DELAY ESTIMATION BASED ON MULTIPLE STAGE MESSAGE PASSING WITH ATTENTION MECHANISM USING A REAL NETWORK COMMUNICATION DATASET

Cláudio Modesto[1], Rebecca Aben-Athar[1], Andrey Silva[2], Silvia Lins[2], Glauco Gonçalves[1], Aldebaro Klautau[1]
[1]LASSE, Federal University of Pará, Brazil, [2]Ericsson Research, Brazil

NOTE: Corresponding author: Cláudio Modesto, claudio.barata@itec.ufpa.br

Abstract – *Modeling network communication environments with Graph Neural Networks (GNNs) has gained notoriety in recent years due to the capability of GNNs to generalize well for data defined over graphs. Hence, GNN models have been used to abstract complex relationships from network environments, creating the so-called digital twins, with the objective of predicting important quality of service metrics, such as delay, jitter, link utilization, and so on. However, most previous work has used synthetic data obtained with simulations. The research question posed by the "ITU Graph Neural Networking Challenge 2023" is whether GNN models are capable of estimating the mean per-flow delay network, using data from a real network environment. The solution presented in this paper achieved first place in the mentioned challenge. It adopted a GNN based on multiple-stage message passing and the attention mechanism to predict the mean per-flow delay. Furthermore, feature selection was used to choose a reasonable subset of input parameters. The developed GNN model achieved a mean absolute percentage error under 20.1% in the challenge test dataset, which was composed by network conditions not used in the training dataset.*

Keywords – 5G, delay estimation, digital twin, feature selection, real dataset.

1. INTRODUCTION

Given the rapid evolution of communication technologies and the emergence of 5G and 6G networks, the demand for continuous connectivity and efficient real time communication has never been more significant. As the complexity of networks continues to grow, optimizing them becomes critical. In this sense, it highlights the importance of predicting users' Quality-of-Service (QoS), such as network delay, to reduce latency, ensure timely delivery of services, and provide more efficient communication and a consistent experience for users in the digital scenario. Whether enabling ultra-low latency for critical applications or supporting massive connectivity for IoT devices, network optimization plays a central role in meeting the diverse needs of today's digital landscape.

Following this reasoning, Network Digital Twins (NDTs) emerge as a promising approach to managing dynamic network behavior through a virtual representation of the physical network. This tool is valuable for analyzing, diagnosing, emulating, and controlling the physical network in real time through data, models, and interfaces, facilitating an interactive mapping between the physical and the digital twin model [1]. A DTN can also be used to handle large amounts of network data, facilitate decision-making and monitoring of the network in real time, conduct troubleshooting, *what-if* analysis, and the strategic planning of network updates [2, 3, 4].

Graph Neural Network (GNN) models are suitable for network modeling as a virtual representation of the physical network due to their ability to process topological information and generalize over data graphs, overcoming the limitations of conventional approaches [5, 6]. This type of neural network is designed to work with graph-structured data, and it can hold the structure information embedded in a graph via a message-passing algorithm among nodes and aggregate the node features at various levels of the graph [7, 8]. With nodes, in the telecommunications field, representing entities such as devices, routers, or communication endpoints; and edges (links) depicting the connections or interaction between them, carrying crucial information regarding the bandwidth and latency characteristics, forming the backbone of the network's topology.

The RouteNet-Fermi architecture, as proposed in [9] for network modeling, extends its scope beyond nodes and links, considering the activated flow of the network as an input graph. This flow follows specific source-destination paths generated from a given traffic model. The model incorporates a modification in the message phase of the Message-Passing Neural Network (MPNN), allowing iterations over multiple stages and offering an effective approach to address complex structures within the network. In this work, we took as a baseline a modified version of the RouteNet-Fermi architecture provided by the challenge organizers.

Therefore, this paper presents the solution developed within the scope of the ITU Graph Neural Networking (GNNet) Challenge 2023, created by the Barcelona Neural Networks Center (BNNC), which provided the competitors with a dataset with characteristics extracted from a real network communication. The presented solution focuses on predicting the QoS metric mean per-flow delay and incorporates feature selection processing combined with the modified RouteNet-Fermi model to reduce the complexity of the model and prevent it from learning irrelevant patterns. In addition, we apply z-score feature normalization, which, unlike *min-max* normalization used in the baseline, can deal with outliers more robustly. Finally, an attention mechanism is also implemented inside the Multiple-Stage Message Passing (MSMP) architecture to improve the model's ability to focus on relevant information during the learning process.

The remaining sections of this paper are organized as follows: Section 2 compares work that used GNN for network modeling to our model developed for the GNNet Challenge 2023. In Section 3, we present the network topology used in this work. Section 4 discusses the baseline, a modified version of the RouteNet-Fermi model. In Section 5, we present our solution, exposing the steps we followed so that our model reached a Mean Absolute Percentage Error (MAPE) of 20.001% in the prediction of mean per-flow delay, followed by Section 6, which presents the results of the experiments, discussing the performance improvements after each change in the model, and Section 7 concludes the paper and discusses future work.

2. RELATED WORK

Network modeling plays a crucial role in the field of computer networks, being essential for the planning and optimization of these infrastructures. Although traditional deep learning models have shown their usefulness in complex scenarios such as non-linear traffic behavior and high-dimensional problems, where their effectiveness is limited by the lack of generalization when faced with different topologies [5]. GNNs emerge as a suitable alternative in this scenario due to their unique ability to generalize to networks not seen during training. They dynamically adapt to operational changes, overcoming conventional limitations, and can be used to understand complex relationships between topology, routing, and traffic [3].

Considering this, numerous studies have employed GNNs to model network communication. Suzuki et al. [10], use a Graph Convolutional Networks (GCNs) to propose a simple semi-supervised learning classifier. The model employs three stacked GCN layers, enabling the estimation of communication delays between pairs of nodes. Jin et al. [4], a GNN model was proposed to estimate the latency between an *origin-destination* pair in a routing network for a prediction task in a Software Define Network (SDN), also using GCN principles. Almasan et al. [11] integrated GNN into a deep reinforcement learning agent to solve routing optimization in optical networks. Similar to our work, they used MPNN to capture meaningful information about the relation between the links and the traffic flowing through the network topologies. Happ et al. [6] is close to our approach in the sense their authors, in the scope of GNNet Challenge 2021, employ an architecture based on the MSMP focusing on predicting the mean per-flow delay in a network communication. But this work differs from ours by using multiple scheduling policies to deal with packets of different flows, using a topology network of synthetic nature with a dataset generated by the OMNeT++ [12] simulator.

Our work distinguishes itself from the others by exploring the MSMP architecture with two stages in the message passing process, which provide an effective way of dealing with complex structures in the network, implementing the attention mechanism algorithm inside the MSMP update function. The attention mechanism, unlike GCN, gives each node in the graph the ability to weigh its neighbors' relevance, not only considering the number of neighbors but also their local structure. Our work stands out by using the network topology based on a real-world dataset[1] obtained from the BNNC testbed that comprises eight hardware routers and two switches interconnecting with the open source projects TRex [13] and DPDK [14]; for traffic generator and capture, respectively.

3. PROBLEM CONTEXT

The definitions of the problem considered in this work were derived from the ITU GNNet Challenge 2023, where, the main objective was to create a network digital twin model capable of predicting accurately QoS performance metrics. Moreover, the starting point for this was a baseline constructed by the organizers with the RouteNet-Fermi GNN model that uses as input different network features from links and flows of a communication topology, intended to generalize this data defined over graphs and estimate the mean per-flow delay metric. Thereby, in a simplified perspective, this problem could be summarized by the pipeline in Fig. 1, where there is the GNN model that should take the network input features to perform the necessary training processing, considering a supervised learning approach, aimed to generate the predicted delay.

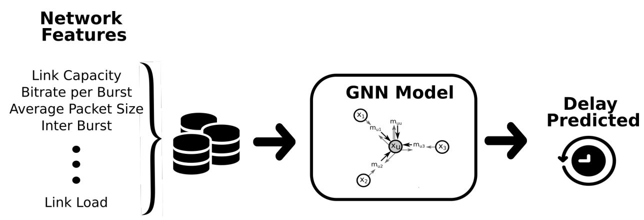


Figure 1 – Systematic perspective of the delay estimation problem.

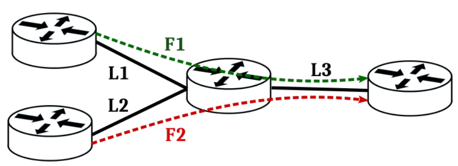
In addition, these network features shown in Fig. 1, are part of a raw real network dataset with two traffic types, called Constant-Bit Rate and Multi-Burst, and with network topology organized as a 4-tuple graph $\Gamma = (\mathcal{V}, \mathcal{E}, \mathbf{r}, \mathbf{s})$ which depicts a set of nodes $\mathcal{V}$, with a range of 5 up to 8 nodes, with each node comprising a network router, organized pair-to-pair to compose a set of edges

[1] `https://bnn.upc.edu/challenge/gnnet2023/dataset/`

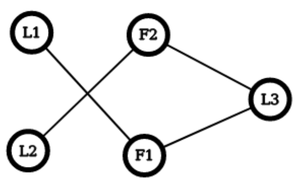
$\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, representing the network link between them. Moreover, each node $\{u, v\} \in \mathcal{V}$ and edge $e \in \mathcal{E}$ have intrinsically a feature vector $\mathbf{r}$ and $\mathbf{s}$ that defines, respectively, the node and edges entity's characteristics, namely node and edges features.

The referred input features for the GNN model utilized are obtained by transforming the homogeneous graph Γ into undirected heterogeneous [15] 3-tuple graph $\mathcal{G} = (\mathcal{V}', \mathcal{E}', \mathbf{x})$ with an object function map $\tau : \mathcal{V}' \rightarrow \mathcal{A}$, where $\mathcal{A}$ is set of the node type, such that each node v belongs to one particular type $\tau(v) \in \mathcal{A}$. In the scope of this work, there are two: flow and link node types. Furthermore, since there are two node types, there exists a node feature $\mathbf{x} = \{\mathbf{x}_f \vee \mathbf{x}_l\}$, where, $\mathbf{x}_f$ and $\mathbf{x}_l$ are respectively flow and link level features. In this sense, with two different node features, these could assume different dimensions, meaning in the end, that two different embedding types should be used for each node feature. Therefore, in a dimensionality analysis, the node features $\mathbf{x}$ of a graph in this work is given by a flow-level $\mathbf{x}_f \in \mathbb{R}^m$ and link-level $\mathbf{x}_l \in \mathbb{R}^n$.

The process to obtain the graph $\mathcal{G}$ is conducted with a transformation whose initial state is the network topology Γ already known, that is, the routers and the links that define the communication between those; and the final state is the graph considering the flows and links in each communication between the routers. For example, the final and the last state is shown graphically in Fig. 2, being in (a) the initial state defined by network topology composed of four routers, with three links ($L1$, $L2$, and $L3$) and two flows ($F1$, $F2$). The final state in (b) considers the two aforementioned types of nodes, and for each dependency between a flow and a link, an edge is defined. For instance, the flow $F2$ depends on two links, $L2$, and $L3$; for this reason there exist two edges that link node $F2$ with the nodes $L2$ and $L3$.



(a) High-level relation between flows and links considering four routers.



(b) Heterogeneous graph generated from flows and links in (a).

Figure 2 – Graph transformation considering the flows and links from a network communication topology.

With this heterogeneous graph, given a node u, its neighborhood $\mathcal{N}_u$ is defined by the edge that links a node v to a node u or vice versa, as described in

$$\mathcal{N}_u = \{v | (u, v) \in \mathcal{E} \vee (v, u) \in \mathcal{E}\}. \quad (1)$$

In qualitative terms, there are plenty of features, more than considered in the baseline, in the dataset offered by the GNNet challenge organizers, to be used as input for both flow and link features. Some of these features are displayed in Table 1. In this table, the features are split into three classes namely, network topology, routing configuration, and traffic configuration. Moreover, each feature has a specific role depending on its type, which can be flow, link, global, label, or topological. Finally, each feature has a unit for a better comprehension of its purpose.

Table 1 – Different types of features in the dataset.

Class	Features	Feature Role	Unit
Network Topology	Nodes	Topological	-
	Edges	Topological	-
	Link capacity	Link	Gbits/s
Routing Configuration	Source	Topological	-
	Destination	Topological	-
	Flow length	Flow	flow
Traffic Configuration	Average bandwidth	Flow	Mbits/s
	Constant bitrate	Flow	Mbits/s
	Link load	Link	ratio
	Maximum Link load	Global	ratio
	Global delay	Global	s
	Normalized link load	Link	ratio
	Number of packets per burst	Flow	packets
	Average packet size	Flow	bytes
	Inter burst gap	Flow	μs
	Bitrate per burst	Flow	Mbits/s
	Traffic type	Flow	string
	Packet size 90 percentile	Flow	bits
	Packet size 80 percentile	Flow	bits
	Packet size 50 percentile	Flow	bits
	Packet size 20 percentile	Flow	bits
	Packet size 10 percentile	Flow	bits
	Type of Service	Flow	bits
	Inter packet gap mean	Flow	ns
	Packets generated	Flow	packets/s
	Inter packet gap variance	Flow	ns
	Packet size variance	Flow	bits
	Delay	Label	s

It is noteworthy that topological and global roles are different kinds of features; the former do not work as input features for the GNN model, but are used to organize information about the source and destination of each packet in the network communication, as well as, the path of these packets; hence this type of feature could be defined as metadata for each flow and link level features. The latter features could be used as input since these types of features measure the general performance of the simulation, i.e. the *maximum link load* feature considers the load of all flows in the simulation and takes the maximum of it to generate this metric. However, is necessary to mention that an underlying preprocessing must be made for it, as the dimension of global features is different from flow and link level features. In other words, global features are just a real number. An example of this preprocessing is the feature that we have created called *normalized link load*. This feature was obtained by dividing the load applied in each link, referred to as *link load*, by the maximum link load when generating the traffic matrix of the scenario.

Finally, the label role is associated with the learning approach used in this work; in this case, the supervised learning to predict the mean per-flow delay.

4. BASELINE

The starting point of this work is associated with the baseline created in the scope of the GNNet Challenge 2023; in this sense, a modified version of the RouteNet-Fermi model. In the first place, the RouteNet-Fermi is an architecture constructed having as a base the MPNN proposed by [16]. In simple terms, the MPNN model was a framework that assumed different architectures over time during the development of GNN theory in the literature. Thereby, this state-of-the-art model is composed of three major blocks: the first one is the embedding initialization phase, the second one is the message phase and the last one is the readout phase. Each block could be defined in different ways and with various works that made it, as shown in [16]. However, the MPNN proposed in this same work does not specify the embedding initialization algorithm but defines that the message phase is constructed with a Gated Recurrent Unit (GRU) layer as proposed initially by [17] in a model called Gated Graph Neural Networks. The readout function could be defined by the same function used in [17] or a set2set model from [18].

Taking the above into consideration, the embedding initialization function defined in the baseline is made with a Multilayer Perceptron (MLP); this MLP considers as input a matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$, where n is the number of nodes and m the number of features per node of graph $\mathcal{G}$. Finally, generating another matrix $\mathbf{H} \in \mathbb{R}^{n \times k}$ called feature embedding; the rows of this matrix are also defined by the number of nodes n in the graph, but the number of columns is defined by a model hyperparameter, the embedding length k. In the case of this baseline, $m = 3$ and $m = 2$ were adopted for the input layer of flow h_f and link h_l embeddings, respectively; where $\{f, l\} \in \mathbb{N}^*$. And $k = 64$ for both embedding types. Pictorially, in Fig. 3 it is possible to identify this embedding creation process in a generalized form.

The message phase shares some similarities, but RouteNet-Fermi implements a modification that makes this phase iterate by a certain amount of time through multiple stages. In this sense, the amount of stages in the message phase is determined by the number of embedding types, and for each one, there is a GRU layer that must process this information and update the embedding with the neighborhood information. As mentioned previously, there are two embedding types, namely flow and link embedding.

The readout phase is also different from the classical MPNN, being used as the function of the MLP proposed by [19]. The readout output is the occupancy rate $\mathcal{O}$ in a given flow, which is used to predict the mean per-flow delay $\mathcal{D}$ by dividing it by the link capacity C that a flow

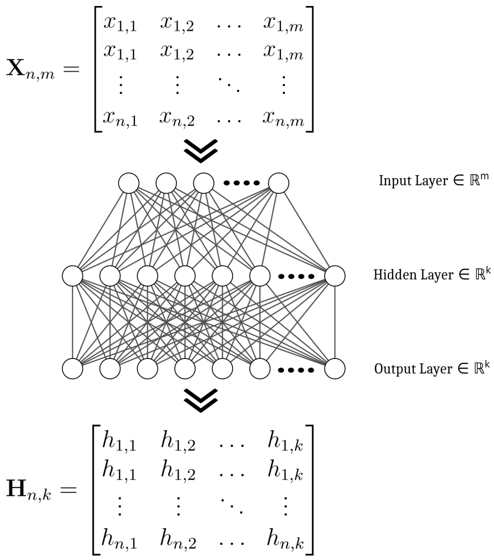


Figure 3 – Embedding initialization scheme.

passes through. Hence, considering that a flow could pass by N links, generating an occupancy rate for each one, the delay predicted is given by

$$\mathcal{D} = \sum_{j=0}^{N} \left(\frac{\mathcal{O}_j}{C_j} \right). \tag{2}$$

Finally, the modified RouteNet-Fermi used in our context can be represented by Fig. 4, which shows the embeddings h_f and h_l created by the MLP in the initialization phase, using the features present in Table 2, designed by their names and units; and these embeddings are being processed during T iterations in message phase, to generate the occupancy at the end of the model closing the loop.

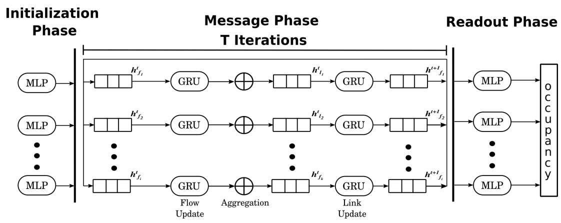


Figure 4 – Message-passing architecture with two stages.

Table 2 – Subset of features utilized as inputs in the baseline model.

Features	Unit
Link capacity	Gbits/s
Average bandwidth	Mbits/s
Link Load	ratio
Packets generated	packets/s
Constant bitrate	Mbits/s

5. PROPOSED SOLUTION

The design of our proposed solution aimed at enhancing the baseline solution defined by the challenge organizers. In this sense, the understanding of our developed solution [2] pass throughout to detail the feature selection process applied in each feature intends to define the best subset of it and use this as the input for the MSMP with two stages, define the feature normalization technique used during the learning process of the neural model, the modifications applied in the MLP presents in the initialization and readout phase of the MSMP, and finally, the attention mechanism layer explored inside of update function of the MSMP.

5.1 Feature selection

The role of Artificial Intelligence (AI) in communication has become more relevant since it is being used in congestion control [20], cybersecurity [21], 5th Generation (5G) mobile networks [22], among others. However, significant challenges arise when obtaining a large set of data from diverse sources to feed AI models for a given problem. The indiscriminate use of this extensive dataset as input to a model can increase complexity and excessive resource consumption, impacting scalability, practical feasibility, and increasing training time [23]. Furthermore, adopting data without prior filtering can expose the model to noisy and redundant information, leading it to learn irrelevant patterns from the training data that are not representative of the real problem (overfitting), thus compromising effectiveness, efficiency, and the capacity of model generalization. Given this, to train an optimal model, it is important to make sure that it uses only the essential features. Feature selection, one of the most crucial techniques in Machine Learning (ML) and data science, automatically chooses relevant features for an ML model based on the problem that must be solved. It is done by including or excluding essential features without changing them, and it helps cut down the noise and reduce the size of input data.

This feature characteristics selection has aroused considerable interest among researchers due to the possibility of its application in different fields. As a result, several feature selection approaches have been proposed over the years for supervised, unsupervised, and semi-supervised problems [24]. For this work, we chose two supervised techniques: filter-based and wrapper-based approaches.

5.1.1 Filter-based approach

This technique is called "filter-based" because it uses the selected metric to find irrelevant attributes and is filtered out from the model. This approach was chosen for its simplicity and to evaluate the features individually, since the filter methods assess the relevance of each feature and rank them according to their scores, reflecting their relationship with the target variable, the label in the dataset. This model is faster than the wrapper approach, making it particularly suitable for situations with many features. It results in a better generalization since it acts independently of the learning algorithm [25].

For this approach, the Mutual Information (MI) algorithm [26] was chosen. The MI between two random variables is a non-negative value, which measures the dependency of these variables. It is equal to zero if and only if two random variables are independent, and higher values mean higher dependency. For continuous random variables, the mutual information is given by [27]:

$$I(x;y) = \int_x \int_y p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) \mathrm{d}x\mathrm{d}y. \qquad (3)$$

In Equation *(3)*, $p(x)$ is the probability density of the feature $x, p(y)$ is the probability density of the mean per-flow delay, represented by y, and $p(x,y)$ is the joint density. Therefore, the criterion $I(x;y)$, in our context, measures the dependence between the features and the mean per-flow delay.

In numerical terms, to avoid dealing with integrals in Equation *(3)*, one possibility is to convert the continuous random variables to discrete values using binning. Instead of doing that, in this work a method of entropy estimation based on the distances of the k-nearest neighbors [28, 29] was used. This method locally estimates distributions through statistics of distances between data points, providing a more refined view of the characteristics of the continuous data set.

5.1.2 Wrapper-based approach

The wrapper method was used because filters may not identify features that bring complementary information from others. This approach includes assessing the importance of features using a specific machine learning model, in our case, the linear regression, and often gives better performance results than the filter method because it takes into account the feature dependencies and directly incorporates bias in the learning algorithm [24]. For this approach, an exhaustive feature selection search was used, which traverses the entire search space, evaluates every possible set of features, compares their performance, and chooses the best performing subset. This approach is computationally demanding, especially for large datasets, yet it ensures the optimal feature subset [30].

The subset of features was evaluated based on the Mean Square Error (MSE) metric, with the best performing subset being the one with the features that best contribute to the prediction of the mean per-flow delay.

5.1.3 Feature selection preprocessing

The network features utilized as input to the feature selection models, was organized in a matrix inserted in the $\mathbb{R}^{n \times m}$ space, with n denoting the amount of flow and m the number of features. This organization extends to the

[2] Our solution is available at `https://github.com/ITU-AI-ML-in-5G-Challenge/ITU-ML5G-PS-007-GNN-m0b1us`

mean per-flow delay, which is our label. Before the application of feature selection algorithms, aiming to reduce the data volume, certain flow features from Table 1 were discarded after the analysis and testing process. Furthermore, since the link-level features were fewer in number, these types of features were not considered in the process. Given that, some flow exhibited values on different scales, the z-score normalization, presented in Subsection 5.2, was also applied to the data.

5.1.4 The chosen features

From the flow-level traffic in the dataset, all features except for traffic type, since this is a non-numerical feature and packet size variance, underwent the feature selection process, as detailed in Table 3. This table considers that these features are grouped into two classes, one class associated with traffic configuration features, and another one regarding routing configuration features.

Table 3 – Subset of features to be analyzed by the feature selection algorithms.

Features	Feature Role	Unit
Flow length	Flow	flow
Average bandwidth	Flow	Mbits/s
Constant bitrate	Flow	Mbits/s
Number of packets per burst	Flow	packets
Average packet size	Flow	bytes
Inter burst gap	Flow	μs
Bitrate per burst	Flow	Mbits/s
Type of Service	Flow	always 0
Packet size 90 percentile	Flow	bits
Packet size 80 percentile	Flow	bits
Packet size 50 percentile	Flow	bits
Packet size 20 percentile	Flow	bits
Packet size 10 percentile	Flow	bits
Inter packet gap mean	Flow	ns
Packets generated	Flow	packets/s
Inter packet gap variance	Flow	ns

The subset of features that obtained the highest rankings through the filter method closely matched the optimal subset identified by the wrapper method, affirming their suitability for our model. Table 4 presents the final subset of flow-level features used as input to our model, which was selected using feature selection techniques, organized in descending order from the highest contribution to the lowest, as generated by the filter algorithm.

Graphically, the feature selection pipeline is depicted in Fig. 5, illustrating the matrix features and the sequential application of z-score normalization to the dataset. Following normalization, the features underwent a feature selection process utilizing both filter and wrapper-based methods. The algorithms employed yielded comparable results in identifying the most influential characteristics of our model, with the wrapper method encompassing features with the highest scores as determined by the MI

Table 4 – Subset of features obtained after the feature selection.

Features	Feature Role	Unit
Flow length	Flow	flow
Average bandwidth	Flow	bits/s
Inter packet gap variance	Flow	ns
Bitrate per burst	Flow	Mbits/s
Packet size 90 percentile	Flow	bits
Packets generated	Flow	packets/s
Constant bitrate	Flow	Mbits/s
Number of packets per burst	Flow	packets
Inter packet gap mean	Flow	ns
Inter burst gap	Flow	μs

algorithm.

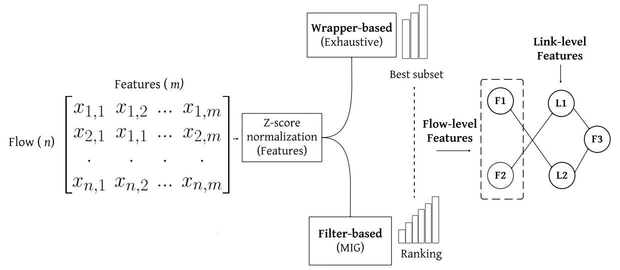


Figure 5 – Steps that describe the feature selection process adopted.

5.2 Feature normalization

The next step of our proposed solution is to suit each input feature in a common range, performing a different normalization approach from the baseline. In this sense, the z-score normalization was used in our solution instead of the default feature normalization found in the baseline model, that is *min-max* normalization. Usually, the z-score approach can lead to a model more robust to outliers' values than *min-max* [31] considering the distribution of each feature, since the normalization parameters of a node feature **x** is defined by the mean μ and the standard deviation σ of this array, instead their maximum and minimum. Therefore given a sample x of this array, the normalization value is defined by

$$x_{norm} = \frac{x - \mu}{\sigma}. \tag{4}$$

5.3 SeLU activation function

The use of Scaled Exponential Linear Unit (SeLU) represents an improvement in different layers for each MLP in our solution's initialization and readout phase. In this sense, in optimization and challenge context, not necessarily the use of Rectified Linear Unit (ReLU) represents the best choice since different activation functions could achieve better performance [32]. Moreover, once the activation function, in the case of MLP, to be used is chosen, it is necessary to take a step further and match this ac-tivation function with the right weight initialization algo-rithm. Thus, in the case of the baseline, this further step was not taken, and by default, Keras uses the *Glorot* with uniform distribution as an initialization algorithm [32]. Therefore, to suit the SeLU in each layer of MLP present in the initialization and readout phase of the MSMP, with the right algorithm, the *LeCun* algorithm with normal dis-tribution was used.

5.4 Attention mechanism

Before the training process starts, the approach of giving importance to a feature instead of another is made using feature selection techniques, as previously mentioned. In this sense, the basic premise relies on the fact that features that do not contribute to increasing the performance of the model are discarded. However, during the training process, it is also interesting to perform a kind of fine-tuning to identify again the importance of selected features, but in this case, not eliminating them definitively, but decreasing or increasing their relevance from scores. In the literature, there are several ways to compute these scores [33], with emphasis on the method proposed by Vaswani et al. [34] called *scaled dot-product attention*; but, in the graph-structured data, a widely used manner to obtain it is Graph Attention Networks (GAT), proposed by Veličković et al. [35] as the first GNN architecture to implement an attention mechanism applied over graphs.

The GAT model was subsequently enhanced by Brody et al. [36] that proposed using

$$\alpha_{uv} = \frac{\exp(\mathbf{a}^T \text{LeakyReLU}(\mathbf{W}[\mathbf{h}_u || \mathbf{h}_v]))}{\sum\limits_{k \in \mathcal{N}_u} \exp(\mathbf{a}^T \text{LeakyReLU}(\mathbf{W}[\mathbf{h}_u || \mathbf{h}_k]))} \tag{5}$$

for computing attention scores, which was adopted in this work. In Equation *(5)*, α_{uv} corresponds to the normalized attention coefficient between nodes u and v; **a** is a learnable weight vector; and **W** is a learnable weight matrix. The $\cdot^T$ and $||$ are the matrix transpose and concatenation operations, respectively.
The difference between the attention equation in [35] and Equation *(5)* is the order in which each operation is applied. In the latter equation the multiplication by **W** occurs after the concatenation between the node embeddings, $\mathbf{h}_u$ and $\mathbf{h}_v$, whereas in the former this concatenation is done after the multiplication of each embedding node by the weight matrix. Besides, in Equation *(5)* the vector **a** is not used as an argument in the LeakyReLU function.

This attentional model and its variants are also models that have a narrow relationship with the MPNN architecture as shown by the authors in [37]. This relationship is defined in terms of generality and interpretability [38], in such that they express this association in terms that MPNN represents a generalization of GAT models, where the main difference is how the arguments of update function are defined in each model. In mathematical terms, the GAT model is also defined as a subset of the MPNN model [39].

Considering either the GNN model formal definitions [37], this generality relationship becomes more evident when compared to the arguments of update function $\phi(\cdot)$ of both models, where it is possible to note additional terms in the GAT model if compared to the MPNN. In this sense, using

$$h_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} \psi(\mathbf{x}_u, \mathbf{x}_v)\right) \tag{6}$$

and

$$h_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} a(\mathbf{x}_u, \mathbf{x}_v)\psi(\mathbf{x}_v)\right), \tag{7}$$

it is possible to depict that the additional term, is the presence of the attention mechanism a in the GAT architecture inside the aggregation function $\bigoplus$. Hence, considering this proposed generalization, it is utterly understandable to leverage these attention mechanisms by applying it inside of the update function present in the MPNN model that scales the output from message function $\psi(\cdot)$. Therefore, the idea of the use the attention relied exactly on the concept of exploring the attention layer simultaneously with the MSMP structure present in the baseline.

6. EXPERIMENTS

In the scope of this work, three experiments were conducted besides the baseline, considering different architecture modifications in the MSMP compared to the baseline, dif-ferent model hyperparameters, distinct training param-eters, and the number of input features for each experi-ment. Regarding these architectural changes, the first ex-periment was proposed to decrease the MAPE by using a different activation function in the MLP of the initializa-tion and readout phase. Therefore, except for the output layer of the readout phase, all activation functions of all other MLP layers were replaced by the SeLU activation function. The second experiment organization ignored the SeLU modifications and maintained the ReLU as ac-tivation functions equal to the baseline, but promoted a modification in the message phase of MSMP using the attention mechanism. Finally, the last experiments represented the final solution whose model architecture combines the SeLU modifications of the first experiment with the attention mechanism proposed in the previous one.

Moreover, when comparing the results obtained from each modification with the baseline model, the metric MAPE was used for the Loss Function (LF), whose definition is expressed by

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{\hat{y}_i - y_i}{y_i} \right|. \tag{8}$$

6.1 Hyperparameters

On hyperparameters of MSMP for each experiment, such as the embedding length for each type and the number of iterations, Table 5 summarizes the values for each one.

Table 5 – MSMP hyperparameters for each experiment.

Experiment	Iterations	Flow Emb. Length	Link Emb. Length
Baseline	8	64	64
MSMP + SeLU	8	64	64
MSMP + Attention	12	16	16
Final Solution	12	16	16

For the training parameters, there are two distinctions between the experiments, in this case, the number of epochs and the normalization approach adopted in the training process. Each experiment used the adamW as an optimizer, *Early Stoppings* and *Reduce on Plateau* algorithms with the same parameters. Thus, Table 6 summarizes these distinct parameters for each experiment.

Table 6 – Training hyperparameters for each experiment.

Experiment	Epochs	Normalization Approach
Baseline	100	*min-max*
MSMP + SeLU	150	*min-max*
MSMP + attention	150	*min-max*
Final Solution	150	z-score

Finally, regarding the input features, we considered the output of the feature selection applied in the flow-level feature type, presented in Table 4 and the traffic type; along with link-level features presented in Table 1. Therefore, the input features for each experiment are given by Table 7.

Table 7 – Subset of features used as input for each experiment.

Features	Feature Role	Unit
Link capacity	Link	Gbits/s
Flow length	Flow	flow
Average bandwidth	Flow	Mbits/s
Inter packet gap variance	Flow	ns
Link load	Link	ratio
Normalized link load	Link	ratio
Average packet size	Flow	bytes
Bitrate per burst	Flow	Mbits/s
Packet size 90 percentile	Flow	bits
Packets generated	Flow	packets/s
Constant bitrate	Flow	Mbits/s
Traffic type	Flow	String
Number of packets per burst	Flow	packets
Inter packet gap mean	Flow	ns
Inter burst gap	Flow	μs

6.2 Results

The first result is related to the experiment with the baseline model. This model achieved the LF MAPE in the last epoch equal to 41.423% in the testing dataset. In similar terms, the second experiment showed that the impact of combining a greater subset of selected features, with the SeLU improvements in the MLP of initialization and readout phase, is substantial to conducting better results with an LF MAPE of about 26.360%. A reduction, in relative terms of 36.363% in comparison to the baseline. By these results, it seems like a good approach to combine a selected and greater subset of features along a better activation function in the MLP used in both MSMP phases.

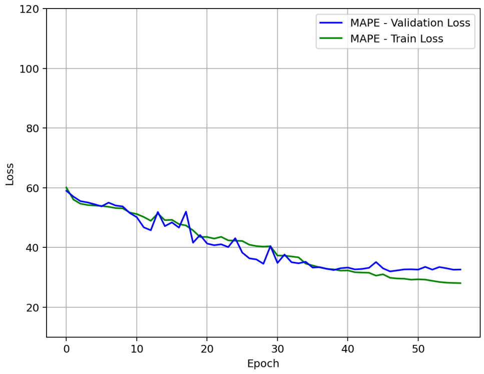
The third experiment aimed to improve the baseline using the same subset as used in the previous experiment but changing the model's hyperparameters and, instead of modifying two modules of MSMP, a modification was proposed only in the message phase using the attention mechanism layer of Equation *(5)*. This configuration led to an LF MAPE equal to 24.021%, and a reduction of 42.010%. This experiment shows that the suit MSMP hyperparameters and the use of attention mechanism, assigning different degrees of importance to each feature, are enough to overcome the improvements generated by the SeLU in the aforementioned MLPs.

Finally, the last experiments aimed to explore the results of both improvements created by the previous experiments in a combination that merges the attention mechanism in the message phase and the use of the SeLU activation functions and using a different normalization feature approach, in this case, the z-score. And, indeed, this combination generated a final result that achieved an LF MAPE of about 20.001%, and with a reduction of 51.715%.

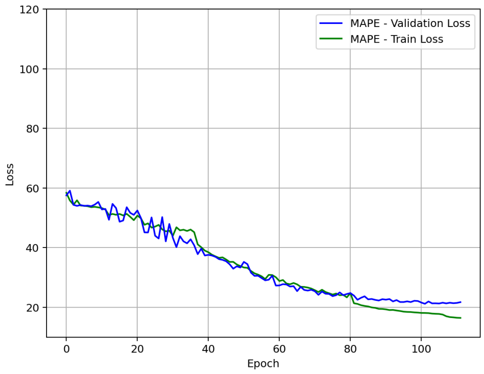
Taking into account these four experiments, the model behaviors throughout the training and validation process, in terms of MAPE per epoch, are summarized in Fig. 6. In addition, in terms of predicted delay in the test dataset, Fig. 7 shows a relationship based on delays predicted in each experiment by the known ground truth. In this sense, each plot shows a heatmap, where how much more distant, a point is from the red line, the lighter their color will be. Consequently, the ideal result, MAPE equal to 0 %, is all scatter points concentrated on the identity red line.

7. CONCLUSIONS AND FUTURE WORK

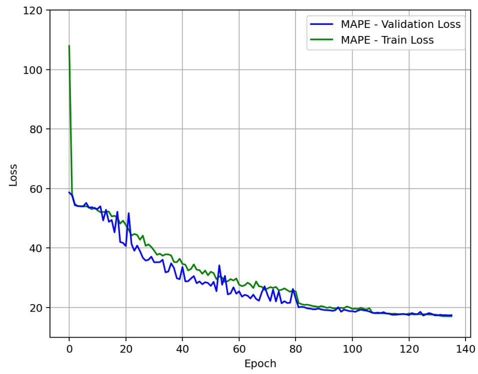
In this paper, we show an approach that uses GNN based on MSMP modules that operate in a real network dataset to estimate the mean per-flow delay in a given network communication modeled as a graph. It was used as a baseline modified RouteNet-Fermi proposed by the GNNet Challenge 2023 organizers. In this sense, to improve this baseline model, making the MAPE decrease, three experiments with modifications in all modules of the MSMP were proposed. Therefore, in comparison with the baseline, the experiment that demonstrated a better performance utilized the z-score approach as the feature normalization, an attention mechanism layer in the message phase of the MSMP, and the SeLU activation function in specific MLP layers in the initialization and readout phase. With this architecture, we were able to achieve a MAPE of about 20.001%, in the test dataset, for predicting the



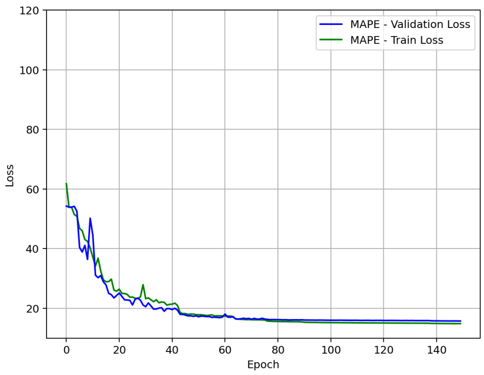
(a) Baseline LF MAPE curves.



(b) MSMP+SeLU LF MAPE curves.

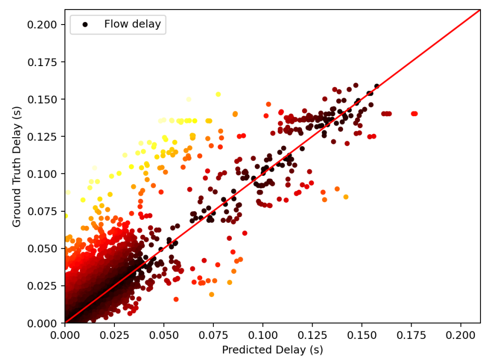


(c) MSMP+attention LF MAPE curves.

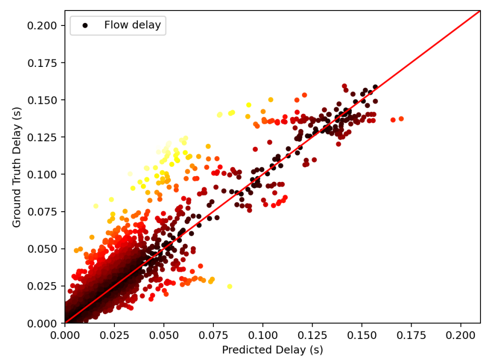


(d) Final solution LF MAPE curves.

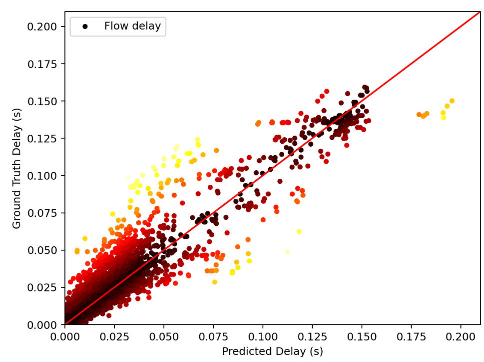
Figure 6 – MAPE LF curves in training and validation for all experiments.



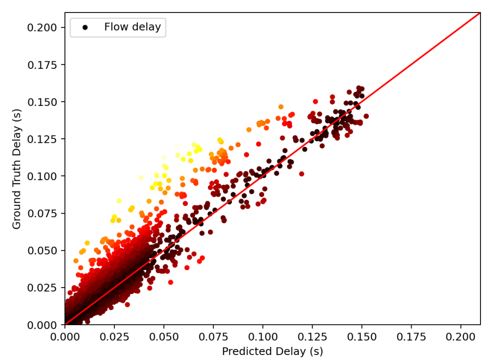
(a) Predicted delay made by the baseline model.



(b) Predicted delay made by the MSMP+SeLU model.



(c) Predicted delay made by the MSMP+attention model.



(d) Predicted delay made by the final solution model.

Figure 7 – Predicted delay in test dataset for all experiments.

mean per-flow delay.

Regarding future work, it is possible to expand estimating the mean per-flow delay by proposing different architectures for MSMP. The modularity of this model allows the use of different deep-learning architectures, to increase the performance of the model. Furthermore, it is also interesting to extend this work for the estimation of different QoS parameters, such as jitter and packet loss. In this sense, this type of estimation is already done considering synthetic datasets, also making use of the RouteNet-Fermi model, but in the literature, there is a lack of conclusions in terms of scalability and generality of a GNN model taking into account real network communication environments considering these other metrics.

ACKNOWLEDGEMENTS

This work was supported by Ericsson Telecomunicações S.A, the Brazilian Network for Education and Research (RNP), the Project Smart 5G Core And MUltiRAn Integration (SAMURAI) (MCTIC/CGI.br/FAPESP Grant 2020/05127-2) and by the Brazilian National Council for Research and Development (CNPq), Project 405111/2021-5.

REFERENCES

[1] Recommendation ITU-T Y.3090 (2022). *Digital twin network – Requirements and architecture*.

[2] Paul Almasan, Miquel Ferriol-Galmés, Jordi Paillisse, José Suárez-Varela, Diego Perino, Diego López, Antonio Agustin Pastor Perales, Paul Harvey, Laurent Ciavaglia, Leon Wong, Vishnu Ram, Shihan Xiao, Xiang Shi, Xiangle Cheng, Albert Cabellos-Aparicio, and Pere Barlet-Ros. "Network Digital Twin: Context, Enabling Technologies, and Opportunities". In: *IEEE Communications Magazine* 60.11 (2022), pp. 22–27. DOI: 10.1109/MCOM.001.2200012.

[3] Krzysztof Rusek, Jose Suarez-Varela, Paul Almasan, Pere Barlet-Ros, and Albert Cabello. "RouteNet: Leveraging Graph Neural Networks for Network Modeling and Optimization in SDN". In: *IEEE Journal on Selected Areas in Communications* PP (June 2020), pp. 1–1. DOI: 10.1109/JSAC.2020.3000405.

[4] Yifei Jin, Marios Daoutis, Sarunas Girdzijauskas, and Aristides Gionis. "Open World Learning Graph Convolution for Latency Estimation in Routing Networks". In: July 2022, pp. 1–8. DOI: 10.1109/IJCNN55064.2022.9892952.

[5] Albert Mestres, Eduard Alarcón, Yusheng Ji, and Albert Cabello. "Understanding the Modeling of Computer Network Delays using Neural Networks". In: (July 2018).

[6] Martin Happ, Matthias Herlich, Christian Maier, Jia Lei Du, and Peter Dorfinger. "Graph-neural-network-based delay estimation for communication networks with heterogeneous scheduling policies". In: *ITU Journal on Future and Evolving Technologies* 2 (June 2021). DOI: 10.52953/TEJX5530.

[7] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. "The Graph Neural Network Model". In: *IEEE Transactions on Neural Networks* 20.1 (2009), pp. 61–80. DOI: 10.1109/TNN.2008.2005605.

[8] Debjit Sarkar, Sourodeep Roy, Samir Malakar, and Ram Sarkar. "A modified GNN architecture with enhanced aggregator and Message Passing Functions". In: *Engineering Applications of Artificial Intelligence* 122 (2023), p. 106077. ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2023.106077. URL: https://www.sciencedirect.com/science/article/pii/S0952197623002610.

[9] Miquel Ferriol-Galmés, Jordi Paillisse, José Suárez-Varela, Krzysztof Rusek, Shihan Xiao, Xiang Shi, Xiangle Cheng, Pere Barlet-Ros, and Albert Cabellos-Aparicio. "RouteNet-Fermi: Network Modeling With Graph Neural Networks". In: *IEEE/ACM Transactions on Networking* 31.6 (Dec. 2023), pp. 3080–3095. ISSN: 1558-2566. DOI: 10.1109/tnet.2023.3269983. URL: http://dx.doi.org/10.1109/TNET.2023.3269983.

[10] Taisei Suzuki, Yuichi Yasuda, Ryo Nakamura, and Hiroyuki Ohsaki. "On Estimating Communication Delays using Graph Convolutional Networks with Semi-Supervised Learning". In: *2020 International Conference on Information Networking (ICOIN)*. 2020, pp. 481–486. DOI: 10.1109/ICOIN48656.2020.9016603.

[11] Paul Almasan, José Suárez-Varela, Krzysztof Rusek, Pere Barlet-Ros, and Albert Cabellos-Aparicio. "Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case". In: *Computer Communications* 196 (2022), pp. 184–194. ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2022.09.029. URL: https://www.sciencedirect.com/science/article/pii/S0140366422003784.

[12] András Varga and Rudolf Hornig. "An overview of the OMNeT++ simulation environment". In: Jan. 2008, p. 60. DOI: 10.1145/1416222.1416290.

[13] *TRex - Realistic Traffic Generator*. URL: https://trex-tgn.cisco.com/.

[14] *DPDK Project*. URL: https://www.dpdk.org/.

[15] Yizhou Sun and Jiawei Han. “Mining Heterogeneous Information Networks: A Structural Analysis Approach”. In: *SIGKDD Explor. Newsl.* 14.2 (Apr. 2013), pp. 20–28. ISSN: 1931-0145. DOI: 10.1145/2481244.2481248. URL: https://doi.org/10.1145/2481244.2481248.

[16] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. “Neural Message Passing for Quantum Chemistry”. In: *International Conference on Machine Learning*. 2017. URL: https://api.semanticscholar.org/CorpusID:9665943.

[17] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. “Gated Graph Sequence Neural Networks”. In: *arXiv: Learning* (2015). URL: https://api.semanticscholar.org/CorpusID:8393918.

[18] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. “Order Matters: Sequence to sequence for sets”. In: *CoRR* abs/1511.06391 (2015). URL: https://api.semanticscholar.org/CorpusID:260429228.

[19] Kristof T. Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R. Müller, and Alexandre Tkatchenko. “Quantum-chemical insights from deep tensor neural networks”. In: *Nature Communications* 8.1 (Jan. 2017). ISSN: 2041-1723. DOI: 10.1038/ncomms13890. URL: http://dx.doi.org/10.1038/ncomms13890.

[20] Ingrid Nascimento, Ricardo Souza, Silvia Lins, Andrey Silva, and Aldebaro Klautau. “Deep Reinforcement Learning Applied to Congestion Control in Fronthaul Networks”. In: *2019 IEEE Latin-American Conference on Communications (LATINCOM)*. 2019, pp. 1–6. DOI: 10.1109/LATINCOM48065.2019.8937857.

[21] C Selvi and Editor - Scitech Conferences Isarp. “Impact and limitations of artificial intelligence in cyber security awareness Introduction”. In: *European Chemical Bulletin* 12 (Sept. 2023), pp. 8820–8829. DOI: 10.48047/ecb/2023.12.8.717.

[22] Cleverson Veloso Nahum, Lucas De Nóvoa Martins Pinto, Virgínia Brioso Tavares, Pedro Batista, Silvia Lins, Neiva Linder, and Aldebaro Klautau. “Testbed for 5G Connected Artificial Intelligence on Virtualized Networks”. In: *IEEE Access* 8 (2020), pp. 223202–223213. DOI: 10.1109/ACCESS.2020.3043876.

[23] N. Hoque, D.K. Bhattacharyya, and J.K. Kalita. “MIFS-ND: A mutual information-based feature selection method”. In: *Expert Systems with Applications* 41.14 (2014), pp. 6371–6385. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2014.04.019. URL: https://www.sciencedirect.com/science/article/pii/S0957417414002164.

[24] JC Ang, Andri Mirzal, Habibollah Haron, and Haza Nuzly Abdull Hamed. “Supervised, Unsupervised, and Semi-Supervised Feature Selection: A Review on Gene Selection”. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 13 (Sept. 2015), pp. 1–1. DOI: 10.1109/TCBB.2015.2478454.

[25] Noelia Sánchez-Maroño, Amparo Alonso-Betanzos, and María Tombilla-Sanromán. “Filter Methods for Feature Selection – A Comparative Study”. In: Dec. 2007, pp. 178–187. ISBN: 978-3-540-77225-5. DOI: 10.1007/978-3-540-77226-2_19.

[26] Roberto Battiti. “Using Mutual Information for Selecting Features in Supervised Neural Net Learning”. In: *Neural Networks, IEEE Transactions on* 5 (Aug. 1994), pp. 537–550. DOI: 10.1109/72.298224.

[27] Isabelle Guyon and André Elisseeff. “An Introduction of Variable and Feature Selection”. In: *J. Machine Learning Research Special Issue on Variable and Feature Selection* 3 (Jan. 2003), pp. 1157–1182. DOI: 10.1162/153244303322753616.

[28] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. “Estimating mutual information”. In: *Physical Review E* 69.6 (June 2004). ISSN: 1550-2376. DOI: 10.1103/physreve.69.066138. URL: http://dx.doi.org/10.1103/PhysRevE.69.066138.

[29] Brian C. Ross. “Mutual Information between Discrete and Continuous Data Sets”. In: *PLOS ONE* 9 (Feb. 2014), pp. 1–5. DOI: 10.1371/journal.pone.0087357. URL: https://doi.org/10.1371/journal.pone.0087357.

[30] Uchechukwu Fortune Njoku, Alberto Abelló Gamazo, Besim Bilalli, and Gianluca Bontempi. “Wrapper methods for multi-objective feature selection”. In: *26th International Conference on Extending Database Technology (EDBT 2023): Ioannina, Greece, March 28-March 31: proceedings*. OpenProceedings. 2023, pp. 697–709.

[31] Lucas B.V. de Amorim, George D.C. Cavalcanti, and Rafael M.O. Cruz. “The choice of scaling technique matters for classification performance”. In: *Applied Soft Computing* 133 (Jan. 2023), p. 109924. ISSN: 1568-4946. DOI: 10.1016/j.asoc.2022.109924. URL: http://dx.doi.org/10.1016/j.asoc.2022.109924.

[32] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition*. O’Reilly Media, Inc., Sept. 2019, pp. 334–338. ISBN: 9781492032649.

[33] Gianni Brauwers and Flavius Frasincar. “A General Survey on Attention Mechanisms in Deep Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 35.4 (Apr. 2023), pp. 3279–3298. ISSN: 1558-2191. DOI: 10.1109/TKDE.2021.3126456.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. “Attention is All You Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS’17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.

[35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. “Graph Attention Networks”. In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=rJXMpikCZ.

[36] Shaked Brody, Uri Alon, and Eran Yahav. “How Attentive are Graph Attention Networks?” In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=F72ximsx7C1.

[37] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Velivckovi’c. “Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges”. In: *ArXiv* abs/2104.13478 (2021). URL: https://api.semanticscholar.org/CorpusID:233423603.

[38] Petar Veličković. “Message passing all the way up”. In: *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*. 2022. URL: https://openreview.net/forum?id=Bc8GiEZkTe5.

[39] Josephine Thomas, Alice Moallemy-Oureh, Silvia Beddar-Wiesing, and Clara Holzhüter. “Graph Neural Networks Designed for Different Graph Types: A Survey”. In: *Transactions on Machine Learning Research* (2023). ISSN: 2835-8856. URL: https://openreview.net/forum?id=h4BYtZ79uy.

AUTHORS

Cláudio Modesto received a B.Sc. degree in computer engineering from the Federal University of Pará (UFPA), Belém, Pará, Brazil in 2024. He has been part of the Research and Development Center for Telecommunications, Automation, and Electronics (LASSE) since 2023. He recently participated in the ITU/GNNet Challenge 2023 as leader of the m0b1us team, which won first place in the competition. His current research interests include signal processing, machine and deep learning applied to graphs.

Rebecca Aben-Athar is an undergraduate electrical engineering student at the Federal University of Pará (UFPA), Belém, Pará, Brazil. She has been part of the Research and Development Center for Telecommunications, Automation, and Electronics (LASSE) since 2021. She recently participated in the ITU/GNNet Challenge 2023 as a member of the m0b1us team, which won first place in the competition. Her current research interests include machine learning, deep learning, and cloud computing.

Andrey Silva received a B.Sc. degree in computer engineering, and an M.Sc. degree in electrical engineering from the Federal University of Pará, Brazil, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree in electrical engineering. His current research interests include machine learning, deep learning, and digital signal processing.

Silvia Lins received a B.Sc. degree in computer engineering, an M.Sc. degree in electrical engineering, and a Ph.D. from the Federal University of Pará (UFPA), Brazil. She is currently an experienced researcher in the network architecture and protocols area with Ericsson Brazil. Her current research involves machine learning for 5G and beyond network scenarios and applications. In the past, she also worked with information-centric networks, transport networks, and 3G/4G mobile networks.

Glauco Gonçalves has been an associate professor at the Federal University of Pará since 2021. He was previously a professor at the Federal Rural University of Pernambuco (2013-2021) where he worked developing research and teaching. He has a Ph.D (2012) and a master's degree (2007) in computer science from the Informatics Center of the Federal University of Pernambuco. Glauco received his bachelor's degree in computer science from the Federal University of Pará in 2005. He works in the area of computer science applying computational modeling methods in different areas such as 6G networks, cloud computing, and the Internet of Things.

Aldebaro Klautau received a bachelor's (Federal University of Pará, UFPA, 1990), M.Sc. (Federal Univ. of Santa Catarina, UFSC, 1993), and Ph.D. degrees (University of California at San Diego, UCSD, 2003) in electrical engineering. He is a full professor at UFPA, where he is the ITU Focal Point and coordinates the LASSE Research Group. He is a researcher at CNPq, Brazil, and a senior member of the IEEE and the Brazilian Telecommunications Society (SBrT). His work focuses on machine learning and signal processing for communications.