A Robust Malicious Traffic Detection Framework with Low-quality Labeled Data

Lingfeng Yao*, Weina Niu*, Qingjun Yuan‡, Beibei Li†, Yanfeng Zhang§, Xiaosong Zhang*
*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China
†School of Cyber Science and Engineering, Sichuan University, Chengdu, China
‡Henan Key Laboratory of Network Cryptography Technology, Zhengzhou, China
§Intelligent Policing Key Laboratory of Sichuan Province, Sichuan Police College, Luzhou, China

***Abstract*—Deep learning (DL) techniques have been widely applied in detecting malicious activities from network traffic. However, it is challenging to collect a traffic dataset with sufficient correct labels. The generalization ability of DL-based malicious traffic detection systems decreases when training with mislabeled data. Therefore, several methods have been proposed to detect malicious traffic from low-quality labeled training data. These methods divide noisy and clean samples based on the divergence of their prediction loss. However, this simple criterion is not effective on traffic data due to the obfuscation and redundancy nature of malicious traffic. In this paper, we propose a novel two-stage framework for malicious traffic detection from low-quality training data, which mainly consists of noisy sample filtering and label refinement. Firstly, with the help of the small loss criterion, we filter out most of the noisy samples from training data while ensuring that the filtered dataset covers sufficient clean samples. Next, we introduce a double-constrained similarity rule to provide a comprehensive measure of the similarity between samples and construct a topological graph. Lastly, we exploit the topological relations extracted from this graph to refine the labels based on the neighbor consistency criterion. We validate the effectiveness of our framework with a real-world malicious traffic dataset, achieving an accuracy of 90% even with 80% symmetric noise labels. Additionally, results from the publicly available BoT-IoT dataset demonstrate the adaptability of our framework to Internet of Things (IoT) environments.**

***Index Terms*—Malicious traffic detection, Label noise, Low-quality data, Deep learning**

I. Introduction

Intrusion detection systems (IDS) play an important role in detecting malicious behaviors in network environments. Traditional rule-based intrusion detection methods are outdated as they rely on hand-crafted rules, leading to a high false alarm rate in large-scale network scenarios [1]. Currently, network-based intrusion detection methods have been widely used in various network scenarios to detect malicious traffic. These methods can automatically extract deep features of traffic data and learn potential divergence between different classes of traffic. However, the performance of existing approaches tends to highly depend on clean training data.

In general, when constructing malicious traffic datasets, experts simulate both normal and attack activities in a sandbox environment and label the traffic based on different IP address mappings associated with attackers and regular users. However, a complete attack activity may include some normal activities [2], and the reliance on the single IP rule can inevitably introduce noisy labels, i.e., mislabeled traffic data, into the training set. Moreover, the obfuscation and redundancy nature of encrypted traffic poses a significant challenge for human annotators to accurately label the collected data in real-world scenarios. Network-based malicious traffic detection systems may fit mislabeled samples during the training stage, resulting in a great decline in the detection performance. Therefore, obtaining robust detection models from noisy data (low-quality data) is currently considered an urgent and necessary problem.

Many scholars have carried out research to solve this labeling noise problem [3]–[6]. The main idea of these methods is to separate noisy samples from clean ones based on the divergence of their prediction loss and divide the original dataset into the labeled set (clean samples) and the unlabeled set (noisy samples), respectively. This can transform the learning with noisy labels (LNL) task into a semi-supervised learning (SSL) task. However, these approaches have limitations in both sample selection and semi-supervised learning. In the sample selection stage, existing methods [3]–[5] rely heavily on predicted loss variances of samples while ignoring the topological information from the feature space. This may inevitably introduce confirmation errors during the training process. Bahri *et al.* [6] proposed to filter mislabeled samples based on the assumption that samples of the same class are closer in the feature space. However, due to the obfuscation and redundancy nature of encrypted malicious traffic, it is difficult to precisely divide clean and noisy data. On the other hand, the performance of semi-supervised learning [3] [4] is dependent on the gain from prediction consistency, where the model is trained to output consistent predictions for data-augmented samples. Unlike image inputs, network traffic data lacks reasonable data augmentation techniques such as rotation and cropping. Consequently, these data augmentation-based SSL methods are not applicable to tackle label noise in traffic detection systems.

To obtain a robust malicious traffic detection model with low-quality training data, this paper proposes a novel two-stage learning framework. Firstly, in the preliminary filtering stage, we employ the small loss criterion to select noisy samples from each class of traffic data. This ensures that the cleaned dataset

This work was supported by the Opening Project of Intelligent Policing Key Laboratory of Sichuan Province (No. ZNJW2023KFQN003). (Corresponding author: Weina Niu.)

covers a sufficient number of clean samples while significantly reducing the label noise rate. Secondly, in the label refinement stage, we propose a double-constrained similarity rule, i.e., feature distance similarity and probabilistic trend similarity, to construct the similarity topology graph of samples. The former assumes that samples of the same class have similar locations in the high-dimensional feature space [6] and the latter assumes that these samples also have similar optimization dynamics [7]. Then, the refinement of potential noisy labels is guided by the neighborhood consistency assumption that adjacent samples in the topology map share the same label. Our main contributions can be summarized as follows:

- We filter noisy samples from each class based on the small loss criterion. This ensures a comprehensive distribution of clean samples while effectively reducing the label noise rate of the training set, thus mitigating the impact of mislabeled data on label refinement.
- We propose a novel double-constrained similarity rule to construct a similarity topology graph and update labels using a topology graph-based label refinement mechanism. It effectively avoids the accumulation of noisy labels and improves the detection performance of the proposed model.
- We verify the effectiveness of the proposed framework on our encrypted malicious traffic dataset constructed in the real world and a popular public traffic dataset. Our proposed framework outperforms several state-of-the-art methods under different label noise scenarios.

II. Related Work

A. Noisy Labels in Traffic

Building malicious traffic detection systems requires a large amount of accurately labeled data. Currently, most malicious traffic datasets are constructed by simulating known attacks and capturing network traffic in a controlled environment [8]. However, during these simulations, certain legitimate traffic such as ARP messages, can be generated in the early and later stages of the attack [2]. This legitimate traffic may erroneously be identified as malicious due to the rigid machine labeling mechanism, thus inevitably introducing label noise. On the other hand, unlike tasks such as image labeling where distinguishing between a cat and a dog is quite straightforward, labeling traffic data is more challenging. Due to its redundancy and obfuscation nature, different experts may provide varying labels based on their experience. For instance, when labeling the 496 samples in a dataset, only 12% of them received consistent labels from the majority of experts [9]. Consequently, manual labeling can result in severe label noise impact.

B. LNL-Based IDS

LNL methods aim to learn robust models from low-quality labeled datasets. Zhang *et al.* [10] introduced an adaptive generalized loss that automatically adjusts the model's sensitivity to noise labels. However, this method can not work well in highly noisy scenarios. Zhao *et al.* [11] proposed a label proofreading framework based on multi-model voting prediction to realize malicious traffic detection, but the integrated model still relies on a limited amount of clean labeled data, which may not align with real-world assumptions. Li *et al.* [3] and Karim *et al.* [4] transformed the LNL problem into an SSL problem using the small loss criterion. However, these methods might not be suitable for network traffic data due to the lack of reasonable data augmentation techniques. Bahri *et al.* [6] employed label propagation on samples based on their feature distribution, but its performance suffered a significant decline as the label noise rate increased.

III. The Proposed Framework

In this section, we first briefly describe our data preprocessing strategy for network traffic datasets. Then our proposed framework will be described in detail. The overall workflow of our framework is shown in Fig. 1.

A. Data Preprocessing

When constructing network traffic datasets, it is necessary to include some important categorical attributes, such as destination port numbers and network protocols, while preserving numerical features like packet length and flow duration. The values of these attributes are discrete and hard to be leveraged by deep learning models directly. The conventional approach [12] involves converting categorical features into one-hot encoding features. However, this approach may not be suitable for network traffic datasets [1]. Take the destination port as an example, a network traffic dataset can include up to 65,536 distinct ports. Applying one-hot encoding in ports would introduce 65,536 new sparse features to the dataset, leading to the curse of dimensionality.

Following Hou *et al.* [13], we employ the idea of mapping the categorical features to their respective frequencies. We count the occurrences of each feature value in the training set and normalize them to obtain the frequency of each value, thus creating a mapping from feature values to their respective frequencies. For the test data, we transform feature values into frequency values using this mapping. Specifically, for feature values that do not appear in the mapping, we set their frequency values to 0. The frequency is calculated as shown in Eq. 1.

$$Freq(v) = \begin{cases} n_v/N & \text{if } v \in \mathbb{V} \\ 0 & \text{if } v \notin \mathbb{V} \end{cases} \tag{1}$$

where $\mathbb{V}$ denotes the set of values for the categorical feature in the training data. v is a specific value of this categorical feature. n_v signifies the count of occurrences of this specific value, and N refers to the total number of training data. For instance, the UDP protocol is commonly used for data transmission. If out of 1000 training samples, the UDP protocol appears 340 times, the frequency of protocol UDP is calculated as 0.34 using Eq. 1. Subsequently, we replace the protocol UDP with its frequency of 0.34 in both the training and testing datasets.

We normalize the numerical features in the dataset and drop rows containing outliers or missing values. The dataset is then divided into a training set and a testing set in a 7:3 ratio. As

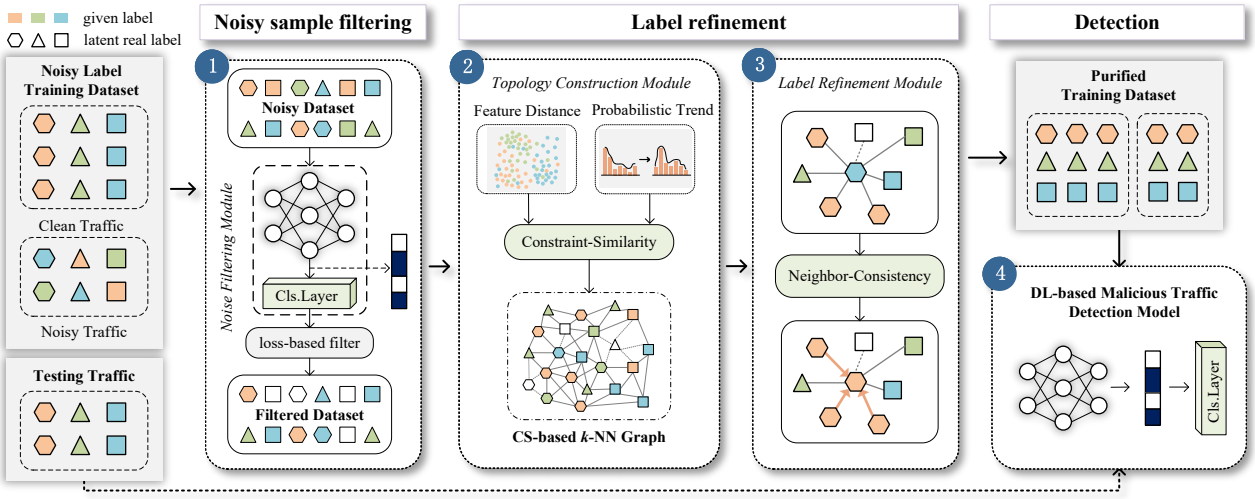

Fig. 1: Overview of the proposed framework. In the first stage, we perform preliminary noisy sample filtering to filter out most of the noisy samples. In the second stage, we construct a topological graph based on the double-constrained similarity rule and utilize it to refine mislabeled samples. Finally, we obtain the purified dataset, which is used for training and achieving malicious traffic detection.

illustrated in Fig. 1, the testing set plays no role in the training process and remains free from label noise. In the training set, we introduce two realistic noise labels by flipping the labels according to the settings detailed in Section IV-B.

B. Noisy Sample Filtering

Let $\mathbb{D} = \{(x_i, y_i)\}^N$ represents the training dataset for network traffic, which includes noisy labels. Here, x_i denotes the i-th traffic sample, y_i is the given label corresponding to x_i in the training set, and N signifies the total number of training samples. For each x_i, there exists one unique latent real label $\hat{y}_i \in \{1, 2, \cdots, C\}$, where C denotes the total number of traffic categories. If $y_i \neq \hat{y}_i$, then x_i is considered the noisy traffic, otherwise, it is regarded as clean traffic.

As depicted in step one, our noise filtering module comprises two essential components: a feature extractor, denoted as $f(.;\theta)$ with parameter θ, and a classification layer, represented as $g(.;\phi)$ with parameter ϕ. The feature extractor f is a fully connected neural network that removes the classification layer, which is designed to transform the input network traffic features into meaningful representations with latent information. Meanwhile, the classification layer g outputs model predictions for each sample.

The previous research [14] has demonstrated that deep neural networks tend to prioritize the learning of clean samples in the early stages of model training. Consequently, clean samples exhibit lower prediction losses compared to noisy ones. By evaluating the divergence in prediction losses between noisy and clean samples, we can identify and filter out easy instances of noisy traffic. First, we initiate model pre-training, denoted as $g(f(.;\theta);\phi)$ for several epochs using the training set $\mathbb{D}$ which contains a certain ratio of noisy labels. During this process, we update the parameters θ and ϕ. Then, we employ the cross-entropy loss to quantify the divergence

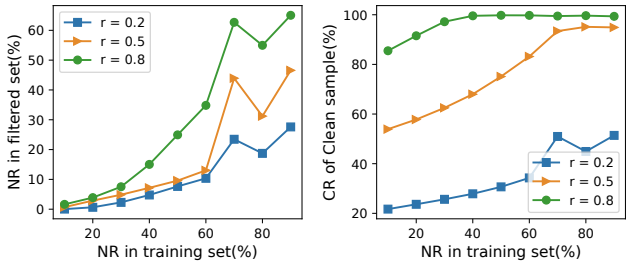

Fig. 2: Noise rate (NR) and clean sample coverage rate (CR) with different portion r.

between the model's predicted probability $g(f(x_i;\theta);\phi)$ and the given label y_i for each sample. $\mathbb{L}_c$ denotes the set of prediction loss for each class c, where $c \in \{1, 2, \cdots, C\}$:

$$\mathbb{L}_c = \{-y_i \log g(f(x_i;\theta);\phi) \mid y_i = c\} \tag{2}$$

During the noise filtering process, we maintain class balance by retaining a fraction r of the sample labels with low prediction losses from each class. The remaining $1-r$ portion of samples are considered as noise, and their labels are discarded. Consequently, we aggregate the divided clean and noisy samples of each class to create $\mathbb{D}_{clean}$ of Nr and $\mathbb{D}_{noisy}$ of $N(1-r)$.

In Fig. 2 left, we depict how different values of hyper-parameter r relate to the noise rate in the filtered dataset compared to the original training set. Fig. 2 right illustrates the relationship between the coverage of clean samples in the filtered dataset and the noise rate in the original training set. We set r to 0.5, effectively reducing noise while preserving essential information from clean samples.

C. Label Refinement

Following step one, we obtain the filtered dataset and the feature extractor f with updated parameters θ through pre-training. Inspired by previous studies [15] [7], we note that the feature space is more robust to noisy labels than the output space, and samples belonging to the same class exhibit similar optimization dynamics. Building upon these insights, we introduce a double-constrained rule that leverages feature distance similarity and prediction probability trend similarity to construct a topological relationship graph among samples. Subsequently, we employ learned topological similarity to perform label refinement.

1) Feature distance similarity: Given that contextual information in the feature space is more robust to noisy labels, we explore high-dimensional information in the feature space. We employ the cosine distance to quantify the similarity among two high-dimensional features $\delta_i = f(x_i;\theta)$ and $\delta_j = f(x_j;\theta)$.

$$S_{feat}(i,j) = 0.5(1 + \frac{\delta_i \cdot \delta_j}{\parallel \delta_i \parallel_2 \parallel \delta_j \parallel_2}) \tag{3}$$

where $S_{feat}(i,j)$ ranges from 0 to 1. The closer $S_{feat}(i,j)$ approaches 1, the more similar δ_i and δ_j are.

2) Probabilistic trend similarity: Considering the consistent optimization dynamics among samples from the same class, we presume that samples sharing the same class have similar change trends in their prediction probability. We define the prediction probabilistic change trend as:

$$\Delta p_i = g_{end}(f(x_i;\theta);\phi) - g_{start}(f(x_i;\theta);\phi) \tag{4}$$

where g_{end} represents the predicted probabilities at the end of the pre-training stage and g_{begin} signifies the beginning. We employ Jensen-Shannon divergence (JSD), denoted as $d(i,j)$, as the criterion for quantifying the similarity of change trends between two predicted probabilities.

$$\begin{aligned} d(i,j) &= \text{JSD}(\Delta p_i, \Delta p_j) \\ &= \frac{1}{2}\text{KL}(\Delta p_i || \frac{\Delta p_i + \Delta p_j}{2}) + \frac{1}{2}\text{KL}(\Delta p_j || \frac{\Delta p_i + \Delta p_j}{2}) \end{aligned} \tag{5}$$

where KL(.) is the Kullback-Leibler divergence function. The closer $d(i,j)$ approaches 0, the more similar Δp_i and Δp_j are.

3) Topology construction: Based on these two similarity metrics, we propose a novel double-constrained similarity rule to construct the topological graph.

$$Sim(x_i, x_j) = (1-\sigma)S_{feat}(i,j) + \sigma(1 - d(i,j)) \tag{6}$$

where σ is the fusion coefficient to adjust the portion of these two similarities.

For each sample within the dataset, we select the k samples with the highest similarity as their neighbors and connect them accordingly to construct the topological graph. Subsequently, we proceed to refine the labels for the entire graph. The underlying concept is that the true latent label of each sample should align with its nearest neighbors. Therefore, we identify these closest neighbors for each instance as its refined label, while excluding samples previously marked with noisy from consideration. As a result, we obtain a clean training dataset. Utilizing this purified training set, we can develop a robust model for malicious traffic detection.

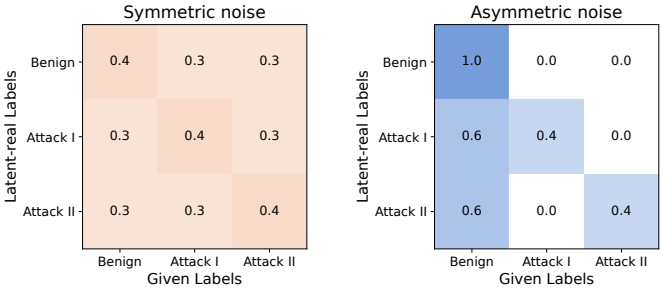

Fig. 3: Illustration of symmetric (left) and asymmetric (right) scenarios at 60% noise ratio.

IV. Experiments

In this section, we evaluate the effectiveness of our proposed framework on two malicious traffic datasets. We also compare the performance of the state-of-the-art LNL methods with our work.

A. Datasets

1) Malicious_TLS dataset [16]: We construct a more comprehensive dataset, named Malicious_TLS, which encompasses 22 types of encrypted malicious traffic. Over a span of four years, from 2018 to 2021, we captured benign TLS traffic and encrypted malicious traffic from real edge devices. In comparison to existing malicious traffic datasets, our dataset includes a more extensive range of malicious traffic categories and realistic traffic characteristics. With multiple sources of threat intelligence, we ensure that the labels in the dataset are accurate. The dataset is publicly available in https://github.com/gcx-Yuan/Malicious_TLS.

2) BoT-IoT dataset [8]: The BoT-IoT dataset, widely used in the study of malicious traffic, originates from the Cyber Range Lab of UNSW Canberra. This dataset is developed on a simulated environment, featuring network traffic generated by IoT botnet attacks.

B. Noise Settings

In alignment with BoAu [16], we introduce two distinct label noise scenarios: symmetric and asymmetric. Fig. 3 displays schematic diagrams of both label noise scenarios at 60% noise rate, respectively. Symmetric noise denotes that the labels of each traffic class are randomly flipped to other labels. For instance, some datasets are manually labeled by analyzing the features, which can easily result in labeling errors caused by either a lack of expertise or inadvertent mistakes. These errors can occur across all traffic categories. However, in asymmetric scenarios, only the labels of malicious traffic could be wrongly labeled as benign. For example, when utilizing multiple IDSs to assign labels to samples through a

TABLE I: Comparison with state-of-the-art methods in accuracy result (%) on Malicious_TLS

Noise type	Sym.				Asym.	
Methods/Noise rate	20%	40%	60%	80%	20%	40%
Baseline	82.53	80.88	71.66	53.39	78.23	55.50
GCE [10]	77.20	76.82	51.28	41.40	36.25	36.25
RAD [11]	89.28	88.64	88.08	43.25	87.35	53.88
Co-Teaching [5]	79.86	62.77	41.37	36.25	83.01	65.54
DivideMix [3]	82.73	76.18	72.32	47.84	75.96	47.82
UNICON [4]	84.43	82.17	68.82	49.18	77.60	53.68
Deep k-NN [6]	86.64	85.35	84.61	67.54	79.94	68.39
Ours	**91.70**	**90.95**	**90.48**	**90.46**	**90.55**	**85.85**

TABLE II: Comparison with state-of-the-art methods in accuracy result (%) on BoT-IoT

Noise type	Sym.				Asym.	
Methods/Noise rate	20%	40%	60%	80%	20%	40%
Baseline	83.86	80.67	79.24	51.88	85.57	69.81
GCE [10]	92.87	87.49	83.83	22.20	92.06	11.11
RAD [11]	93.11	92.56	87.04	30.78	91.16	85.05
Co-Teaching [5]	97.36	96.26	95.93	84.34	97.56	82.38
DivideMix [3]	88.63	87.88	84.05	82.12	88.73	82.65
UNICON [4]	87.09	89.57	81.96	73.64	86.80	81.91
Deep k-NN [6]	92.22	91.62	87.64	85.90	91.61	91.17
Ours	**98.65**	**97.39**	**98.18**	**97.27**	**99.00**	**97.35**

voting mechanism, the prevailing strategy involves minimizing false positives to enhance system availability. Consequently, a large amount of malicious traffic may be incorrectly labeled as benign, thereby introducing asymmetric noise.

C. Experimental Setup

In the training phase, we utilize an SGD optimizer. The hyperparameters are set as follows: a learning rate of 0.01, a batch size of 128, 10 pre-training epochs, and a total of 100 epochs for training. Notably, we choose $r = 0.5$ to make a balance, maximizing clean sample coverage while effectively reducing noise. We set $k = 100$ to capture comprehensive local similarity information for each sample, and σ is set to 0.5 to strike a balance between the two similarity rules. We mainly use accuracy rates to measure the performance of all methods. All experiments are performed using PyTorch 1.13.0 and trained on a Linux server equipped with an Intel(R) Core(TM) i9-10920X@3.50GHz, 256GB RAM, and an NVIDIA GeForce RTX 3080 GPU.

D. Comparison Methods

We compare our framework with several popular LNL methods, all of which utilize an identical malicious traffic classification network structure. To replicate DivideMix and UNICON for the malicious traffic dataset, we devise a simple data augmentation strategy that randomly adds perturbations to some features or removes features (by setting values to 0). These methods can be summarized as follows.

Baseline is trained directly using the original training dataset containing noisy labels.

GCE [10] is a robust loss function for label noise, which combines the advantages of MAE and CE. Following the previous work, we set hyperparameter q = 0.7.

RAD [11] is a label calibration framework that relies on a multi-model voting prediction strategy. A small amount of clean data is necessary for initializing the framework.

Co-Teaching [5] trains two networks simultaneously, and cross-training the other to provide potentially clean labeled data based on the small loss criterion.

DivideMix [3] models the per-sample loss distribution and dynamically divides the training data into clean (labeled) and noise (unlabeled), converting the LNL task into an SSL task.

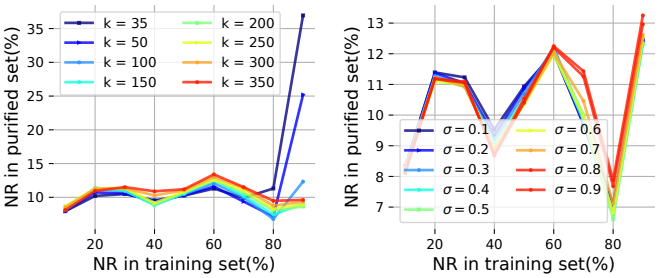

Fig. 4: Effect of hyperparameter k and fusion coefficient σ settings on purified dataset noise rate (NR).

UNICON [4] improves the data selection strategy of DivideMix, ensuring class balance of the divided labeled data. Contrastive learning is employed to enhance SSL.

Deep k-NN [6] utilizes a simple k-nearest-neighbor filtering method on the feature space to remove the mislabeled data.

E. Experimental Results and Analysis

Tables I and II present the classification accuracy of our method and other state-of-the-art methods on both datasets from moderate to severe label noise, respectively. It is clear that our method outperforms all competitors, achieving the highest classification accuracy and robustness, especially in highly noisy scenarios. When the noise rate exceeds 60%, our method maintains an accuracy of over 90% while other methods experience an obvious degradation in robustness. Among them, Deep k-NN demonstrates notable effectiveness on traffic datasets, which proves the significance of neighbor information for identifying noisy traffic data. However, as the noise rate increases, the impact of noisy labels on neighbor information becomes more pronounced. DivideMix and UNICON fail to exhibit exceptional performance on traffic data due to the lack of reasonable data augmentation schemes. Co-Teaching demonstrates great performance on BoT-IoT, but its effectiveness is limited when applied to the real malicious traffic dataset with a large number of categories. Neither RAD nor GCE exhibits strong performance in high-noise scenarios.

As shown in Fig. 4, we investigate the impact of varying hyperparameters, k and σ, on the noise rate within the purified dataset. For the number of neighbors k, the difference is negligible when the noise rate is below 70%. However, at high noise rates, a reduction in k leads to a substantial decrease

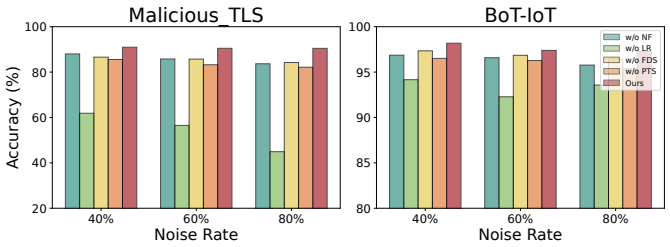

Fig. 5: Ablation experiment results on two datasets.

in noise reduction in the purified dataset. The experimental results also reveal that variations in σ have a certain effect on the noise rate in the purified dataset and a σ value near 0.5 demonstrates better performance in high-noise scenarios.

F. Ablation Study

In this subsection, we conduct an ablation study to assess the effectiveness of key components.

- ***w/o NF*** - we skip the noisy sample filtering and proceed directly to label refinement.
- ***w/o LR*** - we skip label refinement and train the model using the filtered dataset.
- ***w/o FDS*** - the k-NN topology graph is constructed based solely on probabilistic trend similarity.
- ***w/o PTS*** - the k-NN topology graph is constructed based solely on feature distance similarity.

As shown in Fig. 5, we assess the performance of our framework on both datasets after removing essential components. The results show that the model's performance degrades significantly when the topology graph is constructed directly on the noisy dataset without prior noise filtering. The label noise is effectively reduced through preliminary filtering while preserving a comprehensive and clean sample distribution, thus enhancing the subsequent label refinement. The most pronounced performance degradation is observed when the detection model is only trained on the filtered dataset, emphasizing the crucial role of label refinement. The results also demonstrate a substantial decline in the model's performance when relying solely on FDS or PTS. Considering only FDS neglects the essential information that samples of the same real class exhibit similar optimization dynamics. Conversely, relying solely on PTS overlooks high-dimensional information in the feature space.

V. Conclusion

In this paper, we proposed a novel two-stage framework for malicious traffic detection from low-quality training data. In the first stage, we filtered out most of the noisy samples from the training data while ensuring clean samples' integrity in the filtered dataset. In the second stage, we presented a double-constrained similarity rule, which offered a comprehensive assessment of sample similarity and facilitated the construction of the topological graph. Subsequently, we utilized the topological relationships extracted from this graph to refine the low-quality labels. By performing extensive experiments on multiple datasets, we demonstrated that our framework worked significantly better than the state-of-the-art methods. In the future, we will conduct a more comprehensive study on the fundamental distinctions between noisy and clean samples. Additionally, we will validate the generalizability of our framework over more scenarios.

References

[1] Steve TK Jan, Qingying Hao, Tianrui Hu, Jiameng Pu, Sonal Oswal, Gang Wang, and Bimal Viswanath. Throwing darts in the dark? detecting bots with limited data using neural data augmentation. In *IEEE symposium on security and privacy*, pages 1190–1206. IEEE, 2020.

[2] Giovanni Apruzzese, Pavel Laskov, and Aliya Tastemirova. Sok: The impact of unlabelled data in cyberthreat detection. In *IEEE European Symposium on Security and Privacy*, pages 20–42. IEEE, 2022.

[3] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020.

[4] Nazmul Karim, Mamshad Nayeem Rizve, Nazanin Rahnavard, Ajmal Mian, and Mubarak Shah. Unicon: Combating label noise through uniform selection and contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9676–9686, 2022.

[5] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

[6] Dara Bahri, Heinrich Jiang, and Maya Gupta. Deep k-nn for noisy labels. In *International Conference on Machine Learning*, pages 540–550. PMLR, 2020.

[7] Hui Tang and Kui Jia. Towards discovering the effectiveness of moderately confident samples for semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14658–14667, 2022.

[8] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova, and Benjamin Turnbull. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Generation Computer Systems*, 100:779–796, 2019.

[9] Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67, 2010.

[10] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[11] Zilong Zhao, Robert Birke, Rui Han, Bogdan Robu, Sara Bouchenak, Sonia Ben Mokhtar, and Lydia Y Chen. Enhancing robustness of on-line learning models on highly noisy data. *IEEE Transactions on Dependable and Secure Computing*, 18(5):2177–2192, 2021.

[12] Cedric Seger. An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing, 2018.

[13] Yubo Hou, Sin G Teo, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, and Tram Truong-Huu. Handling labeled data insufficiency: Semi-supervised learning with self-training mixup decision tree for classification of network attacking traffic. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[14] Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242. PMLR, 2017.

[15] Ganlong Zhao, Guanbin Li, Yipeng Qin, Feng Liu, and Yizhou Yu. Centrality and consistency: two-stage clean samples identification for learning with instance-dependent noisy labels. In *European Conference on Computer Vision*, pages 21–37. Springer, 2022.

[16] Qingjun Yuan, Chang Liu, Wentao Yu, Yuefei Zhu, Gang Xiong, Yongjuan Wang, and Gaopeng Gou. Boau: Malicious traffic detection with noise labels based on boundary augmentation. *Computers & Security*, 131:103300, 2023.