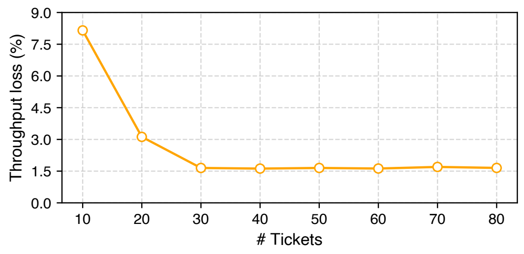

Fig. 7: Throughput loss with different numbers of tickets for one fiber cut scenario. We evaluate it on Abilene with 1.75x demand scale using Archie.

For example, when the scale is 1.75, the average throughput loss is 3.46% for static restoration compared to 3.05% for frequent restoration. This demonstrates that dynamic restoration schemes outperform static restoration even with unnecessarily excessive configuration overheads in heavy traffic situations.

Third, Archie with three moderate settings of T (5, 10, 15) has small performance difference. The difference in throughput loss between the best and worst settings is only 0.2% on average among all topologies and demand scales.

D. *Impact of Number of Candidate Tickets*

We show throughput loss under different numbers of candidate tickets in Abilene with a demand scale of 1.75x in Fig. 7. When the ticket number is small ($\leq$30), throughput loss is relatively large because the candidate ticket set is not large enough to cover a good ticket. At this stage, adding new candidate tickets greatly reduces the throughput loss. However, as the number of candidates exceeds 30, throughput loss stops improving obviously. Notice that with more candidate tickets, we need more time to calculate the labels for training (the TE optimizer needs to be called TZ times to obtain one label), and it requires more cost to train the learning model. Thus, we elect to use 30 candidate tickets in our evaluation, considering its good performance and relatively low model preparation cost. Network operators should use a proper number of candidate tickets according to their network scale and demand dynamics.

V. Analysis

Our evaluation has shown Archie's superior performance. However, as deep learning models act as a black box to a large extend, we wish to gain a better understanding of the use of ConvLSTM in Archie, particularly the insights it "learns" for dynamic ticket selection. In this section, we seek to answer these questions from two angles: (1) Spatially, does Archie pay more attention to specific flows that are more important to ticket selection? (§V-A). (2) Temporally, for a given flow, does Archie identify any special traffic patterns that are more impactful to ticket selection? (§V-B).

In this section we use two small topologies in Fig. 8 for simplicity of analysis, and generate five candidate tickets for each topology using the method in §IV-A. We use the demands of flows between the first four and six nodes of the traffic trace in Abilene topology for G_1 and G_2, respectively. The trace contains 3000 steps. The first 70% of the trace is used to train Archie and Demand prediction method, and the rest for testing. We use $K = 1$ shortest path for TE and set $T = 10$. Other settings remain the same as in §IV unless specified otherwise.

A. *Spatial Features*

To explore spatial features that are important to ticket selection, we employ a DNN analysis method known as *occlusion analysis* [42]. This method involves occluding part of the model input and comparing the output differences before and after the occlusion. The greater the gap, the more important the occluded part is. *Occlusion analysis* has been widely used in many image classification tasks using CNN [13, 20, 22, 42].

We apply occlusion analysis to Archie's ConvLSTM model by making the history demand inputs of specific flows to be zero, as shown in Fig. 8. For brevity, we focus on the two most frequently selected tickets from the 900-step testing trace data. We identify the corresponding ticket selection epochs that lead Archie to select these two representative tickets, and leverage them to conduct occlusion analysis. For each of these identified epochs, we enumerate each restored IP link, occlude all flows traversing this link, and obtain a new ticket using the same ConvLSTM model (trained with complete training data). We then calculate the normalized throughput loss due to occlusion with this new ticket. Table III provides the wavelength information for the links associated with the two tickets of the two topologies.

Fig. 9 shows the results of our occlusion analysis. Observe that the higher the throughput loss is for an occluded link, the larger its capacity becomes after restoration with Archie. For example, for ticket$_2$ in G_1, the throughput loss of occluding the three links is 1.56:1.28:1.01, while their restored capacity under this ticket is 3:2:1 (wavelengths have the same capacity). This implies that Archie is able to identify flows, and in turn

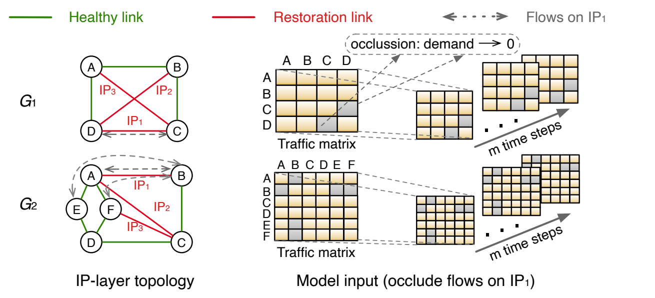

Fig. 8: Illustration of two evaluated IP-layer topologies and occlusion analysis method. The figure depicts certain fiber cut scenarios with restoration links in red (i.e., IP_1, IP_2, IP_3) and healthy links in green. The method of occluding flows on restoration links is also shown. When occluding flows on IP_1 in G_1, demand of bidirectional flow C to D becomes 0 for the m-step input.

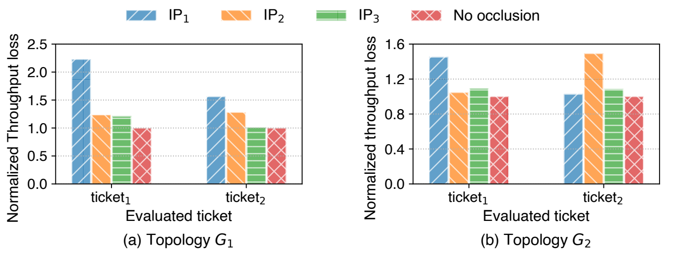

Fig. 9: Throughput loss comparison of occluding the flows on different restoration links. No occlusion means we do not occlude any flow in Archie's input. The results are normalized to the case of no occlusion.

IP links, that are more important to restore, and pick tickets that recover them maximally.

To verify if Archie's capability to identify critical flows can actually help ticket selection, we add this feature into Demand prediction. Specifically, we train new prediction models that pay more attention to critical flows identified by Archie, select new tickets based on newly predicted traffic, and calculate their throughput loss. We achieve this by modifying the MSE loss utilized in training the prediction model which has been used in many traffic prediction works [5, 11, 28, 38]. Formally,

$$J = \frac{1}{T \times F} \sum_{t}^{T} \sum_{f}^{F} \alpha_f (y_{ft} - h_{ft})^2,$$

where J is the MSE loss, F is the number of flows, y_{ft} and h_{ft} are the ground truth and predicted demand of flow f at future time step t, and α_f is the assigned weight for flow f. When $\alpha_f = 1$ for all flows, it is traditional MSE. We now train the new prediction models using weighted MSE loss with weights determined by the information in Table III. For IP links in each representative ticket, more wavelengths after restoration correspond to the higher importance of the flows traversing it, and thus, we allocate more weight to these flows. We calculate the throughput loss at the identified epochs that correspond to each representative ticket. Fig. 10 demonstrates that with the weighted MSE loss, Demand prediction can also achieve a reduction in throughput loss.

Note that it is difficult to determine which flows are important, or what weights should be set to these flows that lead to the best ticket selection. Archie, through training with past traffic and labeled tickets, seems to be good at solving these questions which attributes to improved performance.

Topology	Information item	ticket_1			ticket_2		
		IP_1	IP_2	IP_3	IP_1	IP_2	IP_3
G_1	# Wavelengths	4	1	1	3	2	1
	Weights_1 (α_f)	2.0	1.0	1.0	2.0	1.5	1.0
	Weights_2 (α_f)	3.0	1.0	1.0	3.0	2.0	1.0
G_2	# Wavelengths	8	2	2	2	8	2
	Weights_1 (α_f)	2.0	1.0	1.0	1.0	2.0	1.0
	Weights_2 (α_f)	3.0	1.0	1.0	1.0	3.0	1.0

TABLE III: Wavelengths on restoration IP links and weights settings for weighted MSE loss in Demand prediction. α_f of the restoration link is set to all flows traversing this link. For ticket in each topology, we use two weight sets based on the capacity (wavelengths) on links, as suggested by Archie.

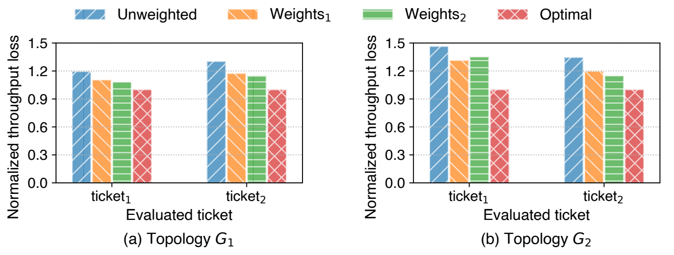

Fig. 10: Throughput loss comparison using various weight settings in Demand prediction, with the results normalized to Archie. Two weight settings for each ticket in each topology are shown in Table III. The unweighted method refers to the original Demand prediction approach with traditional MSE loss.

B. Temporal Traffic Patterns

Another dimension we explore is temporal patterns utilized by Archie to determine the best ticket. We use LSTMvis [36], an LSTM visualization technique that examines hidden states at each time step along with corresponding input and counts the number of hidden states that exceed a predetermined threshold within an input interval comprising multiple time steps. We consider a input feature more important if there are more corresponding hidden states exceeding the threshold.

We analyze hidden states of all ConvLSTM layers for each flow along with time in Archie.[2] We also do the same to the ConvLSTM model when it is trained only for demand prediction (without the FC network for ticket selection). In Fig. 11, we see that Archie reacts more violently to the traffic spikes in the trace and there are 3 hidden states exceeding the threshold 0.5 between time steps 25 and 30 (the same interval of the traffic spike), while no significant changes are observed in other intervals. Demand prediction stays more stable throughout the whole trace and no hidden states exceed the threshold in all times. Therefore, Archie focuses more on the demand spikes in the traffic trace, but Demand prediction tends to capture the average-case over the long term.

To see if this feature actually helps restoration ticket selection, we directly use the learned parameters of the ConvLSTM module in Demand prediction for Archie, freeze this part of the neural network, and re-train Archie's FC network. We refer to this new model as Hybrid in our work. Fig. 12 shows that Hybrid's throughput loss is always between Archie and Demand prediction. Across two tested topologies, Hybrid's

[2]To ensure that the hidden state's information comes from the same flow, we specifically use a 1x1 convolution kernel in §V-B. The position of a flow's hidden state in its hidden state matrix is the same place as the flow in its traffic matrix. We have 16 hidden states for one flow in total because we have $H_1 = 8, H_2 = 4, H_3 = 4$ hidden states in each ConvLSTM layer.

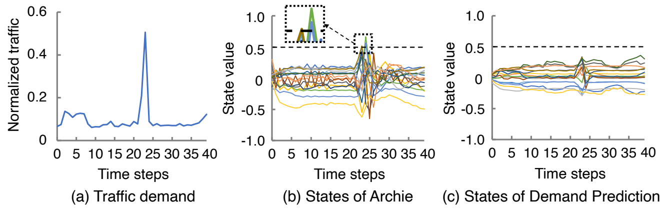

Fig. 11: Archie focuses on traffic spikes: (a) shows an example flow in G_1 with a spike between time step 20 and 25; (b) shows Archie detects the spike with 3 hidden states exceeding the threshold (0.5) in that interval, while (c) shows that no hidden states of Demand prediction exceed the threshold.

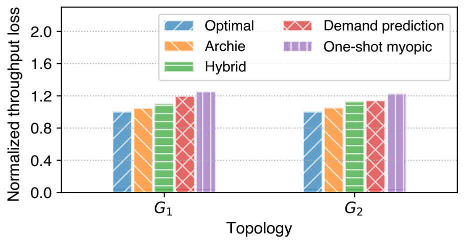

Fig. 12: Comparison of normalized throughput loss between various methods in two topologies when exploring temporal features. We normalize the loss to Optimal, with Hybrid's loss lying between Archie and Demand prediction.

throughput loss is 1.05x and 0.93x that of Archie and Demand prediction on average, respectively. This implies that Archie's ability to focus on traffic spikes is one reason it performs better than Demand prediction. The superior performance of Hybrid compared to Demand prediction indicates that Archie's end-to-end classification model provides other advantages for ticket selection beyond focusing on demand spikes (such as identifying important flows mentioned above in §V-A).

VI. Offline Training Cost

Archie's current design requires a unique model for each fiber cut scenario due to their unique candidate ticket sets that result in generating different label tickets. Although offline model preparation does not affect the online TE performance as we have shown, it still seems computationally intensive and time-consuming, especially when repeated label generation and model training are required for different scenarios. In this section, we discuss two aspects of offline model training cost: (1) Does the current design have acceptable offline training time? (2) What are the possible future ways to improve the efficiency of the offline training process?

To create one model of Archie, we need to generate label tickets for all training traffic using TE optimizers, then train ConvLSTM on GPU. Table IV shows the average time required to generate one label and train one ConvLSTM across different topologies using the hardware in §IV-A. Recall that generating one label ticket entails calling the TE optimizer TZ times. This process takes less than half a minute for a medium-scale topology such as Abilene, and several minutes for larger topologies such as GRnet. To achieve good performance as shown in §IV, we require $O(1000)$ label tickets for one model, which can be generated in parallel, reducing the generation time to $O(1000TZ/P)$, where P is the maximum number of parallel threads. Once label generation is complete, it takes another $O(10)$ minutes to train one ConvLSTM model.

Topology	Arpanet	Abilene	Airtel	GRnet
Label generation (minutes)	0.121	0.308	0.791	8.662
Model training (minutes)	17.34	19.54	22.74	32.20

TABLE IV: Time to generate one label ticket and train one learning model in different topologies. We set $T = 10$ and $Z = 30$ and train the model for 100 epochs. The results here are averaged over 500 repetitions for label generation time, and 5 repetitions for model training time.

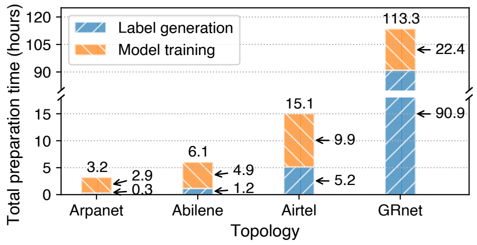

Fig. 13: Total time to prepare all ConvLSTM models for all single fiber failures. We evaluate the time using one single server machine supporting 192 maximum concurrent threads for parallel label generation and one GPU for model training with the same hardware environment as indicated in IV.

The repeated process above should be conducted for all highly probable fiber cut scenarios in practice. We present the practical total time in Fig. 13 to obtain all models for all single fiber failures with the training set in §IV. Note that our experiment server machine supports $P = 192$ concurrent threads for parallel label generation. The results demonstrate that for medium topologies, such as Abilene, the entire offline training process can complete in about six hours, while for larger topologies like GRnet, it may take around five days. We believe this offline time investment is acceptable for real-world use. Additionally, our time evaluation is based on a single server machine with a single GPU. Network operators can leverage more powerful resources to prepare the models in parallel and further reduce the offline training time.

We can also seek more advanced designs in future to reduce Archie's offline preparation cost. First, one may explore other learning methods such as unsupervised learning to decrease label generation time involved in solving complex TE programs. Second, improving the generalization ability of the models can alleviate or even eliminate the need for repeated training for different settings. For instance, we can design models capable of handling different fiber cut scenarios or use transfer learning to avoid repeated training starting from scratch.

VII. Conclusion

We present Archie, a novel TE system with T-step restoration ticket selection, targeting at dealing with dynamic link restoration in a long run. We focus on the restoration ticket selection problem, propose an end-to-end learning approach based on ConvLSTM to extract temporal and spatial features from past demand matrices, and select a ticket that is most likely to perform well for future demands across the T-step horizon. Extensive trace-driven simulation shows that Archie's learning approach is fast without having to solve the TE optimization a number of times, and is also effective in picking a good ticket without explicitly or accurately forecasting future traffic. Robust online performance and good scalability make Archie a competitive solution for practical network use.

References

[1] Adva ROADM. https://www.adva.com/en/products/technology/roadm.

[2] Gurobi Optimizer. http://www.gurobi.com.

[3] Pytorch. https://pytorch.org/.

[4] Firas Abuzaid, Srikanth Kandula, Behnaz Arzani, Ishai Menache, Matei Zaharia, and Peter Bailis. Contracting Wide-area Network Topologies to Solve Flow Problems Quickly. In *Proc. USENIX NSDI*, 2021.

[5] Abdelhadi Azzouni and Guy Pujolle. NeuTM: A Neural Network-based Framework for Traffic Matrix Prediction in SDN. In *Proceedings of IEEE/IFIP Network Operations and Management Symposium*, 2018.

[6] Jeremy Bogle, Nikhil Bhatia, Manya Ghobadi, Ishai Menache, Nikolaj Bjørner, Asaf Valadarsky, and Michael Schapira. TEAVAR: Striking the Right Utilization-Availability Balance in WAN Traffic Engineering. In *Proc. ACM SIGCOMM*. 2019.

[7] Bharat T Doshi, Subrahmanyam Dravida, P Harshavardhana, Oded Hauser, and Yufei Wang. Optical Network Design and Restoration. *Bell Labs Technical Journal*, 4(1):58–84, Aug. 2002.

[8] Thomas Fenz, Klaus-Tycho Foerster, and Stefan Schmid. On Efficient Oblivious Wavelength Assignments for Programmable Wide-Area Topologies. In *Proceedings of the Symposium on Architectures for Networking and Communications Systems*, 2021.

[9] Klaus-Tycho Foerster, Long Luo, and Manya Ghobadi. Optflow: A Flow-Based Abstraction for Programmable Topologies. In *Proceedings of the Symposium on SDN Research*, 2020.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[11] Zhixiang He, Chi-Yin Chow, and Jia-Dong Zhang. STCNN: A Spatio-temporal Convolutional Neural Network for Long-Term Traffic Prediction. In *Proceedings of IEEE International Conference on Mobile Data Management*, 2019.

[12] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. Achieving High Utilization with Software-Driven WAN. In *Proc. ACM SIGCOMM*, 2013.

[13] Cosimo Ieracitano, Nadia Mammone, Amir Hussain, and Francesco Carlo Morabito. A Novel Explainable Machine Learning Approach for EEG-Based Brain-Computer Interface Systems. *Neural Computing and Applications*, 34(14):11347–11360, Jul. 2022.

[14] Rainer R Iraschko and Wayne D Grover. A Highly Efficient Path-restoration Protocol for Management of Optical Network Transport Integrity. *IEEE Journal on Selected Areas in Communications*, 18(5):779–794, Jun. 2000.

[15] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Hölzle, Stephen Stuart, and Amin Vahdat. B4: Experience with a Globally-Deployed Software Defined WAN. In *Proc. ACM SIGCOMM*, 2013.

[16] Weiwei Jiang. Internet Traffic Matrix Prediction with Convolutional LSTM Neural Network. *Internet Technology Letters*, 5(2):e332, Sep. 2021.

[17] Xin Jin, Yiran Li, Da Wei, Siming Li, Jie Gao, Lei Xu, Guangzhi Li, Wei Xu, and Jennifer Rexford. Optimizing Bulk Transfers with Software-defined Optical WAN. In *Proc. ACM SIGCOMM*, 2016.

[18] Takuya Kanai, Yumiko Senoo, Kota Asaka, Jun Sugawa, Hideaki Tamai, Hiroyuki Saito, Naoki Minato, Atsushi Oguri, Seiya Sumita, Takehiro Sato, et al. Novel Automatic Service Restoration Technique by Using Self-Reconfiguration of Network Resources for A Disaster-Struck Metro-access Network. *Journal of Lightwave Technology*, 36(8):1516–1523, Jan. 2018.

[19] Simon Knight, Hung X Nguyen, Nickolas Falkner, Rhys Bowden, and Matthew Roughan. The Internet Topology Zoo. *IEEE Journal on Selected Areas in Communications*, 29(9):1765–1775, Sep. 2011.

[20] Muhammed Kocabas, Chun-Hao P Huang, Otmar Hilliges, and Michael J Black. PARE: Part Attention Regressor for 3D Human Body Estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[21] Adil Kodian and Wayne D Grover. Failure-Independent Path-Protecting P-Cycles: Efficient and Simple Fully Preconnected Optical-Path Protection. *Journal of Lightwave Technology*, 23(10):3241, Oct. 2005.

[22] Caroline König and Ahmed Mohamed Helmi. Sensitivity Analysis of Sensors in A Hydraulic Condition Monitoring System Using CNN Models. *Sensors*, 20(11):3307, Jun. 2020.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, 2012.

[24] Praveen Kumar, Chris Yu, Yang Yuan, Nate Foster, Robert Kleinberg, and Robert Soulé. YATES: Rapid Prototyping for Traffic Engineering Systems. In *Proceedings of the Symposium on SDN Research*, 2018.

[25] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, Chiun Lin Lim, and Robert Soulé. Semi-Oblivious Traffic Engineering: The Road Not Taken. In *Proc. USENIX NSDI*, 2018.

[26] Phi Le Nguyen, Yusheng Ji, et al. Deep convolutional LSTM Network-Based Traffic Matrix Prediction with Partial Information. In *Proceedings of IFIP/IEEE Symposium on Integrated Network and Service Management*, 2019.

[27] Hongqiang Liu, Xin Wu, Ming Zhang, Lihua Yuan, Roger Wattenhofer, and David Maltz. zUpdate: Updating Data Center Networks With Zero Loss. In *Proc. ACM SIGCOMM*, 2013.

[28] Qingyao Liu, Jianwu Li, and Zhaoming Lu. ST-Tran: Spatial-Temporal Transformer for Cellular Traffic Prediction. *IEEE Communications Letters*, 25(10):3325–3329, Jul. 2021.

[29] Athina Markopoulou, Gianluca Iannaccone, Supratik Bhattacharyya, Chen-Nee Chuah, and Christophe Diot. Characterization of Failures in An IP Backbone. In *Proc. IEEE INFOCOM*, 2004.

[30] Brian K Nelson. Time Series Analysis Using Autoregressive Integrated Moving Average (ARIMA) Models. *Academic Emergency Medicine*, 5(7):739–744, Jun. 2008.

[31] Sebastian Orlowski, Roland Wessäly, Michal Pióro, and Artur Tomaszewski. SNDlib 1.0: Survivable Network Design Library. *Networks: An International Journal*, 55(3):276–286, May 2010.

[32] Sumathi Ramamurthy, Laxman Sahasrabuddhe, and Biswanath Mukherjee. Survivable WDM Mesh Networks. *Journal of Lightwave Technology*, 21(4):870–883, Apr. 2003.

[33] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Advances in Neural Information Processing Systems*, 2015.

[34] Rachee Singh, Sharad Agarwal, Matt Calder, and Paramvir Bahl. Cost-effective Cloud Edge Traffic Engineering with Cascara. In *Proc. USENIX NSDI*, 2021.

[35] Rachee Singh, Manya Ghobadi, Klaus-Tycho Foerster, Mark Filer, and Phillipa Gill. RADWAN: Rate Adaptive Wide Area Network. In *Proc. ACM SIGCOMM*, 2018.

[36] Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M Rush. LSTMVis: A Tool for Visual Analysis of Hidden State Dynamics in Recurrent Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):667–676, Aug. 2017.

[37] Massimo Tornatore, Guido Maier, and Achille Pattavina. Availability Design of Optical Transport Networks. *IEEE Journal on Selected Areas in Communications*, 23(8):1520–1532, Aug. 2005.

[38] R Vinayakumar, KP Soman, and Prabaharan Poornachandran. Applying Deep Learning Approaches for Network Traffic Prediction. In *Proceedings of International Conference on Advances in Computing, Communications and Informatics*, 2017.

[39] Ann Von Lehmen, Robert Doverspike, George Clapp, Douglas M Freimuth, Joel Gannett, Aleksandar Kolarov, Haim Kobrinski, Christian Makaya, Emmanuil Mavrogiorgis, Jorge Pastor, et al. CORONET: Testbeds, Demonstration, and Lessons Learned. *Journal of Optical Communications and Networking*, 7(3):A447–A458, Mar. 2015.

[40] Tiejun J Xia, Steven Gringeri, and Masahito Tomizawa. High-capacity Optical Transport Networks. *IEEE Communications Magazine*, 50(11):170–178, Nov. 2012.

[41] Zhiyuan Xu, Jian Tang, Jingsong Meng, Weiyi Zhang, Yanzhi Wang, Chi Harold Liu, and Dejun Yang. Experience-Driven Networking: A Deep Reinforcement Learning Based Approach. In *Proc. IEEE INFOCOM*, 2018.

[42] Matthew D Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *Proceedings of European Conference on Computer Vision*, 2014.

[43] Jiaqi Zheng, Hong Xu, Guihai Chen, and Haipeng Dai. Minimizing Transient Congestion during Network Update in Data Centers. In *Proc. IEEE ICNP*, 2015.

[44] Zhizhen Zhong, Manya Ghobadi, Alaa Khaddaj, Jonathan Leach, Yiting Xia, and Ying Zhang. ARROW: Restoration-Aware Traffic Engineering. In *Proc. ACM SIGCOMM*, 2021.