Gedss: A Generic Framework to Enhance Model Robustness for Intrusion Detection on Noisy Data

Lingfeng Yao*, Anran Hou*, Weina Niu*†, Qingjun Yuan‡, Junpeng He* and Yanfeng Zhang§
*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China
†Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen, China
‡Henan Key Laboratory of Network Cryptography Technology, Zhengzhou, China
§Intelligent Policing Key Laboratory of Sichuan Province, Sichuan Police College, Luzhou, China

***Abstract*—Training a deep neural network-based intrusion detection system requires a large amount of clean labeled data, yet malicious traffic datasets are usually collected from the open-source web community or simulated attack environments, which inevitably contain a large portion of unreliably labeled traffic data. The state-of-the-art methods dealing with label noise combine sample separation and semi-supervised learning (SSL), however, they are hardly usable in the traffic field because traffic data lacks a reasonable data augmentation like image data. To this end, we propose a generic label-noise-resistant framework for malicious traffic detection called Gedss. Unlike previous approaches focusing on data augmentation, our approach improves model performance by enhancing the quality of sample selection and model decision boundaries. The framework contains two parts: sample selection and semi-supervised learning. The sample selection method is presented to divide the original traffic instances into clean ones (labeled set) and noisy ones (unlabeled set). We fit a Jensen-Shannon divergence-based sample prediction loss to a mixture model as the criterion, and the threshold is automatically and dynamically adjusted, which makes our selection mechanism adaptive to various malicious traffic datasets. Besides, a semi-supervised learning method is designed, which uses two networks to jointly predict the pseudo label of the unlabeled set. Considering the class imbalance of divided labeled data, the idea of fine-tuning the models with all data is presented to improve the performance of SSL. Extensive experimental results under different label noise scenarios demonstrate that our approach outperforms state-of-the-art methods.**

***Index Terms*—Label noise, Semi-supervised learning, Intrusion detection, Deep learning, Unreliable data**

I. Introduction

The network intrusion detection system (NIDS) is an effective method to discover malicious activities from network traffic. However, the traditional rule-based network intrusion detection system needs to extract features manually and has a high false positive rate. NIDS based on deep learning (DL) could automatically extract features and learn complicated patterns from traffic data [1]. These DL-based methods perform well with massive clean labeled instances. However, public datasets for intrusion detection systems obtained from open-source web communities or simulated attack environments [2] [3] are hard to be clean, where labels are generated through manual or machine annotation [4] [5]. The manual annotation may result in errors and inconsistencies because of individual carelessness, while machine annotation could be inaccurate due to the complexity of the datasets or the limitations of the annotation algorithms. These incorrectly labeled data, i.e., label noise, may affect the generalization performance of NIDS, as deep neural networks (DNNs) are capable of memorizing noise data [6].

Many studies have been proposed to solve the label noise problem. Xu *et al.* [4] proposed a label noise reduction framework for malware detection by applying an unsupervised outlier detection method to the sample feature vectors. Zhao *et al.* [7] presented a voting-based label proofreading framework for IoT intrusion detection datasets. However, this ensemble model relied on a small number of clean data, which did not satisfy the realistic scenarios. The most popular idea is to separate the noisy and clean samples based on the divergence of their prediction loss, thus transforming the learning with noisy labels (LNL) task into a semi-supervised learning (SSL) task. These methods [8]–[10] showed the effectiveness of the image datasets. However, they are inappropriate for traffic data since the traffic data lacks explainable data augmentation methods like image random cropping and rotation.

To apply the combination of sample separation and semi-supervised learning to the traffic domain, we propose Gedss, a label-noise-resistant framework for malicious traffic detection. Unlike previous methods that focused on improving model performance by data augmentation, our approach prioritizes refining sample selection and learning the distribution of difficult samples to enhance the model's decision boundaries. First, we divide the training set into two subsets and train two models, allowing them to cross-predict in order to mitigate the impact of noisy labels and prevent confirmation errors. To improve the quality of selected data, these two subsets are divided into labeled and unlabeled sets by a dynamic threshold. After that, a semi-supervised learning method is presented to obtain a malicious traffic classifier by utilizing labeled and unlabeled data. Our main contributions are summarized as follows:

- We introduce Cross-Divide, an automatic data selection method. It effectively divides noisy data from clean data by dynamically adjusting the threshold and refining sample selection strategies.
- A semi-supervised learning method for joint prediction is

This work was supported by the Opening Project of Intelligent Policing Key Laboratory of Sichuan Province (No. ZNJW2023KFQN003) and the National Key Research and Development Program of China under Grant 2023QY0101. (Corresponding author: Weina Niu.)

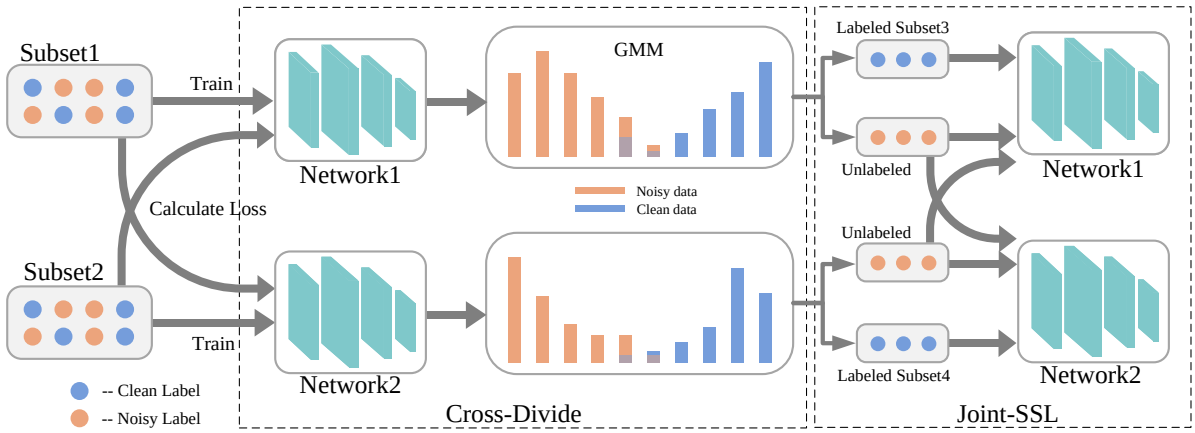


Fig. 1. Overview of the proposed Gedss. Network1 and Network2 are trained with Subset1 and Subset2, respectively. Then, a Cross-Divide part is designed, where these two models cross-predict another subset and utilize the Gaussian mixture model (GMM) to model the Jensen-Shannon (JS) prediction loss for each sample. Based on this loss, this sample is sorted into labeled or unlabeled datasets. Furthermore, a Joint-SSL part is proposed, where the labeled datasets named Subset3 and Subset4 are used for supervised learning to optimize the corresponding models, whereas the unlabeled datasets are jointly predicted by both models for semi-supervised learning.

proposed to train a malicious traffic classifier with labeled and unlabeled data. Moreover, model fine-tuning using all data is presented to address the class imbalance of divided clean data and improve the performance of semi-supervised learning.

- The effectiveness of our framework is validated on three popular malicious traffic datasets. Our approach outperforms state-of-the-art methods by achieving an average accuracy increase of more than 10% on the NSL-KDD, BoT-IoT and CICIDS2017 datasets. In high noise scenarios (with over 60% symmetric noise or 40% asymmetric noise), our approach performs even better, with an average accuracy increase of over 20%.

II. Problem Statement

When building an intrusion detection system, security professionals need a lot of labeled traffic data. Either they download the available intrusion detection datasets from open-source communities or collect and label network traffic manually. The labeling information contained in open-source traffic datasets in the former may be outdated due to the evolution of attacks. The latter also inevitably suffers from label noise due to careless human labeling and rigid machine annotation. Noisy labeled data can significantly impact the detection performance of DL-based models. Therefore, our goal is to obtain an effective intrusion detection model from noisy training data.

Let $\mathcal{D} = (x_i, y_i)^N$ denotes the training dataset with noisy labels, where x_i is the i-th traffic sample, y_i is the given label of x_i, and N denotes the total number of training samples. For each x_i, there exists one unique latent real label $\hat{y}_i$. If $y_i \neq \hat{y}_i$, then x_i is considered the noisy traffic, otherwise it is deemed clean.

III. The Proposed Framework

In this section, we propose a uniform framework to obtain a robust intrusion detection model from noisy data. The framework comprises two essential components: Cross-Divide and Joint-SSL. We call this integrated technique as Gedss, which stands for Generic Data Selection and Joint SSL. The overall framework of Gedss is shown in Fig.1.

A. Cross-Divide: Data selection

To effectively divide clean and noisy data, we introduce an enhanced sample selection strategy Cross-Divide. Our primary idea is to fit the cross-prediction loss of the samples into a double Gaussian distribution, thus separating noisy and clean samples more distinctly. Dynamic threshold is also employed to further improve the quality of sample selection.

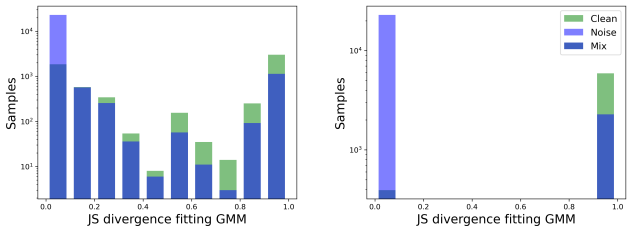
Two models, M_1 and M_2, sharing an identical network structure, are initialized for label noise detection, and the last layer of the network is Softmax. The dataset, denoted as $\mathcal{D}$, contains noisy data that is initially divided into two subsets $\mathcal{D}_1$ and $\mathcal{D}_2$ by random sampling. We make M_1 and M_2 pre-train several epochs with $\mathcal{D}_1$ and $\mathcal{D}_2$ datasets, respectively.

Subsequently, we employ one model to evaluate the training subset of the other model. We adopt the prediction Jensen-Shannon divergence (JSD) as the evaluation criterion and fit the results to a Gaussian mixture model (GMM). Compared with cross-entropy (CE) loss, the JS loss is symmetric in calculation and the range of values is between $[0, 1]$, which is more suitable to fit GMM. For a given sample x_i^k in $\mathcal{D}_k$ and $k \in \{1, 2\}$, the prediction probability from M_{3-k} can be denoted as $p_i = [p_i^1, p_i^2, \ldots, p_i^C]$ and its corresponding ground-truth label as $y_i = [y_i^1, y_i^2, \ldots, y_i^C]$, where C refers to the total number of traffic categories. The JSD of samples prediction losses is defined as follows,

$$\begin{aligned} d_i &= \mathrm{JSD}(y_i, p_i) \\ &= \frac{1}{2}\mathrm{KL}(y_i || \frac{y_i + p_i}{2}) + \frac{1}{2}\mathrm{KL}(p_i || \frac{y_i + p_i}{2}) \end{aligned} \tag{1}$$

where KL is the Kullback-Leibler divergence function.

After getting the JS divergence $\mathbb{D} = \{d_1, d_2, \ldots, d_{N/2}\}$ of every sample, the Expectation Maximization(EM) algorithm is used to fit a two-component GMM. For each sample, its



(a) (b)

Fig. 2. Training on Bot-IoT with 80% noise. (a) warm up after 10 epochs (b) training after 50 epochs

clean probability w_i is equivalent to the posterior probability $p(g|d_i)$, where g is the Gaussian component with smaller mean (smaller loss).

The prediction loss of noisy and clean samples fitting Gaussian distribution is shown in Fig.2(a), it is obvious that most of the noisy data are close to 0 while clean data approach 1. As shown in Fig.2(b), the prediction loss of noisy and clean samples gradually converges into the two ends over the training. A threshold w_{thres} is set to divide labeled data and unlabeled data based on $\mathcal{W} = \{w_i : i \in (1, \ldots, N)\}$. If the threshold is set too low, much noise will be introduced, and if the threshold is set too high, it is hard to represent the complete data distribution because of the little clean data selected. So a dynamic threshold is used to select the sample automatically. Since the noise and clean samples are not well divided in the early stage of training, a small threshold is defined to eliminate the noisy data as much as possible. At the end of the training, the difference between noisy and clean samples is quite clear, so a large threshold is adopted to select clean samples for training as much as possible. We let the threshold increase linearly from 0.2 to 0.8, which can be expressed as,

$$w_{thres} = 0.2 + 0.6 * \frac{e}{numEpoch} \tag{2}$$

To further avoid the risk of noisy memory, we employ a generalized cross-entropy (GCE) loss [11] which is robust against noisy labels. Ghosh *et al.* [12] have demonstrated that mean absolute error (MAE) can effectively restrain noisy data, but MAE has the problems of slow convergence and difficult training. And categorical cross entropy (CCE) is faster to train but less resistant to noise. GCE [11] combines the advantages of MAE and CCE and achieves results on noisy data. The robust loss function is defined as follows,

$$\mathcal{L}_q = \frac{1 - (\sum_{k=1}^{K} y_k \hat{y}_k)}{q} \tag{3}$$

where $q \in (0, 1]$ is a hyperparameter. When $q = 1$, the $\mathcal{L}_q$ loss becomes MAE, and when q approaches 0, the loss becomes CCE. In our experiments, we set $q = 0.7$. Algorithm.1 summarizes our divide method.

B. Joint-Semi supervised learning

Utilizing the Cross-Divide sample selection strategy, we categorize clean and noisy samples into labeled and unlabeled data in each epoch for semi-supervised learning. Algorithm.2 shows the details of our joint-semi supervised learning. The labeled data are used to train and fine-tune the model M_1, M_2, while the unlabeled data are used for semi-supervised learning. The labels of unlabeled data are jointly predicted with M_1 and M_2, which improves the quality of pseudo labels. For a sample $u_i \in \mathcal{D}_{unlabeled}$, the prediction probability from M_1, M_2 can be denoted as p_{1i}, p_{2i}, where $p_{1i} = [p_{1i}^1, p_{1i}^2, \ldots, p_{1i}^C]$, $p_{2i} = [p_{2i}^1, p_{2i}^2, \ldots, p_{2i}^C]$ and C for traffic class. Calculate the average predicted probability $p_i = (p_{1i} + p_{2i})/2$, where the class with the highest prediction probability is considered as the pseudo-label k. The loss of unlabeled samples is denoted as follows,

$$\mathcal{L}_u = \frac{1}{2}(CEloss(p_{1i}, k) + CEloss(p_{2i}, k)) \tag{4}$$

For labeled data x_i, we keep employing GCE instead of CE, and the loss of supervised learning is defined as follows,

$$\mathcal{L}_x = \frac{1}{2}(GCEloss(y_{predict}^{(1)}, y) + GCEloss(y_{predict}^{(2)}, y)) \tag{5}$$

The total loss function we minimize is

$$\mathcal{L}_{total} = \mathcal{L}_x + \lambda_u \mathcal{L}_u \tag{6}$$

where λ_u is the coefficient to control the strength of the unsupervised loss.

However, DNN usually learns easy classes first, which can lead to high prediction loss for hard classes [10]. Such samples are treated as noise, which leads to the problem of few difficult categories in the labeled data. To address the imbalance, we train the model with clean data every three times, and then

Algorithm 1 Cross-Divide

Require: training dataset $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$, number of samples N, warmEpoch = 10, numEpoch = 200, $\theta^{(1)}$ and $\theta^{(2)}$;

$\mathcal{D}_1$ = $\mathcal{D}$.sample($n = N/2$), $\mathcal{D}_2$ = $\mathcal{D}$ - $\mathcal{D}_1$;
$\mathcal{D}_1 = \left(\mathcal{X}^{(1)}, \mathcal{Y}^{(1)}\right)$, $\mathcal{D}_2 = \left(\mathcal{X}^{(2)}, \mathcal{Y}^{(2)}\right)$;
while $e <$ warmEpoch **do**
  for $k = 1, 2$ **do**
    $\theta^{(k)}$ = warmup($\mathcal{X}^{(k)}$,$\mathcal{Y}^{(k)}$,$\theta^{(k)}$);
  end for
end while
Using prediction loss to divide the dataset
while warmEpoch $< e <$ numEpoch **do**
  for $k = 1, 2$ **do**
    for $i = 1$ to $N/2$ **do**
      $\hat{p}_i^k$ = model(x_i^k,θ^{3-k});
      d_i^k = calculateJSD($\hat{p}_i^k$,y_i^k);
      $\mathbb{D}^{(k)} = \left\{d_1^k, d_2^k, ..., d_{N/2}^k\right\}$;
    end for
  end for
  $\mathcal{W}^{(1)}$ = GMM($\mathbb{D}^{(1)}$, n_components=2);
  $\mathcal{W}^{(2)}$ = GMM($\mathbb{D}^{(2)}$, n_components=2);
end while
return $\mathcal{W}^{(1)}$, $\mathcal{W}^{(2)}$;

Algorithm 2 Joint-SSL

Require: $\mathcal{W}^{(1)}$ and $\mathcal{W}^{(2)}$;
while warmEpoch $< e <$ numEpoch **do**
  Determine the threshold, w^k_{thres} using Eq.(1)
  $\mathcal{W}^{(1)}$, $\mathcal{W}^{(2)}$ = CrossDivide(e, $\theta^{(1)}$, $\theta^{(2)}$)
  for $i = 1$ to N/2, $k = 1, 2$ **do**
    $\mathcal{D}^{(k)}_{labeled} \leftarrow \{(x^k_i, y^k_i) : w^k_i \geq w^k_{thres} :\}$
    $\mathcal{D}^{(k)}_{unlabeled} \leftarrow \{(x^k_i, y^k_i) : w^k_i < w^k_{thres}\}$
  end for
  Initialize Model1, Model2 with $\theta^{(1)}, \theta^{(2)}$
  for iter = 1 to num_iters, $k = 1, 2$ **do**
    $\mathcal{L}^k_{\mathcal{X}}, \theta^{(k)}$ = $\mathrm{M}_k(\mathcal{D}^{(k)}_{labeled}, \theta^{(k)})$
    $\hat{\mathcal{Y}}_k$ = $\mathrm{M}_k(\mathcal{D}^{(k)}_{unlabeled}, \mathcal{D}^{(3-k)}_{unlabeled}, \theta^{(k)})$
    $\mathcal{L}_{\mathcal{U}}$ = semiLoss($\hat{\mathcal{Y}}_k$,$\hat{\mathcal{Y}}_{3-k}$)
  end for
  $\mathcal{L} = \mathcal{L}_{\mathcal{U}} + \mathcal{L}_{\mathcal{X}}$
  Update $\theta^{(1)}, \theta^{(2)}$;
end while
return $\theta^{(1)}, \theta^{(2)}$;

fine-tune the model with full data. And a noise-resistant loss function is employed, which tends to learn samples with lower prediction loss. The model trained using clean data has certain prior knowledge that mislabeled samples in difficult classes have higher prediction loss, so the loss function can enable the model to ignore them and learn only the samples with small prediction loss i.e., labels are likely to be more accurate.

IV. Experimental Analysis

In this section, we validate our proposed Gedss on three popular traffic datasets under different label noise ratios.

A. Dataset

1) BoT-IoT dataset [2]: BoT-IoT was created in the Cyber Range Lab of UNSW Canberra, which has been widely used in the study of malicious traffic detection. This dataset is developed on a simulated environment, featuring network traffic generated by IoT botnet attacks.

2) CIC-IDS2017 dataset [3]: CIC-IDS2017 was developed by Canadian Institute for Cybersecurity. The attacks were launched on 12 devices with different operating systems. It included the most common attacks based on the 2016 McAfee report.

3) NSL-KDD dataset [13]: NSL-KDD was created by the University of New Brunswick Canadian Institute for Cybersecurity to address the flaws in the classic intrusion detection dataset KDDCup99.

B. Noise Settings

We employ two types of noise models to simulate the real world: symmetric and asymmetric. We refer to the previous work [14] to set up our noise scenarios. In the symmetric scenario, the labels of an r portion of one particular traffic may be randomly flipped to all other classes. In the asymmetric scenario, only the labels of malicious traffic would be incorrectly labeled as benign samples.

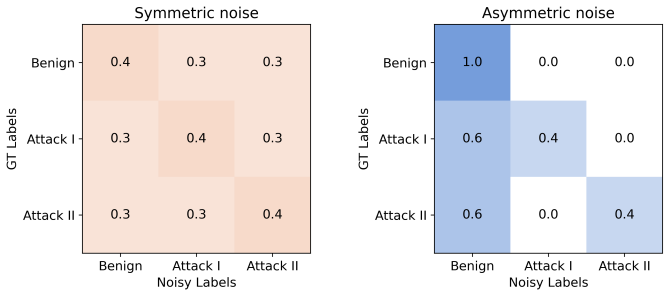


Fig. 3. Illustration of symmetric (left) and asymmetric (right) scenarios at 60% noise ratio.

As shown in Fig.3, we demonstrate the visualization of the confusion matrix plots for symmetric and asymmetric noise at 60% noise ratio, where the dataset contains benign traffic and two types of attack traffic. The y-axis denotes the ground truth (GT) label of the sample, the x-axis denotes the label of the sample in the noisy dataset, and the decimals in the matrix represent the proportions of the corresponding labels. The left side represents the symmetric noise scenario, where 60% of all categories of traffic are randomly flipped into labels of other categories. The right side represents the asymmetric noise scenario, where only the labels of attack traffic are converted into benign traffic labels.

C. Experimental Setup

In the training phase, we train the model with an Adam optimizer. The learning rate is set as 0.01 to optimize the network with 200 epochs. All of the experiments were performed using Pytorch 1.13.0 and trained on a Linux server with Intel(R) Core(TM) i9-10920X@3.50GHz, 256GB RAM, and an NVIDIA GeForce RTX3080 GPU.

D. Baseline methods

We compare the performance of some state-of-the-art robust learning methods with our framework Gedss. Specifically, to replicate DivideMix and UNICON, we design a data augmentation approach that consists of randomly adding noise to certain features and randomly deleting certain features (by setting the feature values to 0). The details of all competitors are as follows.

GCE [11]: Generalized cross-entropy (GCE) loss is a combination of mean absolute error (MAE) loss and cross-entropy (CE) loss, integrating the advantages of both loss to be robust to label noise and to optimize the network easily.

RAD [7]: Robust Anomaly Detection (RAD) is a label calibration framework based on multi-model voting, and this integrated model requires a small number of clean labeled data for initialization.

Co-Teaching [8]: Coteaching iteratively selected a fraction of small-loss data as clean data to train another network.

DivideMix [9]: DivideMix improved the sample selection mechanism of Co-Teaching by fitting the loss to a Gaussian

TABLE I
COMPARISON WITH STATE-OF-THE-ART METHODS IN ACCURACY RESULT (%) ON NSL-KDD

Noise type	Sym.				Asym.	
Methods/Noise ratio	20%	40%	60%	80%	20%	40%
Standard CE	95.11	91.70	88.55	81.12	95.25	88.45
GCE [11]	89.22	88.09	71.29	33.33	88.89	26.54
RAD [7]	95.48	93.67	88.79	59.54	94.18	73.23
Co-Teaching [8]	95.89	94.97	93.27	87.67	**96.20**	91.17
DivideMix [9]	95.65	95.08	93.00	85.87	94.68	88.82
UNICON [10]	**95.98**	95.25	93.74	87.28	95.55	91.80
Gedss	95.95	**95.78**	**94.71**	**91.70**	95.92	**94.54**

TABLE II
COMPARISON WITH STATE-OF-THE-ART METHODS IN ACCURACY RESULT (%) ON CICIDS2017

Noise type	Sym.				Asym.	
Methods/Noise ratio	20%	40%	60%	80%	20%	40%
Standard CE	96.13	92.90	82.49	62.94	96.50	88.90
GCE [11]	97.37	97.14	96.30	12.96	97.75	11.11
RAD [7]	97.70	97.64	94.27	35.76	97.74	57.21
Co-Teaching [8]	99.05	93.13	25.28	11.16	99.03	97.96
DivideMix [9]	95.87	95.03	93.61	93.54	97.27	93.92
UNICON [10]	97.18	96.70	96.19	92.66	97.06	92.44
Gedss	**99.22**	**99.07**	**98.75**	**97.86**	**99.15**	**98.76**

Mixture Model (GMM) and using a fixed threshold to divide clean and noisy data.

UNICON [10]: UNICON improved the sample selection mechanism of DivideMix by maintaining the class balance of the selected data and introducing contrastive learning in semi-supervised learning.

E. Experimental Results And Evaluation

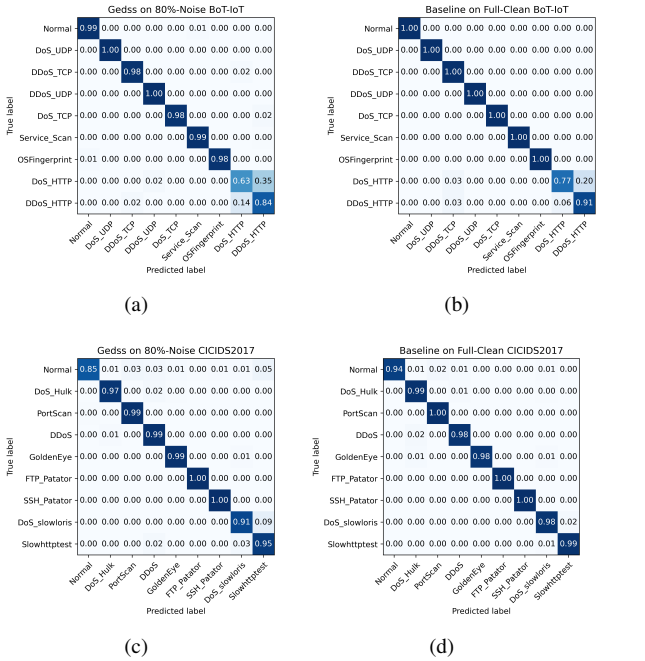
We present the performance of Gedss and other SOTA methods on three traffic datasets. Based on the noise setup approach described above, we considered symmetric noise 20%, 40%, 60%, and 80%, asymmetric noise 20% and 40%. We use accuracy to evaluate the performance of the proposed model.

1) Comparison with SOTA methods: Table I, Table II and Table III show the accuracy results on NSL-KDD, CICIDS2017, and BoT-IoT datasets, respectively. Gedss exhibits advanced detection performance in various noisy scenarios due to the improved data selection strategy that efficiently separates noisy and clean traffic data. Fine-tuning the model to learn the distribution of hard samples improves the model's ability to identify decision boundaries. Therefore, our method has superior detection performance even in high-noise scenarios (over 60% symmetric noise or 40% asymmetric noise), averaging 12.92%, 31.88%, and 24.78% higher accuracy than other methods.

DivideMix and UNICON perform well in low-noise scenarios, but the performance of the models degrades significantly as the noise rate increases. The probable reason is that both methods are aimed at image data, which rely on various data augmentation methods for the field of computer vision, however, these augmentation methods are not feasible on traffic

TABLE III
COMPARISON WITH STATE-OF-THE-ART METHODS IN ACCURACY RESULT (%) ON BOT-IOT

Noise type	Sym.				Asym.	
Methods/Noise ratio	20%	40%	60%	80%	20%	40%
Standard CE	83.86	80.67	79.24	51.88	85.57	69.81
GCE [11]	92.87	87.49	83.83	22.20	92.06	11.11
RAD [7]	93.11	92.56	87.04	30.78	91.16	85.05
Co-Teaching [8]	97.36	96.26	95.93	84.34	97.56	82.38
DivideMix [9]	88.63	87.88	84.05	82.12	88.73	82.65
UNICON [10]	87.09	89.57	81.96	73.64	86.80	81.91
Gedss	**98.91**	**98.87**	**98.28**	**96.08**	**98.50**	**96.62**



(a) (b) (c) (d)

Fig. 4. Classification results of Gedss and Baseline.

data. Co-Teaching has advanced results in low and medium noise scenarios. However, since the method does not consider the class imbalance problem in data selection, the model tends to choose simple classes in high-noise scenarios, resulting in low model performance or even training failure. Both RAD and GCE outperform standard CE at low noise levels but almost completely fail at high noise. Our experiments also show that the standard cross-entropy loss shows robustness to some extent under lower noise conditions.

2) Classification performance evaluation: Next, we evaluate the multi-classification performance of the proposed method. Fig.4(a) and Fig.4(c) illustrate the classification results of Gedss on two datasets under 80% noise. Fig.4(b) and Fig.4(d) show the classification results of a DNN trained on ideal datasets (with fully accurate labels). The classifier obtained by Gedss with 80% noise data has a similar performance to the classifier trained with the ideal datasets, which demonstrates the excellent robustness of our method. As shown in Fig.4(a) and Fig.4(b), the classification performance of DoS_HTTP and DDoS_HTTP is relatively poor. This is because both of them attack by initiating HTTP requests so

TABLE IV
RESULT OF ABLATION ON BOT-IOT

Methods/Noise ratio	40%	50%	60%	70%	80%	Average
Gedss w/o DT	98.13	97.92	97.93	97.43	95.67	97.42(**+0.48%**)
Gedss w/o FT	97.46	97.57	97.66	97.16	94.77	96.92(**+1.03%**)
Gedss w/o JS	97.88	97.81	97.39	96.85	93.50	96.69(**+1.59%**)
Gedss	**98.87**	**98.61**	**98.28**	**97.91**	**96.08**	**97.95**

TABLE V
RESULT OF ABLATION ON CICIDS2017

Methods/Noise ratio	40%	50%	60%	70%	80%	Average
Gedss w/o DT	98.65	98.41	97.74	97.05	93.29	97.03(**+1.72%**)
Gedss w/o FT	96.72	96.33	95.94	95.57	94.38	95.79(**+2.96%**)
Gedss w/o JS	98.63	98.17	98.05	96.52	94.30	97.13(**+1.62%**)
Gedss	**99.07**	**98.75**	**98.75**	**97.61**	**97.86**	**98.41**

they have similar features. And the number of these two types of attacks in the BoT-IoT dataset is small, making it difficult for the classifier to learn their complete feature distribution.

F. Ablation Experiments

In this section, we perform several ablation experiments to verify the effectiveness of the key components of the proposed Gedss. We formulate three methods: (1) Gedss does not use a dynamic threshold, but uses a fixed threshold of 0.5 to divide the data, called Gedss w/o DT. (2) Gedss does not fine-tune the model with all data, and only uses the clean data divided, termed Gedss w/o FT. (3) Instead of fitting the JS divergence to the GMM distribution when dividing the data, Gedss uses the CE loss to fit to the GMM distribution, which we call Gedss w/o JS.

As shown in Table IV and Tables V, we test the symmetric noise 40%-80% scenarios on two datasets, BoT-IoT and CICIDS2017, and the results show a varying degree of decline in the performance of the models due to the missing of important components. A potential problem caused by using a constant threshold is that if the threshold is set too low, many noisy samples are introduced, and if the threshold is set too high, fewer clean samples are selected, making it difficult to represent the entire data distribution. It can be seen that not fine-tuning the model brings a large performance drop. This is because methods based on selecting clean samples with small losses are biased toward learning easy samples, thus ignoring hard samples that provide a huge benefit to the model's predictive performance. Fine-tuning the model with the entire data exposes the distribution of this hard sample to the model and improves the model's ability to identify decision boundaries. The experimental results also prove that the effect of dividing the data by fitting to the GMM distribution through JS divergence is better than fitting to the GMM distribution through CE loss, this is because the range of JS divergence is between 0-1, which is more suitable to be fitted to the GMM distribution, whereas the CE loss needs to be normalized before fitting, which will lose some information.

V. CONCLUSION

In this paper, we proposed Gedss, a generic label-noise-resistant framework for malicious traffic detection. By fitting the JSD-based sample prediction loss to GMM as a criterion, the quality of sample selection in Gedss was significantly improved. The threshold for our sample selection was automatically and dynamically adjusted, which makes Gedss adaptive to various malicious traffic datasets with different noise ratios without adjusting hyperparameters. Considering the class imbalance of divided clean data, the idea of fine-tuning the models by using all data was presented to improve the performance of semi-supervised learning. By performing extensive experiments on multiple datasets, we demonstrated that Gedss consistently worked significantly better compared to state-of-the-art methods.

REFERENCES

[1] Jin Kim, Nara Shin, Seung Yeon Jo, and Sang Hyun Kim. Method of intrusion detection using deep neural network. In *2017 IEEE international conference on big data and smart computing*, pages 313–316. IEEE, 2017.

[2] 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.

[3] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1:108–116, 2018.

[4] Jiayun Xu, Yingjiu Li, and Robert H Deng. Differential training: A generic framework to reduce label noises for android malware detection. *Network and Distributed System Security Symposium*, 2021.

[5] Muhammad Fahim and Alberto Sillitti. Anomaly detection, analysis and prediction techniques in iot environment: A systematic literature review. *IEEE Access*, 7:81664–81681, 2019.

[6] Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks, 2017.

[7] 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.

[8] 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.

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

[10] 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.

[11] 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.

[12] Aritra Ghosh, Himanshu Kumar, and P Shanti Sastry. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[13] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*, pages 1–6. Ieee, 2009.

[14] 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.