# Leveraging Spatiotemporal Relations for Predicting Potential Link Failures

Bruck Wubete, Babak Esfandiari, Thomas Kunz
Systems and Computer Engineering
Carleton University
Ottawa, Canada
bruckwendwessenwubet@cmail.carleton.ca, babak@sce.carleton.ca, tkunz@sce.carleton.ca

***Abstract*— Being able to predict link failures in advance would be of great benefit to network operators. We use Machine Learning (ML) techniques to extract temporal and spatial relations from real network data and use them to predict link failures. We use Interior Gateway Protocol (IGP) configuration changes as a guide to achieve this. We predict link failures in the next five days based on data collected from the previous five days. We propose a modified Variational Auto Encoder (VAE) model to compress the higher dimensional dataset into a latent space that captures time-based relations in the data. We demonstrate that five days is the smallest look-back window of time required to get satisfactory prediction results. Using feature importance plots, we learned that the VAE model was able to capture intricate time-based dependencies in the error counter features to achieve good performance. In addition, using a Graph Convolutional Network (GCN), we were able to aggregate data from neighboring links to improve the model's performance. Neighbors up to two hops away carried relevant information in IGP metric settings and in traffic metric counter features. The relevance of the correlation of the features in time and space is confirmed using standard feature importance wrapper methods. Finally, by combining the VAE and GCN components, we were able to extract spatial and temporal features in conjunction, leading to further improvements. These ML approaches significantly improve existing manual methods of tracking metrics* in time and space currently followed by the operator.**

***Keywords*- IGP; Transformers; VAE; GCN; Machine Learning; Feature Importance; Time Series; Graphs***

## I. Introduction

In this work, we focus on predicting network failures guided by Interior Gateway Protocol (IGP) configuration changes. Links that exhibit frequent IGP configuration changes are called "flapping links". Flapping links go down multiple times a day, causing reroutes and, ultimately, noticeable delays by the users. To avoid such reroutes, the operator sets high IGP metrics. This results in a purposely high cost for the links, which causes the networking devices to pick other less costly paths, effectively diverting traffic away from the flapping links [1]. Once the issue is resolved, the operator resets the IGP configuration metrics to reopen traffic flow through the link. This cycle is frequently repeated for flapping links. As such, we use IGP configuration changes as a guide for detecting network failures caused by flapping links.

Although most of the data normally collected during link failure resolutions is time and space-dependent, limited time series datasets exist for exploring spatiotemporal characteristics in the network management field. By using data from a real production network, this study expands on the limitations of other studies available in the literature that often use either simulated environments or topologies with simple connections. This is observed in optical [2] and IP [3] network studies. There are some existing studies, like [4], that use time series forecasting algorithms and deep learning models for network traffic prediction. However, the success of these models is yet to be validated in real scenarios at a scale representative of a real service provider. This paper finds useful insights from spatiotemporal relations of metrics reported from the IP and the underlying optical layer. The fact that the datasets are collected from an actual production network means that it comes with a set of non-trivial challenges like data size, data drift, and lack of samples for the link failure scenarios we are interested in.

## II. Background

The data in this study is collected from a 400Gbps optimized intelligent network interconnecting thousands of data centers and carriers worldwide, spreading out over four continents and 32 countries. With over 150K Kms of fiber coverage, it comprises thousands of IP and optical devices from different vendors. The primary data sources are OneControl, Cisco WAN Automation Engine (WAE), and IOS XR Traffic Controller (XTC): WAE collects data for all IP interfaces, while OneControl collects data for Ethernet and Optical Interfaces on the optical devices. XTC is the controller that reports the IGP configuration metrics of interfaces on the IP layer.

In a previous study [1], the features in the five day window were flattened into a single row, resulting in five times more feature columns. This was done to conform the shape of the input data into two dimensions, as required by the ML model we used. However, this method erodes the dependencies in the dataset in the time and space axis. Hence, the models used in the previous study did not leverage the spatiotemporal based dependencies in the intrinsic nature of the data. In this study, we explore different approaches to leverage spatiotemporal dependencies of the data. These include XGBoost [5] with time-based difference features and using out-of-the-box time-series classification models like ROCKETS (RandOm Convolutional KErnel Transform) [6], Transformers [7], and Variational Auto Encoders (VAE) [8]. A Graph Convolutional Network (GCN) is used for spatial feature extraction [9].

## III. Related Work

Multivariate time series data used by LSTM models for predicting faulty links is discussed in [10]. They use a single link and six features to train an LSTM model, such as laser bias current, input optical power, output optical power, temperature in the model, and detection point temperature. They claim that they increased fault detection efficiency by 98.41%. This paper expands their work by extending the topology and feature set. Even though we are dealing with a harder problem, we use a more sophisticated VAE model for temporal feature extraction and achieve similar results.

Studies like [11] and [12] show the importance of analyzing temporal and spatial relations in data with both time and space dimensions. From a survey of the available literature in [13], most ML models use vanilla GCNs for spatial feature extraction and either LSTMs, GRUs, or Transformer models for temporal feature extraction. This paper uses a unique implementation of a combination of GCNs with VAE after finding that we can train more purposed VAE models for temporal extraction by combining it with XGBoost.

We have identified that most existing work is limited by either using simulated data that does not reflect real network topologies or addressing a simplified problem using real data. We are expanding on these limitations by using data collected for a real topology and identifying link failures that any issue in the network could cause. We are also proposing a modified VAE model for better extraction of temporal features.

## IV. Methodology

### A. *Data Collection*

The data collected from the three sources mentioned earlier (WAE, XTC, and OneControl) contain features from IP and optical devices. These features represent the various Performance Metrics (PMs) reported by the constituent interfaces of the links. In this study, we focus on the time series features. The following table gives an overview of some of the important features available in the dataset for temporal study. The FLR metric (**Eq.** 1) is a custom metric introduced by the client.

$$\text{FLR} = \frac{\text{Error Packets [pps]}}{(\text{Error Packets [pps]} + \text{Packets In [pps]})} * 10^4 \quad (1)$$

***Table 1:*** Dataset Features

| Feature Name | Description |
|---|---|
| Traffic | Traffic In/Out Mbps (Min, Max, Avg) |
| Packets | Packets In/Out Per Second (Min, Max) |
| Error Packets | Error Packets In/Out Count |
| FLR | Packets Transmission Failure Rate |
| Uni-Directional Min Delay (UMD) | The forward path latency between two neighboring interfaces on a link |

### B. *Data Cleaning and Data Preprocessing*

After data collection, we fill in missing values for the time series features we are interested in. The final dataset is partitioned into five day sliding window frames with a stride value of 1 day. The label for the current window will be one if there are any IGP configuration changes in any of the days in the following window and zero if not. **Table 2** below gives important statistics about the data.

***Table 2:*** Important Statistics

| Statistic | Value |
|---|---|
| Number of Links | 215 |
| Number of IP Interfaces | 430 |
| Number of optical Interfaces | 252 |
| Number of optical Links | 63 |
| Total Number of Windows | 47085 |
| Windows with Link Failures | 963 |
| Class Imbalance | 1.5% |
| Training Months | 5 |
| Test Months | 3 |

### C. *Model Training and Inference*

We train various supervised classification ML models on the final preprocessed dataset and use precision-recall curves to measure the model's actual performance in our model evaluation. F1 score and Average Precision (AP) are used to determine the performance of the models trained in this paper. F1 generalizes recall and precision via harmonic mean computation. AP summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight. Finally, we use paired two-tailed t-tests to calculate statistical significance when comparing the distribution of performance scores models.

We have modified the standard VAE architecture by adding an ML classifier between the encoder and the decoder. The architecture of this temporal-based system may be considered to be a VAE solution with built-in classification functionality. In this approach, we use a loss function where, for a given sample $S_i(\theta, \emptyset)$, the loss is:

$$\mathbb{E}_{Z \sim q_\theta(z|x_i)}[\log p_\emptyset(x_i \mid z)] \quad (2)$$
$$+$$
$$\mathbb{KL}\Big(q_\emptyset(z \mid x_i)\Big) \,||\, p(z) \quad (3)$$
$$+$$
$$-y_i \log(p_i) - (1 - y_i)\log(1 - p_i) \quad (4)$$

The first term (2) is the expectation taken with respect to the encoder's distribution, $q_\emptyset(z \mid x_i)$, as parametrized by θ. $\log p_\emptyset(x_i \mid z)$ is the reconstruction loss. This is the log-likelihood of the reconstructed features against the original features. The second term (3) gives the Kullback-Leibler divergence [8] between the encoder's distribution and $p(z)$ . $p(z)$ is specified as a standard normal distribution with zero mean and variance of one. This is called the regularization term and is put in place to ensure that the representation (z) of each group of features is sufficiently diverse and that similar feature representations have similar latent space representations. The above terms are standard parameters to capture VAE loss. We add an additional classifier loss term (4), which gives the binary cross entropy loss (log loss). The total loss is then summed up for each sample in the batch, and backpropagation is used to update the weights. This helped the encoder's training process as it will try to encode the input samples into the latent space optimally so that they can be decoded and used for classification in parallel.

## V. Experiment Design and Results

### A. *Temporal Relation Detection with Autocorrelation*

We used statistical correlation functions to determine the relation of the time series features in time with itself. The autocorrelation function can be used to calculate the correlation of a time series observation with data points from the same observation in a previous time step. Studies like [14] have shown the use of Autocorrelation and Partial Autocorrelation functions to improve the performance of neural network models on time series forecasting tasks. In this study, we use them to show the correlation of some important features in time. The results of the autocorrelation computation are shown in **Figure 1**. Most of the time series features in the dataset exhibit plots like the ones shown in the figure. The blue area depicts the 95% confidence interval, indicating the significance threshold. From this, we can determine that the correlations in day lags become less significant after lag 3 to lag 5. We deduce that this dataset has a significant relation in time, which is significant up to 3 to 5 days. This is one of the reasons we are using 5-day windows.

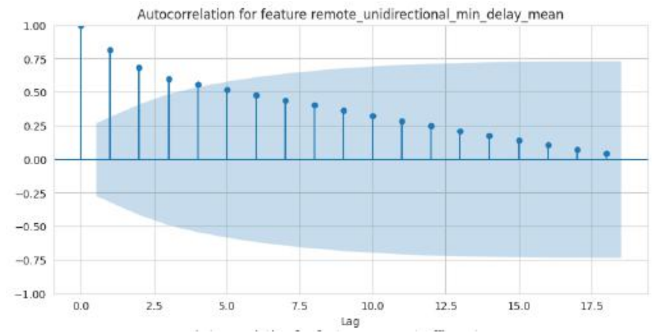


**Figure 1:** Autocorrelation Plots

We have also used Pearson cross-correlation plots to see interesting correlations across the time series features. Some are obvious, like packet in and out counters for source and remote interface, while others are not, like the correlation between uni-directional min delay and link bandwidth. A grid search on the window size confirmed the above statistical inference. A training set was prepared for each window size, and an XGBoost model was trained. Training the model with a window size of more than five days did not yield further improvements in performance.

### B. *Detection and Use of Temporal Relation with XGBoost*

This this sub-section and the next, we cover the steps taken for detection and use of spatiotemporal relation by various approaches. We the plot subsequent results achieved. Finally, we will provide a generalized overview of our recommendations and lessons learned. The F1 and AP scores of these approaches, along with the Precision-Recall Curve, are shown collectively in **Figure 2**.

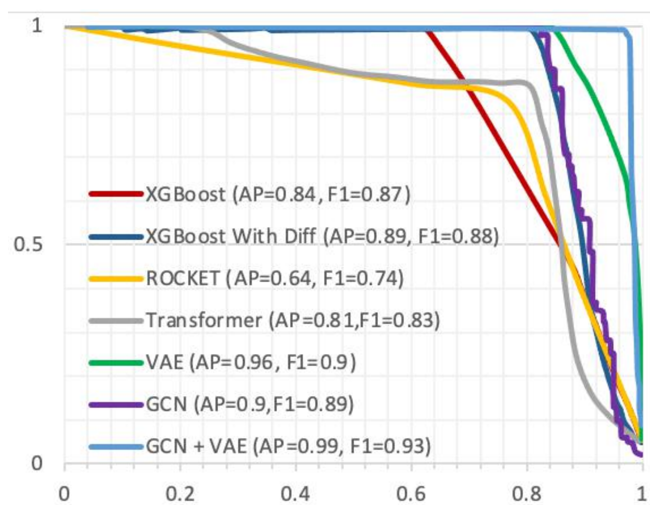


**Figure 2**: Model Performance Chart

To detect the relevance of temporal relations, we shuffled the data features to disarrange the sequence randomly for each link. This is very different from shuffling the dataset consistently across the entire dataset. Shuffling the dataset consistently will keep the performance the same since the order of the columns in the dataset is not used to make decisions. But if we randomly shuffle the columns for each row, the built-in temporal relation in the dataset is lost. This must be done only for the training dataset. An experiment conducted on the randomized dataset showed statistically significant degradation in performance hence showing the value of temporal relation.

In addition, by adding difference features between consecutive days as additional features to the dataset, we were able to significantly improve the XGBoost performance. In the remainder of this paper, we will refer to the difference features as "diff features". Adding these diff features increases the dataset with 745 features. We have used feature importance plots to confirm that the diff features are amongst the top of the feature importance list, meaning that the trend information encoded in them is one of the most important features.

### *C. Leveraging Temporal Relation with Time Series Classification Models*

For a neural network approach of time series classification, we tried ROCKETS, Transformers, and the modified VAE approach. From the results shown in **Figure 2**, we see that the ROCKETS model does not train as well as the XGBoost model with the diff features. This could be due to the intrinsic nature of the model to work with univariate datasets. It has been modified to work with multivariate time series datasets via column concatenation method. This dataset is a sparse multivariate, and hence it may not be well suited for models designed for univariate datasets. The same is true for the Transformer model. It could be too complex of a model for our heavily imbalanced dataset. Although we think that further configuration of Transformers may lead to better results.

The result also shows that the modified VAE model could encode temporal relations more effectively. We can see that this model has an even higher average precision, which is 7 percentage points more than the XGBoost model with diff features. This shows that the VAE model can capture more of the temporal relations than what is achieved by adding diff features. Now that we have a model performing well, we can use SHAP (SHapley Additive exPlanations) as shown in [15] to plot the feature importance plots for each time step to expand the interaction of feature values across the days of the window. The top SHAP values for the above model are given in **Figure 3**. We can see that the VAE model extracts valuable information from more sparse features like FLR rates not previously seen as important by the XGBoost model. This is confirmed by looking at the raw metric windows from the dataset for specific links.

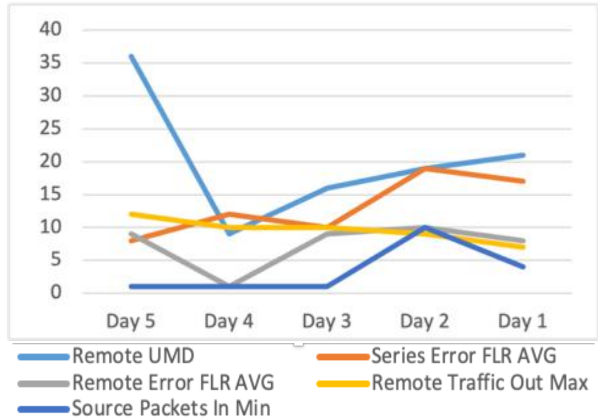


**Figure 3:** Modified VAE Model SHAP Importance Plot

### *D. Leveraging Spatial Features using Graphical Convolutional Networks*

Before using GCNs, we confirmed the relevance of adding neighboring node information by expanding the window data frames with information from the neighbors of a given node. The best neighbor was picked from all available neighbors to a link based on the amount of data on the neighbor. If the neighbor has fewer missing features, it presumably has more relevant information. Based on this heuristic, the trained XGBoost model with the additional features shows improved performance. **Figure 4** shows that the neighboring metrics (the ones in blue) are amongst the more important features for prediction, contributing up to 20% to the performance. It tells us that a neighbor's IGP metric configuration and traffic metric details carry significant information for predicting link failures.

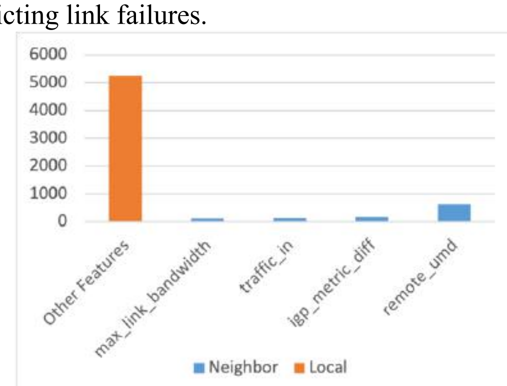


**Figure 4:** Feature Importance for Data from Neighbor

Next, using GCNs, were able to train a model that leveraged the graph topology structure built with node data, neighboring indices, and edge weights. The edge weights were kept static as we did not need to differentiate between neighbors based on importance as we did not have domain knowledge for prioritizing links. The GCN model had a better performance than the XGBoost model with the expanded feature set. This is shown by the model performance of the GCN model in **Figure 2**. Given that the number of hops in the construction of layers for the GCN model is a hyperparameter, fine turning it on this dataset showed that the optimal number of hops is two. Adding more hops has negative return as it could add more noise from links that do not carry relevant information.

### *E. Combining Temporal and Spatial Models*

We discussed that there are alternatives to combine temporal extraction models with graphical models for spatial extraction as shown in [13]. In this paper, we designed an architecture for the combination of the two components by adding the VAE temporal component right after the aggregation of data from neighboring links in the GCN component. Using this architecture, we were able to further improve the resultant model as demonstrated in **Figure 2**. This signifies that when extracting both spatial and temporal components via complex ML neural networks, we are able to outperform all standard models trained so far.

## VI. Conclusion

From this study, two conclusions can be drawn. First, the dataset's significance in temporal relations is discovered through autocorrelation and randomization techniques. Second, using the existing spatial and temporal relations, we can train better models by adding features that encode differences between features across days for temporal features and neighboring metrics for spatial features. However, even better models can be trained that, by design, can extract meaning from sequences like VAEs for temporal features and GCNs for spatial features.

Using feature importance plots, we see that the sequence models could leverage more abstract temporal relations in the dataset that lay in the error counters like FLR rate, while the GCN model leveraged IGP configuration and traffic counter metrics from neighboring links. Finally, a combination of the temporal and spatial neural networks gave the best performance when predicting IGP changes. These spatiotemporal correlations are currently not used by NOC operators for root cause identification of flapping links.

For future work, we suggest performing a similar study on dynamic networks. This is because, in reality, network operators are constantly adding and removing links. This issue must be accounted by ML algorithms for them to succeed more in the field. Another expansion is a closer study of the different ways the spatial and temporal components can be combined to extract more optimized features for ML models. Finally, more studies could be conducted to see if similar ML approaches could be applied to other relevant network issues like network congestion detection, base stations crash detection, and router reset issues.

## VII. References


[1] Bruck Wubete, Babak Esfandiari, Thomas Kunz, Thomas Triplet, David Cote, "Importance of optical metrics for IGP configuration change prediction", The International Conference on Intelligent Computing, Communication, Networking and Services, June, 2023.

[2] X. Chen, R. Proietti, C. -Y. Liu and S. J. Ben Yoo, "Towards self-driving optical networking with reinforcement learning and knowledge transferring," 2020 International Conference on Optical Network Design and Modeling (ONDM), Barcelona, Spain, 2020, pp. 1-3, doi: 10.23919/ONDM48393.2020.9133022.

[3] B. Chen, P. Sun, P. Zhang, J. Lan, Y. Bu and J. Shen, "Traffic engineering based on deep reinforcement learning in hybrid IP/SR network," in China Communications, vol. 18, no. 10, pp. 204-213, Oct. 2021, doi: 10.23919/JCC.2021.10.015.

[4] J. Guo, H. Guo and Z. Zhang, "Research on Intelligent Network Operation Management System Based on Anomaly Detection and Time Series Forecasting Algorithms," 2022 IEEE Conference on Telecommunications, Optics and Computer Science (TOCS), Dalian, China, 2022, pp. 338-341, doi: 10.1109/TOCS56154.2022.10016167.

[5] Tianqi Chen and Carlos Guestrin, " XGBoost: scalable tree boosting system," 2016, International Conference on Knowledge Discovery and Data Mining.

[6] Angus Dempster, François Petitjean, Geoffrey I. Webb, "ROCKET: Exceptionally fast and accurate time series classification using random convolutional kernels," Arxiv, Oct. 29, 2019.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin "Attention is all you need," Arxiv, Jun. 12, 2017.

[8] Diederik P Kingma, Max Welling, "Auto-Encoding variational bayes," Arxiv, Dec. 20, 2013.

[9] Thomas N. Kipf, and Max Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in International Conference on Learning Representations, Toulon, 2017

[10] Das Bhagwan, Wei Ji, Qiang Liu, Pengcheng Liu, "AI-Assisted Failure Location Platform for Optical Network," International Journal of Optics, 2023.

[11] Yichen Zhu, Mengtian Zhang, Bo Jiang, Haiming Jin, Jianqiang Huang, Xinbing Wang "Networked time series prediction with incomplete data," Arxiv, Oct. 5, 2021.

[12] Ken Chen, Fei Chen, Baisheng Lai, Zhongming Jin, Yong Liu, Kai Li et. al, "Dynamic spatiotemporal graph-based CNNs for traffic flow prediction", Arxiv, Dec. 5, 2018.

[13] Guangyin Jin, Yuxuan Liang, Yuchen Fang, Jincai Huang, Junbo Zhang, and Yu Zheng, "Spatio-Temporal Graph Neural Networks for Predictive Learning in Urban Computing: A Survey" Arxiv, Mar. 25, 2023

[14] J. H. F. Flores, P. M. Engel and R. C. Pinto, "Autocorrelation and partial autocorrelation functions to improve neural networks models on univariate time series forecasting," The 2012 International Joint Conference on Neural Networks (IJCNN), Brisbane, QLD, Australia, 2012, pp. 1-8, doi: 10.1109/IJCNN.2012.6252470.

[15] Scott Lundberg, Su-In Lee, "A unified approach to interpreting model predictions," in Conference on Neural Information Processing Systems, Long Beach, 2017.