Feature Name Decoration Enhanced Router Performance Prediction by a Large Language Model

Kyota Hattori
NTT Network Service Systems Laboratories
NTT Corporation
Tokyo, Japan
kyota.hattori@ntt.com

Tomohiro Korikawa
NTT Network Service Systems Laboratories
NTT Corporation
Tokyo, Japan
tomohiro.korikawa@ntt.com

Chikako Takasaki
NTT Network Service Systems Laboratories
NTT Corporation
Tokyo, Japan
chikako.takasaki@ntt.com

Abstract— **Future carrier networks for 6G are expected to verify performance across heterogeneous networks integrating multiple technologies. This integration requires effective verification of network performance under unpredictable conditions. In response, a node modeling method has been proposed for digitally evaluating the performance of actual network nodes. However, creating accurate node models is challenging due to the high cost and difficulty of collecting extensive real-world data under various environmental conditions. Therefore, the objective of this study is to improve the performance of actual network nodes using limited real-world datasets. We investigate the potential of natural language processing technologies to improve the accuracy of router performance estimation. In this paper, we propose a Feature Name Decoration (FND) method using Large Language Models (LLMs) to predict actual router metrics. The FND can help clarify the relationships between specific features, such as router settings and traffic conditions, and their impact on router metrics. The results show that the proposed FND improves the estimation accuracy of actual router performance metrics, including throughput, packet loss rate, and packet delay.**

Keywords— *Router metrics prediction, Feature Name Decoration, Large language models*

I. Introduction

The development of 6th-generation (6G) networks will be driven by rapid technological advances, leading to a paradigm shift towards advanced heterogeneous networks that incorporate multiple technologies [1]. This transition includes 5th-generation (5G) networks, unmanned aerial vehicles, vehicle-to-vehicle communications, and satellite networks. The integration of these elements into 6G networks introduces a variety of operational scenarios that require network configurations to meet their mobility and connectivity needs. This diversity, in turn, introduces challenges of unprecedented complexity and scale. Therefore, future carrier networks will need to verify interoperability and performance across different networks to address this challenge effectively.

To address the need for effective network performance verification, a node modeling method using machine learning techniques has been introduced to digitally evaluate network node performance [2]. This approach enables network operators to design long-term network planning to evaluate whether operator-designed network settings meet service requirements in advance. The node modeling is based on training datasets, including node settings, traffic conditions, and corresponding network metrics. This training is essential for understanding network node behavior and accurately estimating real-world node performance. However, the accuracy of node modeling requires extensive labeled training datasets from real networks. This leads to increasing the cost to collect extensive data from real networks with different traffic patterns. Thus, improving the accuracy for limited real-world datasets remains a critical issue.

To address this issue, it is essential to leverage not only data obtained from the real network, but also knowledge from other sources outside the network to improve the accuracy of router metric predictions. One promising candidate to assist in this task is the application of natural language technology. In the field of natural language processing, large-scale language models (LLMs) are rapidly evolving. LLMs are trained on large amounts of text data to learn the patterns, structures, and nuances of human language. This training enables them to perform tasks such as text generation, translation, and other language-related tasks with a high level of proficiency. In addition, LLMs have been shown to be effective at performing out-of-domain tasks, including domain shifts [3]. This capability is valuable in real-world scenarios where labeled data is sparse. Their flexibility enables the LLMs to quickly adapt to new tasks that involve reasoning, especially when guided by well-designed prompts. However, a key issue is their reliance on pre-trained natural language processing (NLP) knowledge, which limits their adaptability to new or evolving information [4], such as the network domain task in this study.

Given the above background, we propose a Feature Name Decoration (FND) method that utilizes the LLM to improve the accuracy of router performance predictions for limited real-world datasets. The FND method leverages both the operator's knowledge and the LLM's NLP understanding capabilities.

To the best of our knowledge, there are few or no existing studies that use an LLM to predict actual network node metrics. This study focuses on router modeling as a fundamental step, as routers will continue to play a key role in connecting heterogeneous networks in the future.

II. Related work

In this study, the objective is to estimate router metrics based on node settings and traffic conditions in tabular data with limited datasets. Historically, tree-based techniques [5] have been the traditional approach for estimation with tabular data. However, these methods struggle with limited datasets, primarily due to the risk of overfitting. An alternative strategy for limited data is deep reinforcement learning [6], which is

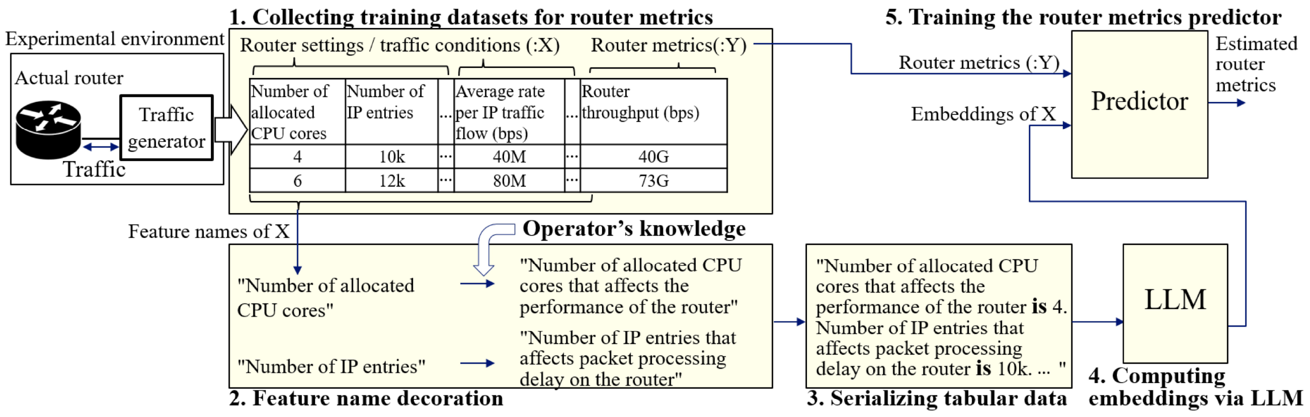
1. Collecting training datasets for router metrics
Experimental environment
Actual router
Traffic generator
Traffic
Router settings / traffic conditions (:X)
Router metrics(:Y)
Number of allocated CPU cores
Number of IP entries
Average rate per IP traffic flow (bps)
Router throughput (bps)
4
10k
40M
40G
6
12k
80M
73G
Feature names of X
Operator's knowledge
"Number of allocated CPU cores"
"Number of IP entries"
"Number of allocated CPU cores that affects the performance of the router"
"Number of IP entries that affects packet processing delay on the router"
2. Feature name decoration
"Number of allocated CPU cores that affects the performance of the router is 4. Number of IP entries that affects packet processing delay on the router is 10k. ... "
3. Serializing tabular data
LLM
4. Computing embeddings via LLM
5. Training the router metrics predictor
Router metrics (:Y)
Embeddings of X
Predictor
Estimated router metrics

Fig. 1. Architecture of the proposed router metrics predictor using an LLM.

trained in simulated environments for real-world applications. However, large differences between simulators and actual environments lead to the learning of incorrect policies, which degrades performance. For example, carrier networks require network capacity over 100 Gbps. In contrast, simulators, constrained by computational time, can only manage a few Gbps using standard server resources [7]. This limitation leads to inaccurate policies and reduced performance. Meanwhile, a transfer learning approach [8] has been proposed to improve the estimation accuracy for the behavior in the real-world in limited datasets by pre-training on simulation domains and fine-tuning, e.g., network traffic prediction [9]. However, parameter adjustments for transfer and fine-tuning require a detailed analysis of task types and similarities, which increases the operational cost of model development [10].

To reduce the complexity and better understand the relationship between each task, one strategy is to use an LLM [3,4] that has extensive natural language knowledge. One approach using the LLM is to generate synthetic data for the target domain to augment algorithm training [11]. Although the LLM is promising for tasks with coarse control factors, it remains challenging for tasks that require fine-grained control, such as numerical data generation. Approaches that integrate LLMs into tabular data estimation effectively use their extensive linguistic knowledge and meta-information from column names [12, 13]. The fine-tuning technique has been adapted to tabular data estimation based on generative language models [12]. Meanwhile, TabLLM [13] refines pre-trained language models to estimate results for tabular datasets. However, the effectiveness of these refined LLMs depends on various factors such as the task description, the sequence of training examples, the hyperparameters, and the learning algorithm itself [14]. Therefore, these approaches require careful selection of training data and fine-tuning of the model, both of which are important for robust performance, which results in higher operational costs.

Our work differs from previous studies by avoiding the complex fine-tuning process. The novelty of our proposed method lies in the integration of the operator's knowledge accumulated by network operators into the LLM, which enhances the feature names with their insights.

III. Problem Formulation

The objective of this study is to improve the estimation accuracy of actual router performance metrics, specifically packet throughput (P_{th}), packet loss rate (P_{loss}), and packet delay (P_{delay}). We investigate the effectiveness of decorating feature names for router metrics with the LLM to establish the relationships between specific features, such as router settings and traffic conditions, and their impact on router metrics. We use a tabular dataset where each row represents a single dataset that serves as a distinct data point. Then, each column contains the features, including router settings, traffic conditions, and corresponding router metrics as labeled data. Here, these datasets are represented as $(\mathbb{X}, \mathbb{Y}) = \{(X_i, Y_i)\}_{i=1}^{n}$, where X_i is the i-th observation described by a set of j features represented by $(X_i^1, X_i^2, \dots, X_i^j)$, and Y_i denotes the corresponding router metrics. Each feature name, represented as a column name, is defined as $F = \{f^1, \dots, f^j\}$. These f^j labels typically correspond to natural language expressions such as "Number of physical ports". Then, f^j is combined with X_i^j to form a linguistic input to the LLM. The LLM then generates embeddings for the corresponding linguistic input. The embeddings are vector representations that facilitate the model's understanding, including the feature relationships. Here, the generated embeddings are defined as e. Finally, a predictor for the router metrics is generated as follows: $\hat{Y}^{(m)} = P^{(m)}(e)$, where the function $P^{(m)}$ represents the regression model for the actual router metrics $m = \{P_{\text{th}}, P_{\text{loss}},$ and $P_{\text{delay}}\}$, $\hat{Y}^{(m)}$ denotes each estimated metric for m.

IV. Router Metrics Prediction Using LLM

Fig. 1 shows the architecture of the proposed actual router performance estimator using an LLM. The proposed method is characterized by using FND to clarify the relationships between the router metrics. Specifically, it integrates the operator's knowledge of router settings and hardware processes, which are related to performance metrics, into the LLM to improve both the understanding and accuracy of metric estimation. This approach surpasses traditional regression methods that rely solely on numerical inputs by using the natural language processing capabilities of LLMs to interpret the relationships between features and router metrics. The specific procedure of the proposed method is outlined in the following five steps, which are detailed in Algorithm 1.

1) Collecting training datasets for router metrics

The algorithm starts by collecting training datasets of router metrics from an actual router. To obtain the router

Algorithm 1: Router performance estimator with FND

```
Input: $\mathcal{D} = (\mathbb{X}, \mathbb{Y})$: Tabular data, $F = \{f^1, \ldots, f^j\}$: $j$
    dimensional feature names, FND: feature name
    decoration function, SL: serialization function
Output: $R^{2^{(m)}}$                    ▷ Output $R^2$ for router metrics $m$
1  $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}} \leftarrow \text{Split}(\mathcal{D})$ ▷ Split data into training and test sets
2  for each router metric $m$ do
3      for $i$-$th$ data $\mathbb{X},\mathbb{Y} \in \mathcal{D}_{\text{train}}$ do
4          for $j$-$th$ feature $\in F$ do
5              $f^{j'} \leftarrow \text{FND}(f^j)$   ▷ Apply feature name decoration
6              $T_i \leftarrow T_i + \text{SL}(f^{j'}, X_i^j)$   ▷ Build a serialized text
           end
7          $e_i \leftarrow \text{LLM}(T_i)$   ▷ Generate embeddings via LLM
8          Train $P^{(m)}$ on $e_i$ and $Y_i$   ▷ Train a predictor for $m$
       end
9      for $i$-$th$ data $\mathbb{X},\mathbb{Y} \in \mathcal{D}_{\text{test}}$ do
10         for $j$-$th$ feature $\in F$ do
11             $T_i \leftarrow T_i + \text{SL}(f^{j'}, X_i^j)$   ▷ Build a serialized text
           end
12         $e_i \leftarrow \text{LLM}(T_i)$   ▷ Generate embeddings via LLM
13         $\hat{Y}_i^{(m)} \leftarrow P^{(m)}(e_i)$   ▷ Predict $\hat{Y}$ for $m$
       end
14     $R^{2^{(m)}} \leftarrow R^2(\hat{Y}_{\text{test}}^{(m)}, Y_{\text{test}}^{(m)})$   ▷ Calculate $R^2$ for $m$
   end
```

metrics, including P_{th}, P_{loss}, and P_{delay}, a traffic generator is used to measure these metrics while changing the router settings and the amount of input traffic. As a result, we obtain tabular data consisting of router settings, traffic conditions, and the corresponding router metrics with their feature names.

2) Feature name decoration

The proposed Feature Name Decoration (FND) employs an enhancement technique that enriches the descriptive quality of feature names in tabular data. Based on the knowledge of network operators, the FND refines feature names to more accurately convey their roles and effects. This knowledge provides the relationships between router configurations and input traffic, as well as their influence on router metrics, to facilitate a finer mapping of feature interactions and router performance. For example, "Number of IP entries" is modified to "Number of IP entries that affect packet processing delay on the router," following the FND methodology. By incorporating these insights into feature names, FND helps LLMs generate embeddings that incorporate the interrelationships of each feature observed in network operations. In the proposed method, FND is assumed to have a predetermined modifier for each feature.

3) Serializing tabular data

To prepare input data for LLM, tabular data is converted to a natural language format through a process known as serialization [13]. We introduce the serialization function $\text{SL}(f^{j'}, X_i^j)$, which takes the decorated j-th feature name $f^{j'}$ generated by $\text{FND}(f^j)$ and the corresponding feature value at the i-th data point X_i^j as inputs to generate a serialized text T_i. Specifically, T_i is generated by inserting "is" between the decorated feature name and its value, forming a structure such as "<decorated feature name> is <feature value>". This process generates statements equal to the number of features, and these statements are then concatenated line by line to form input prompts for the LLM.

TABLE I. EXPERIMENTAL CONDITIONS

Category	Feature name	Values
Router settings	Number of physical ports	2 to 8
	Number of IP entries	0.4 k to 770 k
	Number of allocated CPU cores	4 to 36
	Memory allocation size (GB)	12 to 64
Traffic conditions	IP packet size (Bytes)	46 to 1500
	Number of IP traffic flows	0.4 k to 770 k
	Average rate per IP traffic flow (bps)	64 k to 80 M

4) Computing embeddings via LLM

In this step, the LLM leverages advanced language understanding capabilities to interpret the serialized text and transform it into the embeddings e, which are vector representations of the textual feature relationships. These embeddings provide a richer input to the regressor than traditional numerical data formats alone, allowing it to analyze and exploit natural language-derived relationships. Consequently, the use of these embeddings allows the regressor to analyze data containing natural language-derived feature relationships, potentially improving estimation accuracy. In this study, the LLM is used as a black box, with no modifications to the pre-trained default parameters.

5) Training the router metrics predictor using embeddings from LLM

Finally, the actual router performance regressor $G^{(m)}$ is trained using e and the measured router metrics as the dependent variable Y. This regressor estimates P_{th}, P_{loss}, and P_{delay}, which are evaluated by the coefficient of determination [15], $R^2 = 1 - \sum(Y_i - \hat{Y}_i)^2 / \sum(Y_i - \bar{Y})^2$, where $\hat{Y}_i$ and $\bar{Y}$ are the estimated value for the i-th observation and the mean measured value of the actual router metric y, respectively. $R^2 = 1$ means that the model fits the datasets perfectly.

V. EVALUATION & RESULTS

We evaluated the accuracy of actual router performance estimation using the proposed FND with LLM and compared it to regression models both without and with LLM.

A. Experimental setup

1) Data collection conditions for training datasets

First, to generate tabular data, we collected actual router metrics data under specific experimental conditions using four types of virtual routers. These routers include Cisco CSR 1000V [16], Juniper vMX [17], Vector Packet Processor [18], and Kamuee router [19], all running on an x86-based server equipped with Xeon E5-2697 18-core 2.30 GHz CPUs and 192 GB RAM with 8 SFP+ ports. We measured the performance of these routers in the packet forwarding process in a laboratory environment to acquire training datasets under the conditions listed in Table I. These conditions were designed to evaluate the impact of traffic aggregation on the relationship between router settings and traffic conditions, focusing on the increase in packet processing delay and packet losses due to queue overflow. The Keysight Ixia platform, equipped with 8 SFP+ ports, was used to generate the traffic. Traffic was sent to a single output port at a constant rate from each port with a fixed IP packet size. The

TABLE II . FEATURE NAME PATTERNS EVALUATED IN THIS STUDY

Normal expression	Number of physical ports	Number of IP entries	Number of allocated CPU cores	Memory allocation size	IP packet size	Number of IP traffic flows	Average rate per IP traffic flow
Proposed expression applied FND	Number of physical ports that affects the load on the router	Number of IP entries that affects packet processing delay on the router	Number of allocated CPU cores that affects the performance of the router	Memory allocation size that affects packet loss rate	IP packet size that affects the load on the router as it gets smaller	Number of IP traffic flows that affects the load on the router	Average rate per IP traffic flow that affects the load on the router

TABLE III. EVALUATION RESULTS OF R^2 FOR EACH ROUTER METRIC

Method	**Actual router metrics** P_{th}	P_{loss}	P_{delay}
GBR w/o LLM	0.87 (0.02)	0.79 (0.05)	0.50 (0.15)
XGBoost w/o LLM	0.88 (0.02)	0.82 (0.02)	0.63 (0.11)
GBR w/ BERT	0.90 (0.01)	0.87 (0.02)	0.79 (0.10)
GBR w/ DistilBERT	0.89 (0.01)	0.86 (0.03)	0.77 (0.12)
XGBoost w/ BERT	0.90 (0.02)	0.87 (0.02)	0.78 (0.13)
XGBoost w/ DistilBERT	0.90 (0.02)	0.86 (0.02)	0.75 (0.16)
XGBoost w/ FND and BERT **(ours)**	**0.91** (0.01)	**0.90** (0.02)	**0.93** (0.02)
XGBoost w/ FND and DistilBERT **(ours)**	**0.91** (0.01)	**0.90** (0.02)	**0.93** (0.02)

parameters, including IP packet size, the number of physical ports, and average rate, were varied within the ranges shown in Table I. A total of 930 samples were collected from these routers. In this scenario, we measured P_{loss} (per second), P_{delay} (maximum per second), and P_{th} (bits per second) for each router setting and traffic condition. The training datasets include two main components, router settings and traffic conditions. Router settings include the number of physical ports, IP entries, allocated CPU cores, and memory allocation size. Traffic conditions include IP packet size, the number of IP traffic flows, and average rate per IP traffic flow.

2) LLM conditions

We used two models from the HuggingFace library, BERT [20] and DistilBERT [21], to evaluate the impact of LLM parameter size on router performance estimation. BERT is a well-established model featuring 12 transformer layers with 768 hidden dimensions, and a total of 110 million parameters. DistilBERT is a simpler model maintaining the same hidden dimensions but has only 6 layers with a total of approximately 66 million parameters. This selection allows us to explore how variations in model complexity affect the router performance estimation. The feature names used in our study are shown in Table II. This table compares two types of expressions, the "Normal expression" used as standard input for the LLM and our "Proposed expression" with the FND.

3) Training conditions for regression models

We evaluated the accuracy of estimating actual router performance using three methods: the regression models without LLM, those with LLM, and the model with FND (ours). We used PyCaret [22] to develop the regression models. PyCaret facilitates the selection of an appropriate model and the adjustment of the hyperparameters. Using PyCaret, we selected the two best performing models on the given datasets, eXtreme Gradient Boosting (XGBoost) [5] and Gradient Boost Regressor (GBR) [23]. We configured XGBoost and GBR with a maximum depth of 9, 200 estimators, and a learning rate of 0.001. These processes were performed using an Nvidia Tesla P100 card with 16 GB of memory. The R^2 was evaluated for each model using 10-fold cross validation. The mean and standard deviation of the estimation errors are computed for each of the 10-fold cross-validation runs.

B. Accuracy of predicting actual router metrics

Table III shows the R^2 results, including both mean and standard deviation for P_{th} , P_{loss} , and P_{delay} (standard deviations are shown in parentheses). The regression models, GBR and XGBoost, without LLM exhibit R^2 values of 0.87 and 0.88 for P_{th}, 0.79 and 0.82 for P_{loss}, and 0.50 and 0.63 for P_{delay}, respectively. The improvements in estimation accuracy for P_{delay} are modest because it is affected by fluctuations such as packet jitter. Alternatively, the integration of LLM in both the GBR and XGBoost models shows improvements in R^2 for all metrics compared to the models without the LLMs. This improvement is attributed to the LLM's capability to capture the feature relationships of the router metrics. In this scenario, no significant differences in accuracy are observed between different LLMs or regressor models. To highlight a subtle difference, the combination of XGBoost with BERT performs relatively well. However, there is still room for improvement in the estimation of P_{delay}, where R^2 only increases from 0.63 to 0.78 without considering the uncertain aspects of the network delay. On the other hand, our proposed method combining FND and LLM with XGBoost provides further improvement, with R^2 from 0.90 to 0.93 for P_{th}, from 0.87 to 0.90 for P_{loss}, and especially from 0.78 to 0.92 for P_{delay}, with smaller standard deviations compared to XGBoost with BERT. For P_{delay}, the improvement in R^2 reached up to 18%. These improvements are consistent for both BERT and DistilBERT models.

Fig. 2 shows residual plots for two methods that evaluate actual router metrics, XGBoost with BERT and the proposed method, to identify model inaccuracies such as poor fits or the presence of outliers. The residual plots for the proposed method show tighter clustering around the zero line than those of XGBoost with BERT, indicating fewer errors. These improvements underscore the effectiveness of using FND to clarify the interactions between features, allowing for more accurate regression analysis. This suggests that the proposed method could provide an improved model that accounts for the relationships between the features and the router metrics, potentially improving accuracy over the conventional method.

C. Future Challenges

This study evaluated the use of LLM for a single router scenario. However, future scenarios will require applications for more complex conditions, such as end-to-end wired and wireless scenarios. For these complex scenarios, it is important to evaluate the applicability of small-scale LLMs, which can reduce the operating cost of LLMs without compromising performance [24]. In addition, the proposed

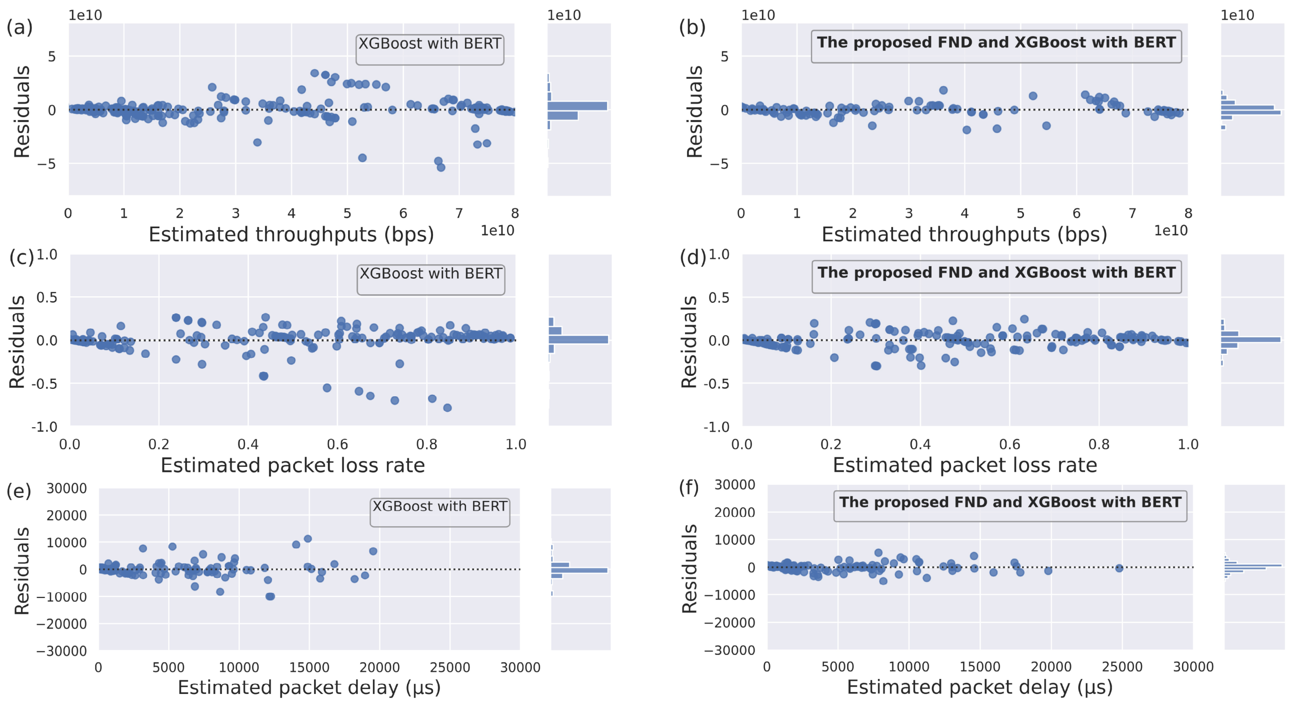

Fig. 2. Residual plots for the predicted P_{th} in (a) and (b), the predicted P_{loss} in (c) and (d), and the predicted P_{delay} in (e) and (f). Plots (a), (c), and (e) use XGBoost with BERT, while (b), (d), and (f) use the proposed FND and XGBoost with BERT.

FND is based on insights and knowledge gained from network operations. Currently, it relies on manual processes to handle FND tasks. However, as the number of features increases, manual integration of operator knowledge becomes more challenging and costly. Therefore, automating the FND process with LLMs, such as the use of DALL-E 3 in image captioning [25], will be essential. In addition, since the evaluation of this study was conducted in a laboratory environment to obtain the datasets, future challenges include dealing with noisy data, such as those collected in the field, and improving the generalizability of the model.

VI. Conclusion

In this study, we leveraged large-scale language models (LLMs) and introduced feature name decoration (FND) techniques in regression analysis to improve the accuracy of actual router performance estimation. The core of our approach is the combination of FND and the LLM to effectively clarify the relationships between features and router metrics. Using actual router data, we demonstrated that the proposed method improves the coefficient of determination (R^2) in the estimation accuracy of actual router performance metrics, including throughput, packet loss rate, and packet delay. The improvement in R^2 reached up to 18%.

References

[1] B. Agarwal et al., "A comprehensive survey on radio resource management in 5G HetNets: Current solutions, future trends and open issues," IEEE Commun. Surv. Tut., vol. 24, no. 4, pp. 2495–2534, 2022.

[2] K. Hattori et al., "Recursive Router Metrics Prediction Using ML-based Node Modeling for Network Digital Replica," IEEE GLOBECOM, 2022.

[3] J. Yang et al., "Harnessing the power of llms in practice: A survey on chatgpt and beyond," ACM TKDD, vol. 18, no. 160, pp.1–32, 2024.

[4] L. Hu et al. "A survey of knowledge enhanced pre-trained language models," IEEE Trans. Knowl. Data Eng., vol. 36, pp. 1413-1430, 2024.

[5] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2016.

[6] W. Zhao et al., "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," IEEE SSCI, pp. 737-744, Dec. 2020.

[7] H. Li, J. Li, and A. Kaufmann, "SimBricks: end-to-end network system evaluation with modular simulation," ACM SIGCOMM, 2022.

[8] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," J. Big Data, vol. 3, no. 1, pp. 1–40, Dec. 2016.

[9] X. Chen et al., "One for all: Traffic prediction at heterogeneous 5G edge with data-efficient transfer learning," IEEE GLOBECOM, 2021.

[10] Y. Lee et al., "Surgical fine-tuning improves adaptation to distribution shifts," ICLR, 2023.

[11] J. Sun et al., "Evaluating large language models on controlled generation tasks," In Proceedings of EMNLP, 3155-3168, 2023.

[12] T. Dinh et al., "LIFT: Language-Interfaced Fine-Tuning for Non-Language Machine Learning Tasks," NeurIPS, 35, 11763–11784, 2022.

[13] S. Hegselmann et al., "TabLLM: Few-shot Classification of Tabular Data with Large Language Models," arXiv:2210.10723, 2022.

[14] E. Perez, D. Kiela, and K. Cho, "True few-shot learning with language models," NeurIPS, 34, 11054-11070, 2021.

[15] D. Chicco et al., "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation," PeerJ Comput. Sci., 7, e623, 2021.

[16] Cisco, "CSR 1000v Series," https://www.cisco.com/c/en/us/products/routers/cloud-services- router-1000v-series/index.html.

[17] Juniper, "vMX Virtual Router, " https://www.juniper.net/gb/en/products/routers/mx-series/vmx-virtual-router-datasheet.html.

[18] D. Barach et al., "High-speed software data plane via vectorized packet processing," IEEE Commun. Mag., vol. 56, no. 12, Dec. 2018.

[19] Y. Ohara et al., "Kamuee: An IP packet forwarding engine for multi-hundred-gigabit software-based networks," in Proc. IC, 2018.

[20] J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv:1810.04805, 2018.

[21] V. Sanh et al., "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," arXiv:1910.01108, 2019.

[22] "PyCaret: An open-source, low-code machine learning library in Python," https://www.pycaret.org.

[23] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," Ann. Stat., vol. 29, no. 5, pp. 1189–1232, 2001.

[24] B. Zhou et al., "A framework of small-scale large multimodal models," arXiv: 2402.14289, 2024.

[25] J. Betker et al., "Improving image generation with better captions," https://cdn.openai.com/papers/dall-e-3.pdf.