Explainable Network Anomaly Detection with GraphSAGE and SHAP

Jihoon Lee
Dept. of Intelligent Electronics and Computer Engineering
Chonnam National University
GwangJu, Korea
suda34@jnu.ac.kr

Seungmin Oh
Dept. of Intelligent Electronics and Computer Engineering
Chonnam National University
GwangJu, Korea
216655@jnu.ac.kr

Jaeho Song
Dept. of Intelligent Electronics and Computer Engineering
Chonnam National University
GwangJu, Korea
soungtain4@jnu.ac.kr

Juhyeon Noh
Dept. of Intelligent Electronics and Computer Engineering
Chonnam National University
GwangJu, Korea
uoahynz@jnu.ac.kr

Minsoo Hanh
Dept. of Computational and Data Science
Astana IT University
Astana 010000, Kazakhstan
m.hahn@astanait.edu.kz

Jinsul Kim
Dept. of Intelligent Electronics and Computer Engineering
Chonnam National University
GwangJu, Korea
jsworld@jnu.ac.kr

***Abstract*— Network anomaly detection is a critical task for maintaining network stability and security. However, existing models often focus solely on achieving high predictive performance, falling short in providing the interpretability and reliability needed for practical applications. To address this limitation, this study proposes a novel approach that combines GraphSAGE and SHAP. GraphSAGE is designed to classify various types of network anomalies effectively by leveraging network data, while SHAP extracts and quantifies the contributions of key features influencing the model's predictions. The experimental results demonstrate that the proposed model achieved high accuracy and F1-score, successfully identifying the most significant features for each anomaly class. This study highlights that the integration of GraphSAGE and SHAP enhances the interpretability and practicality of network anomaly detection. By providing clear explanations for the model's predictions, this approach offers actionable insights for network administrators, making it a valuable tool for real-world network management and security applications.**

Keywords—Network Anomaly Detection, Graph Neural Networks, XAI, GraphSAGE, SHAP

I. INTRODUCTION

The rapid evolution of network environments and the acceleration of digital transformation have underscored the critical importance of network anomaly detection. Traditional signature-based intrusion detection systems are increasingly challenged by the sophistication and diversity of modern cyber-attacks, making it difficult to effectively detect new and unknown threats.[1]

In response, artificial intelligence (AI) techniques have been employed to enhance network anomaly detection. However, many AI-based models function as "black boxes," offering limited insight into their decision-making processes. This opacity hampers trust and transparency, which are essential for the practical deployment of these systems.[2]

To address this challenge, Explainable Artificial Intelligence (XAI) has been introduced, aiming to make AI models more interpretable and their decisions more understandable. In the realm of network anomaly detection, XAI facilitates the analysis of detected anomalies and the identification of key contributing features, thereby enhancing the interpretability of AI-driven security measures.[3]

This study proposes a novel approach that integrates GraphSAGE—a graph neural network framework capable of learning node embeddings in large graphs—with SHAP (SHapley Additive exPlanations), a method for interpreting complex models. By combining these tools, the research aims to develop a network anomaly detection system that not only identifies anomalies but also provides clear explanations regarding the features that most significantly influence each detection.

II. RELATED WORKS

A. Graph Neural Networks

Graph Neural Networks (GNNs) are a groundbreaking technology that extends deep learning to the non-Euclidean domain of graph data structures. GNNs are designed to learn relationships and structural information in graph data by iteratively aggregating and transforming information from neighboring nodes, enabling the learning of expressive node embeddings. This concept was first introduced by Gori et al. in 2005. Their study proposed the first GNN structure to process graph data, showcasing its potential for solving graph-based problems.[4]

In the subsequent application stages of GNNs, the Graph Convolutional Network (GCN) introduced by Kipf and Welling in 2017 marked a significant advancement in GNN technology. GCN employs a normalized Laplacian approach to perform semi-supervised learning, enabling the efficient and powerful learning of node embeddings. This model demonstrated outstanding performance across various domains, including social network analysis and molecular modeling, establishing itself as a cornerstone in GNN research.[5]

GraphSAGE, based on the inductive representation learning framework of sampling and aggregation, was proposed by Hamilton et al. in 2017. This model was designed to learn relationships and structures in large-scale graphs. Unlike traditional GNN models that require the entire graph to be stored and trained, GraphSAGE introduces a neighborhood sampling and aggregation approach to generate embeddings. This enables the model to handle dynamic changes in graphs and predict new nodes. GraphSAGE has demonstrated its utility in various fields, including social networks, biological networks, and recommendation systems,

positioning itself as a robust model for large-scale graph processing.[6]

GNN research has further expanded to Graph Attention Networks (GAT). Velickovic et al. (2018) introduced an attention mechanism in GAT, effectively integrating structural and attribute information of graph data. GAT enables each node to aggregate information from its neighbors based on their relative importance, making it particularly useful in heterogeneous or dynamic graph environments. This model has significantly improved performance across various graph-based tasks, further enhancing the expressiveness of GNNs.[7]

In addition, various AI models derived from GNNs, such as M-GAT, GDN, and MST-GAT, have emerged. This study aims to employ GraphSAGE for anomaly detection, leveraging its capability to learn node relationships and adapt to dynamic network data effectively.

B. Network Anomaly Detection

Network anomaly detection is an essential process for identifying factors that degrade network performance and addressing them promptly. Recently, AI technologies have emerged as powerful tools for detecting anomalies in traffic, latency, jitter, and packet loss. This section introduces AI research related to traffic anomalies, network performance metrics (latency, jitter, packet loss), and network attack detection.

Network performance indicators such as latency and jitter significantly impact user experience. Rusek et al. (2020) RouteNet is a study that employs Graph Neural Networks (GNNs) to model network latency and jitter. By leveraging a graph-based approach, this study accurately predicts delays and jitter within network paths, enabling network operators to proactively detect and address potential performance degradations.[8]

Fotiadou et al. (2021) introduced a novel framework for detecting anomalies in network traffic using deep learning techniques. By leveraging pfSense logs and Suricata intrusion detection data, they proposed semi-supervised approaches based on Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN). Their system classifies network events into multi-class categories, achieving high accuracy (97.27% for LSTM and 97.24% for CNN) while addressing imbalanced dataset challenges through class weighting and dropout layers. This approach highlights the potential of combining real-world network logs with advanced DL architectures to enhance intrusion detection and network anomaly classification.[9]

Ma et al. (2021) provide an in-depth analysis of the state-of-the-art methods for detecting anomalies in graphs. The survey categorizes techniques based on supervised, unsupervised, and semi-supervised learning approaches, highlighting their strengths and application scenarios. Notably, the paper emphasizes the growing role of Graph Neural Networks (GNNs) in anomaly detection tasks, where their ability to learn rich node representations and relationships makes them well-suited for identifying anomalous patterns in complex network structures.[10]

C. Explainable AI(XAI)

Explainable AI (XAI) has emerged as a crucial field aimed at addressing the "black-box" nature of deep learning models. XAI techniques provide insights into model decisions, ensuring transparency, interpretability, and trust in AI systems. In network anomaly detection, XAI allows operators to understand why a specific anomaly was flagged and what features contributed most to the model's decision.

Ribeiro et al. (2016) proposed LIME (Local Interpretable Model-agnostic Explanations) as a technique for explaining the predictions of any machine learning model. LIME generates locally faithful explanations by approximating the model's decision boundary around specific data points. This approach has been extended to network anomaly detection tasks, particularly in understanding why certain traffic patterns are flagged as suspicious. Through LIME, operators can better interpret complex models and gain actionable insights into network anomalies.[11]

Lundberg et al. (2017) introduced SHAP (Shapley Additive Explanations), a unified framework for interpreting predictions made by machine learning models. SHAP assigns importance scores to individual input features, quantifying their contributions to the model's output. This method has been applied in various domains, including network anomaly detection, where it helps identify the key factors influencing anomalous patterns. By providing interpretable explanations, SHAP enables network administrators to pinpoint root causes effectively and act on them.[12]

Verma et al. (2020) provided an extensive review of counterfactual explanations as a method for interpreting machine learning models. Counterfactual explanations aim to provide insights by answering "what-if" questions, describing how minimal changes to input features could alter the model's predictions. This technique emphasizes user-centric interpretability by offering actionable feedback and has been widely explored in decision-critical domains such as finance and healthcare. They highlight the advantages of counterfactual explanations in enhancing transparency and trustworthiness of AI systems, particularly in scenarios where understanding model predictions is essential.[13]

III. METHODOLOGY

This chapter describes the methodologies used in the study of explainable network anomaly detection. Before delving into the details of the methodology, we provide a brief explanation of GraphSAGE and SHAP, which form the foundation of our approach. Since these methods have been introduced in the Related Works section, this section will focus on their operational principles and how they are applied in this study.

A. Background

GraphSAGE, proposed by Hamilton et al. (2017), is a graph neural network framework designed for inductive learning on graph-structured data. It generates node embeddings by sampling and aggregating features from neighboring nodes, typically utilizing edge information to learn rich structural representations. However, inspired by MST-GAT (2023), this study adapts GraphSAGE to focus solely on node features without explicitly utilizing edge connections. This adjustment allows the model to effectively detect anomalies in high-dimensional network data, even in scenarios where the graph structure is ambiguous or undefined.[6][14]

By emphasizing node attributes, GraphSAGE in this study is tailored to learn meaningful patterns from node-level information, enabling it to generalize to unseen nodes and

handle dynamic or evolving network environments effectively. This approach aligns well with the requirements of network anomaly detection tasks, demonstrating the model's adaptability in complex, real-world applications.

SHAP (SHapley Additive exPlanations), introduced by Lundberg et al. (2017), is a unified framework for interpreting predictions made by machine learning models. Built on the principles of cooperative game theory, SHAP assigns importance scores to individual input features, quantifying their contributions to the model's output. These scores are computed by considering all possible feature subsets, ensuring a comprehensive and fair representation of each feature's influence.[12]

In this study, SHAP is utilized to explain the predictions of a GraphSAGE-based anomaly detection model. By analyzing the SHAP values of node features, the model can identify the most influential attributes contributing to the detection of network anomalies. This insight not only enhances the interpretability of the anomaly detection process but also provides valuable information for understanding the root causes of anomalies. SHAP's ability to provide consistent, additive explanations makes it particularly suitable for applications in network anomaly detection, where understanding feature-level contributions is crucial for effective troubleshooting and response.

B. Dataset

1) Dataset Composition

The dataset used in this study is designed for classifying various normal and anomalous network behaviors based on network traffic analysis. It contains a total of 35 features and 8 classes, representing normal traffic and a variety of network anomalies. The dataset was constructed using a variety of protocols and traffic patterns collected in network environments.

Each column in the dataset represents a specific measurement of network traffic, which includes features from the IP layer, TCP layer, UDP layer, ICMP layer, and other metrics such as transmitted/received bytes, unicast packets, and discarded packets.

The class labels in the dataset indicate whether the traffic is normal or anomalous, with the classes defined as follows:

- normal: Normal network traffic.
- tcp-syn: TCP SYN flooding attack, which overloads the server by excessively creating TCP connections.
- slowloris: An attack where the client establishes a connection and deliberately delays requests, consuming server resources.
- udp-flood: A flooding attack leveraging the UDP protocol to consume network bandwidth.
- icmp-echo: An attack that overwhelms the network by sending excessive ICMP echo requests (ping).
- httpFlood: An attack that generates a large number of HTTP requests to overload the web server.
- slowpost: An attack that sends HTTP request bodies at an extremely slow rate, exhausting server resources.
- bruteForce: An attack attempting to bypass authentication by making random guesses at login credentials.

2) Dataset Preprocessing

The dataset exhibited significant class imbalance, with the tcp-syn class containing 960 samples, the bruteForce class containing 200 samples, and other classes ranging between 400 and 780 samples. Such class imbalance can lead to overfitting on the majority class and poor performance on the minority class in machine learning models.

To address this issue, the dataset was balanced by first generating 200 additional samples for the bruteForce class using a WGAN-based synthetic data generation method. After augmentation, all classes were under-sampled to contain 400 samples each, ensuring a balanced dataset for model training and evaluation.

3) Data Splitting

For model training, the dataset was split into training and test sets with an 8:2 ratio. The splitting process employed a random split rather than a stratified split. This choice was made because the model used in this study (GraphSAGE) is designed to learn patterns at the node level, and the dataset had already been balanced across classes, eliminating the need for stratified splitting.

This balanced and randomly split dataset ensures that the model can learn effectively while avoiding bias toward specific classes, thereby improving its generalizability.

C. Model Description

1) GraphSAGE

In this study, we adopted the core design of GraphSAGE but focused on node-centric learning, considering the characteristics of network data. While GraphSAGE typically leverages edge information to learn relationships between nodes, this study adapted the model to learn patterns among node features without utilizing edge connections. This adjustment enables the effective handling of high-dimensional node features and allows for anomaly detection even in environments where the graph structure is not explicitly defined.

The model was constructed with a softmax output layer for class predictions, and CrossEntropyLoss was employed to address the multi-class classification problem. The Adam optimizer was utilized for model optimization, and loss and accuracy were recorded at each epoch to evaluate performance.

2) SHAP

In this study, SHAP (Shapley Additive Explanations) was employed to interpret the model's predictions and quantify the contributions of individual features. Among various SHAP implementations, Kernel SHAP was selected and applied. Kernel SHAP operates as a model-agnostic approach, calculating SHAP values solely based on the input-output relationship, making it independent of specific model architectures.

The SHAP analysis process was structured as follows. First, background data were extracted from the training dataset and paired with the model's prediction function to calculate the impact of each input feature using KernelExplainer. Subsequently, feature importance was analyzed for each class to identify the features contributing most significantly to the predictions. Finally, the SHAP analysis results were

visualized to confirm the contribution of key features for each class. This process allowed us to understand which features predominantly influenced the predictions of the network anomaly detection model.

IV. Experimental Results

This chapter presents the results of the experiments conducted to evaluate the proposed methodology for explainable network anomaly detection. The experiments were designed to assess two primary objectives:

- Model Performance: Evaluating the classification accuracy, precision, recall, and F1-score of the GraphSAGE-based anomaly detection model.
- Feature Interpretability: Demonstrating the interpretability of the model's predictions using SHAP to identify the most influential features contributing to network anomaly detection.

Section **a** describes the performance evaluation metrics and baseline results. Section **b** explores the interpretability results obtained through SHAP analysis.

a) Model Performance Evaluation

The performance of the GraphSAGE-based network anomaly detection model developed in this study was evaluated using metrics such as accuracy, precision, recall, and F1-score. Additionally, the changes in loss and accuracy during the training and testing phases were visualized through graphs to analyze the model's learning stability and generalization capabilities.

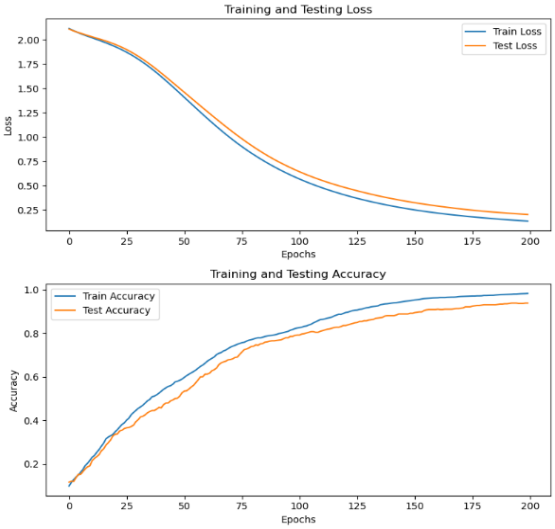

Fig. 1. Training and test loss and accuracy curves

Fig. 1. illustrates the training and testing loss and accuracy curves over 200 epochs for the GraphSAGE-based network anomaly detection model. The top graph shows the loss curves, where both the training and testing losses steadily decrease, indicating effective convergence of the model. The close alignment between training and testing losses suggests minimal overfitting and good generalization capabilities.

The bottom graph depicts the accuracy curves for training and testing datasets. Both curves exhibit consistent improvement, with the training accuracy slightly exceeding the testing accuracy, which is typical in machine learning. The testing accuracy stabilizes near the end of the training process, demonstrating the model's strong performance and learning stability. This analysis supports the model's reliability in network anomaly detection tasks.

TABLE I. Overall Metrics

	Metrics			
	Precision	***Recall***	***F1-Score***	***Accuracy***
Value	0.9439	0.9391	0.94	0.9391

Table I presents the overall performance metrics of the model, summarizing the precision, recall, F1-score, and accuracy across all classes. These metrics provide a comprehensive view of the model's effectiveness in distinguishing normal and anomalous network traffic.

TABLE II. Class-wise Metrics

	Metrics		
	Precision	***Recall***	***F1-Score***
Normal	0.79	0.95	0.86
tcp-syn	0.97	0.93	0.95
Slowloris	0.91	0.89	0.90
udp-flood	0.98	0.99	0.98
icmp-echo	0.99	0.96	0.97
httpFlood	0.94	0.99	0.96
Slowpost	0.99	0.89	0.93
BruteForce	0.99	0.93	0.95

Table II further breaks down these metrics on a per-class basis, offering insights into the model's performance for each specific traffic type, including both normal and various attack patterns. This class-wise analysis helps to identify the strengths and weaknesses of the model in detecting individual anomaly types, which is crucial for understanding its practical applicability in real-world network environments.

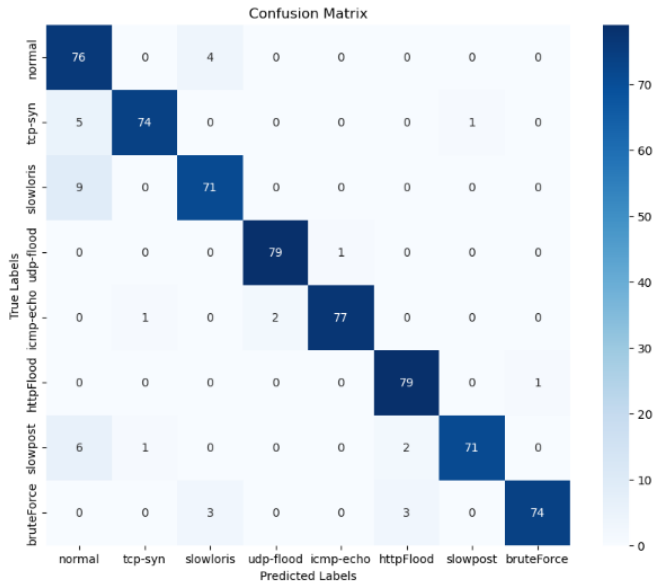

Fig. 2. Confusion Matrix of the GraphSAGE-based model

Fig. 2. presents the confusion matrix generated from the predictions of the GraphSAGE-based network anomaly

detection model. Each row represents the true class, and each column represents the predicted class. The diagonal elements indicate the number of correctly classified samples for each class, while off-diagonal elements represent misclassifications. The matrix highlights the model's strong ability to distinguish between different types of network traffic, with high accuracy across most classes. Despite a few misclassifications observed in classes like "normal" and "slowloris," the overall performance demonstrates the model's effectiveness in detecting both normal and anomalous traffic.

b) SHAP analysis

In this section, the SHAP (Shapley Additive Explanations) analysis is presented to interpret the predictions of the anomaly detection model. The primary goal of SHAP analysis is to identify the key features that contribute most significantly to the model's predictions. This interpretability is particularly important in network anomaly detection, where understanding the driving factors behind an anomaly can aid in diagnosing and mitigating network issues.

The SHAP analysis results are visualized using summary plots, which highlight the relative importance of each feature across all predictions. For example, features related to packet transmission rates or specific TCP characteristics might emerge as critical factors for predicting anomalies in classes such as Bruteforce or TCP-SYN attacks. These visualizations provide an intuitive way to understand which features have the greatest impact on the model's decision-making process.

The SHAP value graph illustrates the impact of each feature on the model's predictions. The X-axis represents the SHAP values (feature contributions), where values greater than 0 indicate that the feature increases the likelihood of a specific class prediction. The Y-axis lists the feature names, and the color represents the magnitude of the feature values (red indicates high values, while blue indicates low values). This visualization enables an intuitive understanding of the relationship between feature values and model predictions.

The SHAP values for all 8 classes were analyzed, revealing the contribution of specific features to the model's predictions for each class. However, including SHAP graphs for all classes in the paper would result in visual complexity, potentially hindering the reader's ability to grasp the core insights. Therefore, this study focuses on the SHAP analysis results for the BruteForce class as a representative example to provide a detailed explanation of the model's interpretability and feature importance analysis.

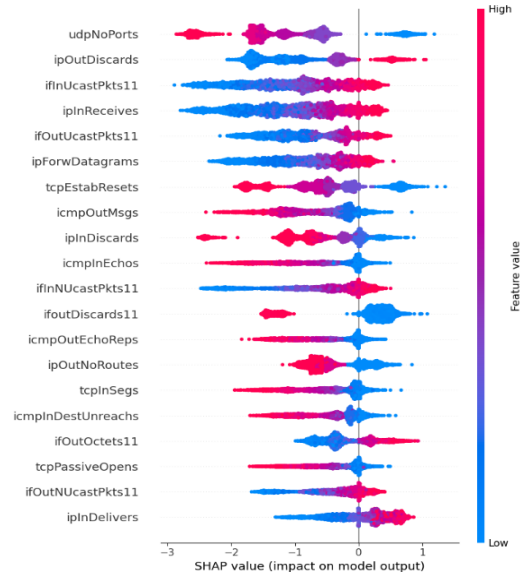

Fig. 3. BruteForce Shap value graph

Fig. 3 illustrates the SHAP value visualization for the BruteForce class. The top-ranked features, udpNoPorts and ipOutDiscards, play a significant role in predicting BruteForce attacks.

Firstly, udpNoPorts exhibits a negative SHAP value when it has high feature values, indicating a reduced likelihood of predicting BruteForce. This aligns with domain knowledge, as high udpNoPorts values are more likely to reflect patterns associated with UDP-based attacks, such as port scanning or flood attacks. In contrast, BruteForce attacks are characterized by TCP-based authentication attempts, meaning that higher udpNoPorts values correlate less with BruteForce predictions.

Secondly, ipOutDiscards shows a positive SHAP value when it has high feature values, increasing the likelihood of predicting BruteForce. This can be attributed to the repetitive authentication attempts in BruteForce attacks, which may lead to network resource exhaustion and subsequent packet discards. Such network congestion is a common characteristic of BruteForce attacks, making this feature contribution consistent with domain knowledge.

The insights derived from this analysis are not only consistent with known characteristics of BruteForce attacks but also provide actionable intelligence for network administrators. Compared to other anomaly types, such as UDP-based or HTTP-based floods, BruteForce attacks exhibit a distinct reliance on TCP-related features, such as repetitive packet discards and lower UDP-based traffic patterns. This distinction highlights the model's ability to isolate BruteForce-specific root causes, providing actionable insights into anomaly mitigation strategies. By understanding which features significantly impact the model's predictions, administrators can focus on monitoring or mitigating factors like TCP-based authentication patterns and resource exhaustion due to repetitive attempts. Furthermore, linking SHAP values to domain knowledge enhances the model's trustworthiness for root cause analysis, making it a valuable tool for identifying and addressing anomalies in real-world network environments.

V. CONCLUSION

This study proposed a novel approach combining GraphSAGE and SHAP (Shapley Additive Explanations) for network anomaly detection and root cause analysis. GraphSAGE effectively learned anomalous patterns from high-dimensional features of network data through node-centric learning, while SHAP was employed to interpret model predictions and quantify the contributions of key features. This approach not only improved predictive accuracy but also provided interpretability for the results, demonstrating its potential as a practical tool for network management and security.

The experimental results showed that the proposed model achieved high accuracy and F1-scores across various network anomaly scenarios. Moreover, SHAP enabled the identification of key features for each class, proving the capability of performing explainable network anomaly detection. This approach allows network administrators to understand the basis of model predictions and devise appropriate response strategies.

However, this study has several limitations. First, the absence of edge information in GraphSAGE may have limited the model's ability to capture spatial relationships inherent in graph structures. Second, the SHAP analysis revealed feature importance that, in some cases, did not fully align with domain knowledge. This discrepancy may not stem from experimental errors but rather from dataset bias or a lack of dataset completeness. Future studies should aim to enhance dataset diversity and quality to derive more reliable results.

Future research directions include several potential expansions. First, incorporating edge information into GraphSAGE to capture spatial relationships in graph structures could be explored to better leverage the structural properties of network data. Additionally, experiments with other models derived from GNNs, such as GDN, MST-GAT, or GAT, could further advance network anomaly detection by reflecting spatiotemporal or graph-based characteristics. Second, integrating various XAI techniques, such as LIME (Local Interpretable Model-agnostic Explanations) or Counterfactual Explanations, could enable more sophisticated and multifaceted root cause analyses. Finally, validating the proposed model's performance in real-world network environments and assessing its generalizability through follow-up research would be crucial to bridging theoretical findings with practical applications. These efforts are expected to enhance the interpretability and efficacy of network anomaly detection.

In summary, this study contributes to advancing explainability in network anomaly detection and demonstrates that the integration of GraphSAGE and SHAP can serve as an effective tool for network management and security.

ACKNOWLEDGMENT

This paper was supported by the Korea government (Ministry of Science and ICT) through the Institute of Information & Communications Technology Planning & Evaluation (IITP) under the Innovative Human Resource Development for Local Intellectualization program (IITP-2024-00156287, 50%)

and This paper was supported by the Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2024-00345030, Development of digital twin base network failure prevention and operation management automation technology, 50%)

REFERENCES

[1] S. Wang, J. F. Balarezo, S. Kandeepan, A. Al-Hourani, K. Gomez Chavez, and B. Rubinstein, "Machine Learning in Network Anomaly Detection: A Survey," IEEE Access, vol. 9, pp. 3126834, 2021.

[2] Z. Li, Y. Zhu, and M. van Leeuwen, "A Survey on Explainable Anomaly Detection," ACM Transactions on Knowledge Discovery from Data, vol. 18, Issue. 1, pp. 1–54, 2023.

[3] K. Roshan and A. Zafar, "Using Kernel SHAP XAI Method to Optimize the Network Anomaly Detection Model," 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom). IEEE, 2022.

[4] 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, Jan. 2008.

[5] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in International Conference on Learning Representations (ICLR), 2017.

[6] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in Advances in Neural Information Processing Systems (NeurIPS), 2017.

[7] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph Attention Networks," in International Conference on Learning Representations (ICLR), 2018.

[8] K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, and A. Cabellos-Aparicio, "RouteNet: Leveraging graph neural networks for network modeling and optimization in SDN," IEEE Journal on Selected Areas in Communications, vol. 38, no. 10, pp. 2260-2270, 2020.

[9] K. Fotiadou, T. H. Velivassaki, A. Voulkidis, D. Skias, S. Tsekeridou, and T. Zahariadis, "Network traffic anomaly detection via deep learning," Information, vol. 12, no. 5, pp. 215, 2021.

[10] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. Sheng, and L. Akoglu, "A comprehensive survey on graph anomaly detection with deep learning," IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 12, pp. 12012-12038, 2021.

[11] M. T. Ribeiro, S. Singh, and C. Guestrin, "" Why should i trust you?" Explaining the predictions of any classifier," In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1135-1144, 2016.

[12] S. Lundberg and S.I. Lee, "A unified approach to interpreting model predictions," In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17), pp. 4768-4777, 2017

[13] S. Verma, V. Boonsanong, M. Hoang, K. Hines, J. Dickerson and C. Shah, "Counterfactual Explanations and Algorithmic Resources for Machine Learning: A Review," ACM Computing Surveys, vol. 56(12), no. 312, pp. 1-42, 2024

[14] C. Ding, S. sun and J. Zhao, "MST-GAT: A multimodal spatial–temporal graph attention network for time series anomaly detection," Information Fusion, vol. 89(c), pp. 527-536, 2023