LKN: Graph Neural Networking Challenge 2023 3rd Place Solution Report

Kaan Aykurt*, Maximilian Stephan*, Serkut Ayvasik*, Johannes Zerwas*
*Chair of Communication Networks, Technical University of Munich, Germany

***Abstract*—Unlike the existing methods such as Queuing Theory (QT) or network simulation studies, emerging state-of-the-art Graph Neural Networks (GNNs) for network performance analysis open possibilities for creating digital twins of networks. In this report, we present our solution for the Graph Neural Network Challenge 2023.**

I. Introduction

Over the past decades, the integration of technology into society is accelerating. The increasing popularity of augmented reality, virtual reality, and telepresence applications shape the requirements for the next generation of networking. As the complexity of networks is growing, managing and modeling networks is also becoming a challenging task.

Although various tools are actively being researched for network performance analysis, an end-to-end framework for autonomous management focusing on real networks is not yet existent. To close this gap, as the initial step, we participated in the Graph Neural Networking Challenge 2023 to investigate potential mechanisms of creating a real network digital twin.

The remainder of this paper is structured as follows: Sec. II provides an overview of the Graph Neural Networks (GNNs). Sec. III discusses the solution methodology. Finally, we conclude and discuss the next steps in Sec. IV.

II. Background

A. Graph Neural Networks

In recent years, GNNs [1] are widely used for modeling data which can be expressed as graphs. While traditional neural networks assume a connection structure with a fixed-dimension input space (e.g. fully-connected neural networks, or convolutional neural networks), GNN architecture is determined dynamically based on the input graphs. This means that they have the potential to exploit graph-structured characteristics of the input data to model the relationships between the graph elements.

Predictions: A network manager's all-time dream is to sit back and watch the network, while the maintenance tasks are autonomously managed. The goal of network management is to include the human in the loop only for overseeing the actions the network management framework takes. GNN-based ML models create opportunities to predict network characteristics with high accuracy, and therefore serve as a powerful tool for network analysis and autonomous network management.

Generalization Capabilities: One of the critical requirements of a successful network management framework is the ability to adapt and perform well under unknown circumstances. GNNs display promising abilities to generalize and perform well for unseen data, topology, and traffic characteristics [2]. Hence, this makes GNNs a good basis to realize DTs.

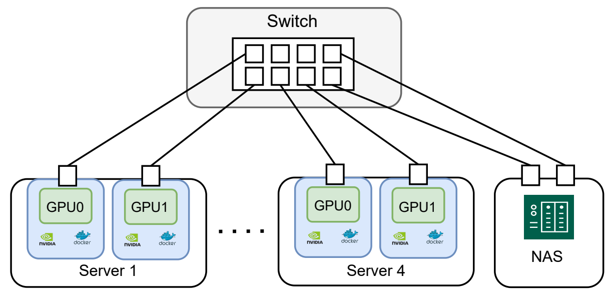

Fig. 1. Overview of the LKN compute testbed.

In addition to network characteristics, domain-specific information can also be modeled as nodes in graphs. Considering modern network topologies, in which multi-domain characteristics also need to be taken into account, the incorporation of domain-specific knowledge plays a key role in a successful network management framework.

III. Methodology

A. Approach

Our approach to the challenge consists of a continuous loop of preprocessing, training, and analysis. We initially focus on feature extraction via preprocessing and conduct a study to train with various hyperparameter selections. Upon the analysis of the trained model, we take a step back and try different features with different hyperparameters.

B. Testbed

Fig. 3 shows the testbed for training and evaluation of GNN models. Our testbed consists of five servers. Four of the servers are equipped with two GPUs each. Although the GPU models are different in different servers, each GPU has its own interface on the NIC. Additionally, we have a Network Attached Storage (NAS) server where we host the challenge data. All of the servers are connected via an Intel Tofino P4 switch with 100G links providing all-to-all connectivity.

We utilize Docker containers to ease the reproducibility of our measurement setup. In the end, we don't employ training parallelism, but train a single model configuration on a single

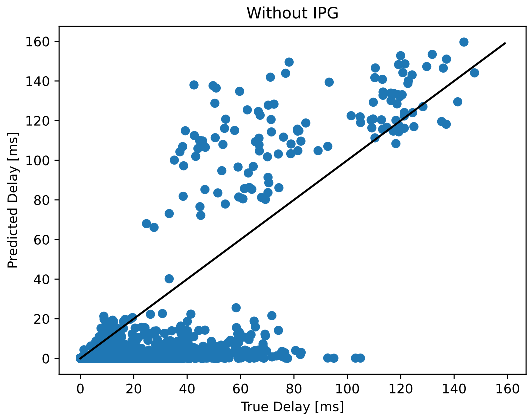

Fig. 2. Without IPG.

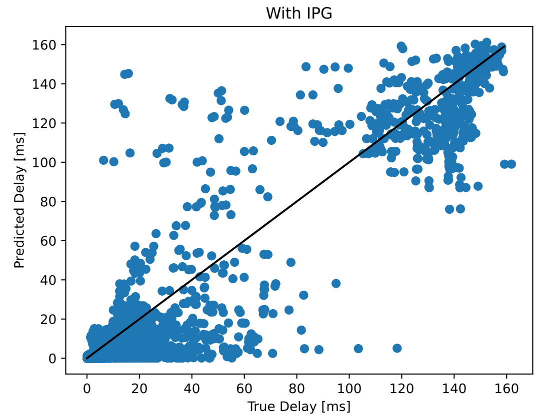

Fig. 3. With IPG.

model. The final evaluated model is trained in a Tesla T4 GPU and training time for one epoch took 12 minutes.

C. Model

We rely on the baseline implementation of RouteNet-Fermi for our model. Our experience consists of changing model hyperparameters as well as model layers. We propose to update the link and path hidden states with RNNs and LSTMs. However, our analysis of the results indicates that GRU update for link hidden states and RNN update for path hidden states, which is already proposed by the baseline provides the best results, hence we go ahead with the existing model.

For the model training, we employ early stopping criteria where we stop the training if the change in the validation loss is smaller than a selected threshold of 0.0001. Additionally, we adaptively reduce the learning rate by 99% of the previous epoch in each iteration.

D. Features

Our final solution relies on the provided features by the baseline preprocessing algorithm. Additionally, we extract the feature of Inter-Packet-Gap (IPG). We calculate IPG by the time difference of packets in a flow. From the list of differences, we extract the mean, variance and every 10th percentile of IPG. With the additional moments in addition to mean and variance, this gives an idea of the burstyness and distribution of the arrival of packets. With our extended feature, we present a significant improvement to the baseline model.

E. Results

Fig. **??** presents the scatter plot of true and predicted delays without the extended IPG feature. In this scenario, the reported MAPE is around 45. We present our final findings in Fig. **??**. With the inclusion of IPG, we improve the baseline MAPE by 25% and achieve a test MAPE of 35.39.

IV. Conclusion

Real networks exhibit complex characteristics. Upon the analysis of the raw data and results, hardware artifacts in addition to the lack of knowledge of queue states make the challenge harder to predict flow level delay. However, our findings indicate that there can be room for improvement upon extending the features that may contribute to congestion, and hence the delay. We would like to thank the organizers for the challenge and giving us access to such a dataset. We envision that with further improvements, creating a real network digital twin may be feasible.

References

[1] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[2] J. Suárez-Varela, S. Carol-Bosch, K. Rusek, P. Almasan, M. Arias, P. Barlet-Ros, and A. Cabellos-Aparicio, "Challenging the generalization capabilities of graph neural networks for network modeling," in *Proc. ACM SIGCOMM Conference Posters and Demos*, 2019, pp. 114–115.