Hindawi
Security and Communication Networks
Volume 2021, Article ID 3017797, 12 pages
https://doi.org/10.1155/2021/3017797

Research Article

GTF: An Adaptive Network Anomaly Detection Method at the Network Edge

Renjie Li,[1,2,3] **Zhou Zhou**,[1,2] **Xuan Liu**,[4,5] **Da Li**,[6] **Wei Yang**,[1,2] **Shu Li**,[1,2] **and Qingyun Liu**[1,2,3]

[1]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China*
[2]*National Engineering Laboratory for Information Security Technology, Beijing, China*
[3]*School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China*
[4]*College of Information Engineering (College of Artificial Intelligence), Yangzhou University, Yangzhou, China*
[5]*School of Computer Science and Engineering, Southeast University, Nanjing, China*
[6]*Department of Electrical and Computer Engineering, University of Missouri-Columbia, Columbia, USA*

Correspondence should be addressed to Wei Yang; yangwei@iie.ac.cn

Received 22 September 2021; Accepted 17 November 2021; Published 20 December 2021

Academic Editor: Yuyu Yin

Network Anomaly Detection (NAD) has become the foundation for network management and security due to the rapid development and adoption of edge computing technologies. There are two main characteristics of NAD tasks: tabular input data and imbalanced classes. Tabular input data format means NAD tasks take both sparse categorical features and dense numerical features as input. In order to achieve good performance, the detection model needs to handle both types of features efficiently. Among all widely used models, Gradient Boosting Decision Tree (GBDT) and Neural Network (NN) are the two most popular ones. However, each method has its limitation: GBDT is inefficient when dealing with sparse categorical features, while NN cannot yield satisfactory performance for dense numerical features. Imbalanced classes may downgrade the classifier's performance and cause biased results towards the majority classes, often neglected by many exiting NAD studies. Most of the existing solutions addressing imbalance suffer from poor performance, high computational consumption, or loss of vital information under such a scenario. In this paper, we propose an adaptive ensemble-based method, named GTF, which combines TabTransformer and GBDT to leverage categorical and numerical features effectively and introduces Focal Loss to mitigate the imbalance classification. Our comprehensive experiments on two public datasets demonstrate that GTF can outperform other well-known methods in both multiclass and binary cases. Our implementation also shows that GTF has limited complexity, making it be a good candidate for deployment at the network edge.

1. Introduction

In the past few decades, the Internet of Things (IoT) and cloud services have penetrated many aspects of our lives and served quantities of applications, for example, automated vehicles, medical applications, industrial IoT, and cloud data centers [1–4]. These emerging applications have shown considerable potential in improving the quality of life and network services. However, the proliferation of these new technologies also has led to an increasing trend of cyberspace attacks and other threats, making security concerns still hamper IoT adoption. Reportedly, the losses caused by cybercrime in the United States exceeded $4.2 billion in 2020 [5]. As a result, network security is a critical concern in our daily lives and business operations. There is an urgent need for efficient and reliable anomaly detection mechanisms to shield our network.

Traditional NAD methods, such as firewalls and rule-based Network Intrusion Detection Systems, are often insufficient to detect unknown attacks due to the inability to keep up with the most recent and sophisticated attacks. With the prevalent application of artificial intelligence, machine learning (ML), especially deep learning (DL), has attracted much attention in edge computing and cloud computing [6–10], due to its advantages in discovering

hidden patterns from vast amounts of data. ML/DL techniques are now widely used for the purpose of NAD, enhancing the security of the networking infrastructure and crucial data.

A typical ML/DL-based NAD method, which aims to detect anomalous network traffic by observing traffic data over time to distinguish potential attacks from normal traffic, usually takes the tabular data as the input and reads the data in CSV format. The tabular data consists of series of network traffic records, each of which is a network connection session (or a flow) and is labeled as either normal or a specific attack type. In particular, the tabular input means that the input features of a NAD method can have both categorical and numerical ones. For example, transaction protocol types and service types are usually regarded as categorical ones, while the duration and source/destination bytes are numerical values. Therefore, a classification model must be able to learn effectively with tabular input data. In general, among traditional ML methods, decision-tree-based ensemble methods (e.g., Gradient Boosting Decision Tree, GBDT [11]) dominate the use cases for tabular input data due to their superior performance. On the other hand, the deep learning methods are more preferred for unstructured input data (e.g., images, speech, and text) [12]. Because of its popularity and performance, this paper focuses on GBDT. While some recent researches confirm that GBDT is still the most accurate method on tabular data [13, 14], others claim to outperform GBDT [15, 16] or come within a hair's breadth of GBDT's performance [17]. In general, each of them holds its pros and cons dealing with tabular data.

On the one hand, GBDT has better effectiveness in handling dense numerical features than sparse categorical features. Like many other tree-based models, GBDT can automatically collect and combine the helpful numerical features to fit the training targets properly by picking the features with the most significant statistical information gain to build the trees [18]. Since categorical features are generally converted to high-dimensional and sparse one-hot encodings, GBDT will obtain small information gain on sparse features. As a result, GBDT cannot handle categorical features efficiently. In addition, GBDT and other tree-based methods are fast to train and have better interpretability. On the other hand, DL methods' advantage mainly lies in their capability in handling sparse categorical features by learning parametric embeddings to encode categorical features and their power in learning from large-scale data. The main limitation of DL methods, such as Fully Connected Neural Network, is their shortcoming in learning with dense numerical features directly, mainly because of complex optimization hyperplanes and the risk of falling into local optimums [19]. Therefore, DL methods cannot match the performance of GBDT in many tasks and datasets [15, 20].

Another challenge of the NAD task is the class imbalance of the real-world network traffic captured by edge devices [21], making it challenging for the classifier to make decisions on such skewed data distribution. In such cases, learning-based classification methods are always designed to achieve the highest overall accuracy, which may produce a bias towards the majority class [22]. Similar scenarios also exist in other real-world applications, such as credit fraud detection [23] and medical diagnosis [24], but we focus on the NAD task at the network edge in this paper. Anomalies rarely occur, and normal data usually accounts for a large proportion. Furthermore, the minority class ordinarily carries the concepts with more significant interests than the majority class [25].

Accordingly, developing an adaptive method to address two major challenges of NAD tasks, that is, tabular inputting and imbalance problem, is desired. Inspired by some recent studies, we intend to combine Neural Networks and tree-based models to learn effectively from tabular data and introduce a well-designed loss function to deal with class imbalance. In this paper, we propose a novel method for the NAD task, called **GTF**, an ensemble of **G**BDT and **T**abTransformer enhanced with **F**ocal Loss. We explored the GBDT2NN [26] and the TabTransformer-based classifier [17] to handle numerical features and categorical features, respectively, as shown in Figure 1. As for class imbalance, we utilize Focal Loss, which is proposed in the field of object detection for solving the extreme foreground-background class imbalance, which degrades the first-stage detector's performance [27], to deal with imbalanced traffic classification. Besides, all of these methods are first aimed at binary classification problems, and we extend them to the multiclass NAD task. In summary, the main contributions of our work are listed as follows:

(i) We introduce a novel supervised NAD method with adaptive learning, named GTF, which improves the robustness and effectiveness for tabular data with class imbalance problems. Our method is applicable to various kinds of classification tasks but is particularly useful for NAD.

(ii) We consider the tabular input data of NAD tasks and introduce two advanced models, that is, TabTransformer and GBDT2NN. Our proposal combines the advantages of GBDT and NN to handle both categorical and numerical features efficiently.

(iii) By integrating Focal Loss, the proposed GTF can adapt to scenarios in which the performance suffers from class imbalance and compensate for the degradation of the classification model in such scenarios.

(iv) We also propose an adaptive learning framework for GTF to automatically search for optimal parameters without the expert's experience. Experiments demonstrate that GTF could achieve superior results on two well-known NAD datasets, that is, KDD'99 and UNSW-NB15, and achieve robust performance in both multiclass and binary cases.

(v) We evaluate the complexity of GTF in terms of computational requirements and runtime. Our analysis shows that GTF is really efficient and scalable. Thus, it is a good fit to deploy on constrained edge devices.

The rest of the paper is organized as follows. We summarize the related work in Section 2, followed by our proposed method in Section 3. In Section 4, we provide the

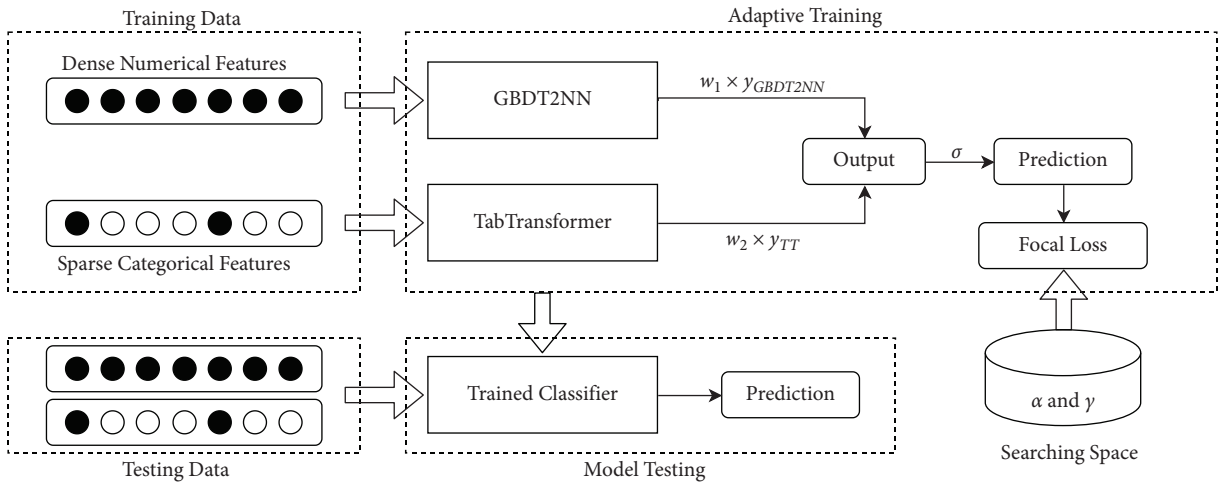

Figure 1: The framework overview of GTF. We use two modules to treat categorical and numerical features, respectively. The adaptive training module can learn optimal w_1 and w_2 through optimizer and find best α and γ used for Focal Loss from searching space automatically. For $\mathbf{y}_{TT}$, TT represents Tab Transformer.

experimental details and results. Finally, we draw the conclusion in Section 5.

2. Related Work

As aforementioned, each of GBDT and Neural Network has its own weaknesses when facing the tabular data. Since sparse categorical features may impair the growth of trees in GBDT because of tiny statistical information gain, some methods require to encode categorical features into dense numerical values, which can be handled well by tree-based models. Some GBDT methods can directly take categorical features as their inputs, such as LightGBM [28] and CatBoost [29]. For example, CatBoost transforms categorical features to numerical before each split is selected in the tree by using various statistics on combinations of categorical features and combinations of categorical and numerical features. However, it may cause information loss. Binary coding [30] is another choice to encode features, which enumerates possible binary partitions of categorical features. But this method may cause overfitting and bring bias when there is not enough data in each category [28]. Neural Networks have been applied in many fields, but they are not well suited for tabular data. They mainly focus on the sparse categorical features and pay less attention to the dense numerical features. Although NN generally employs normalization [31] or regularization [12] for numerical features before the training phase, they usually cannot outperform GBDT and fail to find optimal solutions for tabular decision manifolds. For learning effectively with tabular data, some recent researches also try to combine the advantages of NN and GBDT. Although these methods are believed to have decision ability like trees to some extent, they mainly focused on computer vision or click prediction tasks rather than the NAD task with tabular inputting. Moreover, they may suffer from some disadvantages, like being inefficient and redundant.

Imbalance classification, also known as Imbalance Learning, has been one of the most challenging problems in machine learning and deep learning. Many research works have been proposed to solve such problems and they can be summarized into three categories: data-level methods, algorithm-level methods, and cost-sensitive learning. To combat imbalance, typical algorithms adopt resampling techniques before training, such as SMOTE, ADASYN, NearMiss, and Tomek Link [32–35]. Most data-level methods have a distance-based design. On the one hand, resampling on large-scale data may lead to a high cost of computing the distance between samples. On the other hand, distance-based design may not be applicable for categorical features or missing values. Except for the time consumption of oversampling, undersampling may lose important information. Algorithm-level methods usually combine ensemble learning algorithms with advanced resampling techniques introduced to reduce the variance and they have achieved superior performance. But some of these ensemble methods are more time-consuming (e.g., SMOTEBagging and SMOTEBoost [36, 37]) and others have the risk of underfitting or overfitting (e.g., EasyEnsemble and BalanceCascade [38]). Cost-sensitive learning takes costs associated with the different classes into account. It mainly consists of two approaches: (1) assign the corresponding cost directly to each category and (2) employ metalearning during the training phase by preprocessing (usually data-level techniques) and postprocessing steps. However, some of them require a prerequisite of domain experts to set a cost matrix, and some have high computational complexity.

In summary, although there are increasing works that build more effective models and deal with skewed class distributions, most of them cannot completely solve the challenge of the NAD task (tabular input space and class imbalance). In this paper, we integrate NN and GBDT into a whole framework and adopt the advanced loss function to address the imbalance, which is suitable for real-world NAD datasets.

3. Methodology

In this section, we provide the formalized problem definition and our proposed method. Specifically, we focus on the imbalanced NAD task with tabular data as input. The whole framework, as the adaptive training module shows in Figure 1, consists of two components: TabTransformer for sparse categorical features and GBDT2NN for dense numerical features. We also introduce Focal Loss and adaptive tuning to guide the training phase. We will describe the details of each component in the following subsections.

3.1. Problem Definition. First of all, we only consider the input dataset in tabular format, which is very common in real-world applications at the network edge. Besides, we assume that there are n categories of network traffic, where $n \geq 2$ and at least one category is the normal network traffic class. Then, for the i-th category, we use N_i to denote its sample size. In this paper, we define the dataset as "an imbalanced dataset" when the sample sizes from different classes have wide ranges. In this paper, we use "majority classes" to refer to those with large sample sizes and "minority classes" to refer to other classes with much smaller sizes. Our ultimate goal is to improve the classification accuracy of minority classes without affecting the performance of overall classes.

3.2. TabTransformer for Categorical Features. Motivated by the initial success of Transformer [39] in NLP, Huang et al. adopted the idea to tabular data and proposed TabTransformer [17], which is an architecture that provides and exploits contextual embeddings of categorical features. Their study suggests that, for tabular data, TabTransformer can achieve comparable performance to tree-based ensemble approaches and outperform the state-of-the-art deep learning methods. As shown in Figure 1, we utilize TabTransformer's advantage in handling categorical features, so we only use a part of its original structure. We remove the continuous features in the input, as well as the following normalization layer and concatenation layer related to these features. Figure 2 shows the architecture of TabTransformer used in this paper.

Generally speaking, TabTransformer comprises a column embedding layer, followed by a stack of N transformer layers, and a multilayer perceptron (MLP) before the loss function. Each transformer layer consists of a multihead self-attention layer followed by a position-wise feedforward layer. To learn categorical features more effectively, TabTransformer applies embedding technology on sparse vectors to get low-dimensional dense representation before stepping into transformer layers, denoted as

$$\mathbf{E}_{\phi}(\mathbf{x}_{\text{cat}}) = \left\{\mathbf{e}_{\phi_1}(x_1), \ldots, \mathbf{e}_{\phi_m}(x_m)\right\}, \tag{1}$$

where $\mathbf{x}_{\text{cat}}$ represents all the categorical features with x_i being i-th categorical feature. $\mathbf{e}_{\phi_i}(x_i)$ is corresponding embedding vector for x_i, which can be learned by back-propagation. Based on the above equation, the first transformer layer takes

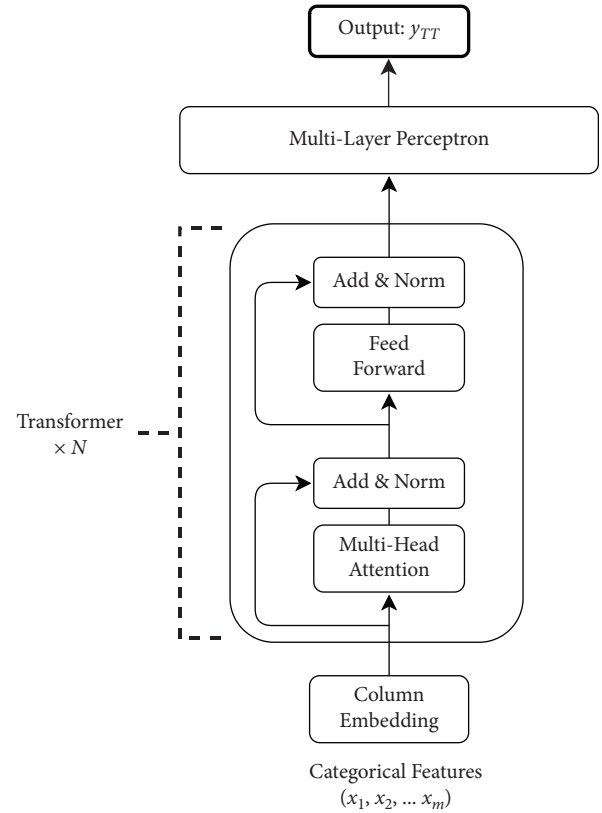

Figure 2: The architecture of TabTransformer.

$\mathbf{E}_{\phi}(\mathbf{x}_{\text{cat}})$ as its input, passes the output to the second transformer layer, and so forth. Unlike the original TabTransformer, we directly pass the output of the stack of transformer layers into an MLP to get the prediction. The prediction $\mathbf{y}_{TT}$ can be formulated as follows:

$$\mathbf{y}_{TT}(\mathbf{x}_{\text{cat}}) = \mathscr{M}\left(f\left(\mathbf{E}_{\phi}(\mathbf{x}_{\text{cat}}); \boldsymbol{\theta}_1\right); \boldsymbol{\theta}_2\right), \tag{2}$$

where function f denotes N transformer layers, $\mathscr{M}$ denotes the MLP, and θ_1 and θ_2 denote the parameters of two components.

3.3. GBDT2NN for Numerical Features. Gradient Boosting Decision Tree (GBDT) [11] is a widely used ensemble model of decision trees. In many application domains, it outperforms other machine learning algorithms such as Random Forest and Support Vector Machine. GBDT, as a tree-based gradient boosting algorithm, can build new trees by computing the information gain and fitting the residuals of previous trees. As mentioned in Section 2, GBDT's strength lies in learning overdense numerical features but it fails to grow trees effectively using sparse categorical features. The path from the root node to the leaf node can build a decision rule, which can act as a vital cross feature. As a result, we choose GBDT to deal with numerical features.

While using GBDT alone is much easier, combining it with NN models is way more challenging. Most of the prior studies try to distill the trees of GBDT into an NN model but only transfer model knowledge in terms of the learned

function without considering other informational knowledges in the tree structure. In [26], Guolin et al. proposed a novel idea to efficiently distill the learned trees, called GBDT2NN, which could perfectly approximate the decision function and tree structure of GBDT with the help of the strong expressiveness ability of NN. For a single tree t, it can be distilled into an NN $\mathcal{N}$ denoted as follows:

$$y^t(\mathbf{x}) = \mathcal{N}\left(\mathbf{x}\left[\mathbb{I}^t\right]; \boldsymbol{\theta}\right) \times \mathbf{q}^t, \tag{3}$$

where $\mathbf{x}[\mathbb{I}^t]$ is the input of $\mathcal{N}$, $\mathbb{I}^t$ represents the used features given from GBDT, and θ is the parameter of $\mathcal{N}$. $\mathbf{q}^t$ denotes the leaf values of tree and $\mathbf{q}_i^t$ is the leaf value of i-th leaf. Since GBDT will get various trees after training, constructing an NN for each tree is very inefficient. In order to improve the efficiency, they proposed Leaf Embedding Distillation and Tree Grouping to downsize the scale of NNs. The Leaf Embedding Distillation adopts embedding technology and converts the one-hot representations of leaf indexes to dense vectors as the targets to be approximated in the learning process. The Tree Grouping divides the trees into k groups, and each group $\mathbb{T}$ has $s = \lceil m/k \rceil$ trees, where there are m trees in total. Finally, the output of GBDT2NN can be denoted as

$$\mathbf{y}_{\mathrm{GBDT2}NN}(\mathbf{x}) = \sum_{j=1}^{k} y_{\mathbb{T}_j}(\mathbf{x}), \tag{4}$$

where $y_{\mathbb{T}_j}(\mathbf{x}) = \mathcal{N}(\mathbf{x}[\mathbb{I}^{\mathbb{T}}]; \theta^{\mathbb{T}})$, which represents the output of $\mathcal{N}_j$ for j-th tree group.

3.4. Combination of TabTransformer and GBDT2NN. As described in previous subsections, we now own the output of TabTransformer and GBDT2NN. So, we are ready to combine them to perform end-to-end training. To get prediction $\hat{\mathbf{y}}$ of the whole model, we assign different trainable weights that can be obtained from back-propagation, that is, w_1 and w_2, for $\mathbf{y}_{TT}$ and $\mathbf{y}_{\mathrm{GBDT2NN}}$ as

$$\hat{\mathbf{y}}(\mathbf{x}) = \sigma\left(w_1 \times \mathbf{y}_{\mathrm{GBDT2NN}}(\mathbf{x}) + w_2 \times \mathbf{y}_{TT}(\mathbf{x})\right), \tag{5}$$

where σ is the activation function for the last layer, for example, softmax for multiclass classification, and the loss value can be expressed as

$$\mathrm{Loss} = \mathscr{L}(\hat{\mathbf{y}}(\mathbf{x}), \mathbf{y}), \tag{6}$$

where $\mathbf{y}$ is the true label of sample $\mathbf{x}$ and $\mathscr{L}$ is the loss function.

3.5. Focal Loss. Focal Loss (FL) [27] is proposed to resolve the class imbalance in object detection tasks, and it is an improvement on the traditional cross-entropy (CE) loss. A proven ability of Focal Loss to solve the imbalance problem in the NAD task has been discussed in [40]. So, we also use FL as the loss function to focus on hard samples while avoiding the bias towards the easy samples. Hard samples are those in the training set which cannot be well predicted, and easy samples are the opposite. According to [27], for a binary classification task, FL is as follows:

$$FL = -\sum_{i=1}^{m} \alpha y_i (1-\hat{y}_i)^{\gamma} \log(\hat{y}_i) + (1-\alpha)(1-y_i)\hat{y}_i^{\gamma} \log(1-\hat{y}_i), \tag{7}$$

where $\hat{y}_i$ represents the probabilistic predictions as defined in Section 3.4, y_i represents the labels of input samples, α is a balanced variant, $\gamma \geq 0$ is called *focusing parameter*, and m is the number of samples. When $\gamma = 0$, it turns into CE loss. For the multiclass classification task, we can use the concept of one-vs-all to extend FL as follows:

$$FL(\hat{y}_i, y) = -(\alpha y + (1-\alpha)(1-y)) \cdot \left(1 - \left(y \cdot \hat{y}_i + (1-y) \cdot (1-\hat{y}_i)\right)\right)^{\gamma} \cdot \left(y \log(\hat{y}_i) + (1-y)\log(1-\hat{y}_i)\right), \tag{8}$$

where we assume that there are m samples and n classes, and then y is the one-hot encoding of the labels, and $\hat{y}_i$ represents the probabilistic predictions with the size of (m, n).

To obtain optimal α and γ, we also deploy adaptive training in the proposed framework by feeding different α and γ from the searching space into Focal Loss. According to [27], we set ranges of α and γ to be (0.25, 0.75) and (0.5, 5), respectively. In the training phase, we only need to choose a target metric, such as $F1$-score, and GTF can automatically search for the best parameters that yield the best performance. Such an adaptive training method enables GTF to be suitable for any scenarios and avoids the need for prior knowledge to set appropriate parameters.

4. Experiment

In this section, we will perform comprehensive evaluations of GTF on two public datasets and compare it with several well-known methods. We will first describe the detailed experimental setup. Then, we will analyze the performance of GTF in both multiclass and binary cases to illustrate its effectiveness.

4.1. Experimental Setup. In this section, we will start with details about datasets and the evaluation criteria. Then, we will brief the comparison methods and ablation study and, finally, our implementation.

4.1.1. Dataset. To illustrate the effectiveness of our proposed method, we conduct experiments on two publicly available intrusion detection datasets, as listed in Table 1.

KDD Cup 1999 dataset [41], also known as KDD'99, is widely used by data mining techniques for the NAD task and includes a wide variety of intrusions simulated in a military network environment. It has been preprocessed into 41

features per network connection and consists of 5 categories of traffic. Besides the Normal traffic, there are 4 types of attacks: DoS (Denial of Service attacks), Probe (Scanning attacks), R2L (Remote to Local attacks), and U2R (User to Root attacks). Among the entire dataset, the majority class is DoS, which occupies 79.2% of the training set and 73.9% of the testing set. On the contrary, U2R only accounts for 0.1% of the training set and 0.2% of the testing set.

UNSW-NB15 dataset [42], published in 2015, is usually used as an alternative to KDD'99. Compared to KDD'99, it can better reflect modern low footprint attack scenarios and thus is more accurate to simulate the real-world traffic. Similar to KDD'99, the UNSW-NB15 dataset has 49 features (7 irrelevant features are removed and the reduced size of feature set is 42) and 9 types of attacks. The types of attacks are Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms.

Imbalance Ratio per Label (IRLbl) is a commonly used indicator to quantify the degree of class imbalance in a dataset. It calculates the ratio of the number of majority class i's samples (N_{majority}) over the number of the class i's samples (N_i) in the multiclass case, as shown as follows:

$$\text{IRLbl}_i = \frac{N_{\text{majority}}}{N_i}. \tag{9}$$

As for the binary case, we simply use *IR* because there only exists one minority class.

Both KDD'99 and UNSW-NB15 datasets are provided in CSV format and have different degrees of imbalanced class distributions. Tables 2 and 3 list the statistics of each dataset.

We consider both imbalanced binary and multiclass classification. In the binary case, we divide UNSW-NB15 into two categories: Normal and Attack. Then, we apply random sampling to the Attack class of the training set to construct datasets with different degrees of imbalance, where we set *IR* to 50, 100, 500, and 1000, respectively.

4.1.2. Implementation. We use scikit-learn (https://scikit-learn.org/stable/index.html), LightGBM (https://lightgbm.readthedocs.io/en/latest/index.html), imbalanced-ensemble (https://github.com/ZhiningLiu1998/self-paced-ensemble), CatBoost (https://catboost.ai/) and PyTorch (https://pytorch.org/) packages to implement these classifiers. We train these models with following parameters:

(i) *Tree-Based Models*. We set the learning rate at 0.01, the number of trees at 128, and the max number of leaves at 10.

(ii) *NN-Based Models*. We use AdamW optimizer with a learning rate of 0.001, a batch size of 1024, and the early stopping rounds of 20.

(iii) *TabTransformer*. We set the embedding dimension at 32, the number of Transformer layers at 6, and the number of attention heads at 8.

(iv) *GBDT2NN*. We decide to use 10 and 20 as fixed values for the number of tree groups and the leaf embedding dimension, respectively.

In order to simulate computationally constrained edge devices, we conduct all experiments on a laptop running Windows 10 with 8 GB RAM and a six-core Intel(R) Core(R) CPU. As in most papers, we perform 10-fold cross-validation for tree-based models and run all NN-based models five times with different random seeds.

4.1.3. Evaluation Criteria. Traditionally, accuracy metrics may have a bias towards the majority class and cannot reasonably reflect the model performance in our scenarios. As a result, we propose using the other evaluation criteria for both overall and individual metrics. All of these metrics are implemented in scikit-learn, a widely used Python library. In order to define our proposed criteria and their equations, let us use TP_i/FP_i to denote true/false positive and TN_i/FN_i to denote true/false negative for a given class i.

(1) Individual Metrics. To evaluate the performance on individual class, we consider Recall, Precision, and $F1$-score as individual metrics. They are defined in the equations below. Based on these equations, we can see that $F1$-score is a weighted average of the Recall and Precision and is usually considered as a trade-off between them.

$$\begin{aligned} \text{Recall}_i\left(R_i\right) &= \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i}, \\ \text{Precision}_i\left(P_i\right) &= \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}, \\ F1\text{-score}_i\left(F1_i\right) &= 2 \cdot \frac{R_i \times P_i}{R_i + P_i}. \end{aligned} \tag{10}$$

(2) Overall Metrics. To evaluate the overall performance in the multiclass case, we choose the Area Under the Receiver Operating Characteristic Curve (ROCAUC) and the Matthews Correlation Coefficient (MCC). ROCAUC shows the insensitivity of class imbalance (when *average* == *'macro'* and *multi_class* == *'ovo'* are set in scikit-learn). MCC is a correlation coefficient, whose value ranges from −1 to 1. A coefficient of 1 means a perfect prediction. Generally speaking, MCC is considered as an unbiased and more comprehensive metric for class-imbalanced tasks. In the binary case, we only use MCC as the overall metric.

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}}. \tag{11}$$

4.1.4. Comparison Methods and Ablation Study. Since our goal is to address the imbalance issue in NAD tasks while learning effectively from tabular input data, we need to evaluate GTF based on these two scenarios. In order to conduct a comprehensive comparison and analysis, we use various models in the evaluation. First, we use LightGBM as our baseline due to its excellent performance and reliability. For NN-based models, we choose the original

Table 1: Details of the datasets used in experiments.

Dataset	Training	Testing	Numerical features	Categorical features
KDD'99	494 021	311 029	34	7
UNSW-NB15	175 341	82 332	37	5

Table 2: Class distribution and IRLbl of KDD'99.

	0	1		2		3		4	
	Num.	Num.	$IRLbl_1$	Num.	$IRLbl_2$	Num.	$IRLbl_3$	Num.	$IRLbl_4$
Training	391 458	97 278	4.0	4107	95.3	1126	347.6	52	7528.0
Testing	229 855	60 593	3.8	4166	55.2	16 345	14.1	70	3283.6

*0: DoS, 1: Normal, 2: Probe, 3: R2L, and 4: U2R.

TabTransformer because it has been proven to be superior to recent deep Neural Networks for tabular data while matching the performance of tree-based ensemble models, like GBDT. For tree-based models, we include CatBoost, which could outperform other GBDT frameworks significantly and can handle categorical features very efficiently. Last but not least, we include two state-of-the-art methods for imbalance classification in NAD tasks. One is an algorithm-level method named Self-Paced-Ensemble (SPE [43]) and the other is a cost-sensitive method named FLAGB [40].

To prove the improvement brought by GTF, we also design two additional ablation experiments. In the first ablation experiment, we only use GBDT2NN without TabTransformer to evaluate the performance of GBDT2NN. As described in the preceding paragraph, TabTransformer has been tested separately, so we do not repeat it. In another ablation experiment, we weaken the GTF by removing Focal Loss to estimate its importance to GTF (represented by GT(F) in subsequent sections).

4.2. Results and Analysis. We first evaluate the performance of GTF in the multiclass case and show the results of both overall and individual comparisons on the two datasets in Tables 4 and 5, respectively. Note that the top-2 results are marked in bold. Due to the space limitation, we only present the most relevant metrics, that is, ROCAUC and MCC, for overall metrics and $F1$ for individual metrics. We also consider the binary case as described in Section 4.1.1 and show the experiment results in Figure 3. Lastly, we describe the results of the ablation study and give a computational complexity analysis of the proposed GTF.

4.2.1. Results on KDD'99. It can be seen that GTF outperforms other methods on both ROCAUC and MCC in the multiclass case, which explicitly indicates the advantage of our approach on imbalanced tabular data. Besides GTF, TabTransformer, GBDT2NN, and FLAGB also demonstrate enhancement on overall metrics. CatBoost and SPE achieve slightly better ROCAUC compared to baseline but worse performance on MCC. Compared to the baseline, GTF improves the ROCAUC (87.71% versus 76.79%) and MCC (82.46% versus 69.59%) simultaneously.

In terms of individual metrics, GTF/GT(F) improves the baseline significantly and beats other methods in three classes (0, 1, and 4). For class 3 and class 4, as shown in Table 2, all of their IRLbls in the training set are relatively large. Especially for class 4, IRLbl = 7528. Thus, the baseline cannot give a reliable prediction on minority classes and clearly demonstrates how the classifier's performance is negatively affected by the increased IRLbl. For instance, the $F1$ of baseline declines to 0 in class 4. The performances of TabTransformer and CatBoost also suffer from the large IRLbl. In class 3 and class 4, they both achieve 0 $F1$-score. Interestingly, as far as class 3 and class 4 are concerned, SPE shows the opposite result compared to FLAGB and GTF. It performs best in class 3 but worst in class 4, while the other two perform better in class 4, mainly because of the ability of Focal Loss to focus on minority classes. Compared with FLAGB, GTF has an obvious improvement on $F1$-score of all classes and boosts $F1_2/F1_3/F1_4$ by 2 to 3 times.

4.2.2. Results on UNSW-NB15. For the UNSW-NB15 dataset, although the numbers from GTF are not as eye-catching as those on KDD'99, it is still compelling enough as the metrics are either the best one or close to the best one. Regarding overall metrics, GTF achieves the best result on ROCAUC, which slightly improves the baseline (92.02% versus 91.11%), and the GTF's result on MCC is slightly worse than the baseline (69.87% versus 70.27%). Other methods, such as TabTransformer, do not yield any significant improvement on ROCAUC and achieve similar or worse performance on MCC. Such a situation also reflects on individual metrics. Overall, GTF is the most suitable method because it can boost ROCAUC the most without decreasing MCC. As far as individual metrics are concerned, GTF either performs better than the baseline or achieves very similar numbers. Its improvements on $F1_2$, $F1_8$, and $F1_9$ are maximum among all methods, which is the only one that can boost $F1$ on class 8 and class 9. Among all classes, GTF and SPE significantly outperform other methods in class 2, class 3, and class 5. These classes are relatively common in the training set, and their IRLbels are relatively low (all less than 50). It is counterintuitive to see the numbers, but it further confirms the supremacy of GTF.

Table 3: Class distribution and IRLbl of UNSW-NB15.

	0	1		2		3		4		5		6		7		8		9	
	Num.	Num.	$IRLbl_1$	Num.	$IRLbl_2$	Num.	$IRLbl_3$	Num.	$IRLbl_4$	Num.	$IRLbl_5$	Num.	$IRLbl_6$	Num.	$IRLbl_7$	Num.	$IRLbl_8$	Num.	$IRLbl_9$
Training	56 000	10 491	5.3	1746	32.1	12 264	4.6	33 393	1.7	2000	28.0	18 184	3.1	130	430.8	1133	49.4	40 000	1.4
Testing	37 000	3496	10.6	583	63.5	4089	9.0	11 132	3.3	677	54.7	6062	6.1	44	840.9	378	97.9	18 871	2.0

*0: Normal, 1: Reconnaissance, 2: Backdoor, 3: DoS, 4: Exploits, 5: Analysis, 6: Fuzzers, 7: Worms, 8: Shellcode, and 9: Generic.

Table 4: Comparison of metrics obtained by different methods for KDD'99. The results are expressed in %, and $F1_i$ means $F1$-score in class i.

Model	Overall		Individual				
	ROCAUC	MCC	$F1_0$	$F1_1$	$F1_2$	$F1_3$	$F1_4$
Baseline	76.79	69.59	97.65	74.86	21.34	2.03	0
GTF	**87.71**	**82.46**	**98.44**	84.19	**77.47**	**11.11**	**45.07**
GT(F)	**86.59**	**82.80**	**98.54**	**84.53**	76.18	9.51	20.37
TabTransformer	79.58	82.33	98.39	**84.45**	**81.31**	0	0
GBDT2NN	80.19	82.04	98.41	83.70	76.53	9.60	20.90
CatBoost	82.23	61.86	91.29	71.84	76.50	0	0
FLAGB	85.31	69.98	95.96	78.64	39.87	3.40	**21.13**
SPE	81.04	64.92	97.70	66.67	19.34	**14.22**	1.36

Bold values represent top-2 results.

Table 5: Comparison of metrics obtained by different methods for UNSW-NB15. The results are expressed in %, and $F1_i$ means $F1$-score in class i.

Model	Overall		Individual									
	ROCAUC	MCC	$F1_0$	$F1_1$	$F1_2$	$F1_3$	$F1_4$	$F1_5$	$F1_6$	$F1_7$	$F1_8$	$F1_9$
Baseline	91.11	**70.27**	84.83	**84.95**	5.78	8.79	69.19	0	**40.33**	32.78	**41.18**	**98.18**
GTF	**92.02**	69.87	82.44	84.13	**11.57**	**24.91**	**69.42**	3.14	38.37	**36.36**	**47.78**	**98.35**
GT(F)	90.34	66.42	82.12	82.81	8.62	23.32	68.07	2.18	37.13	28.87	39.31	**98.18**
TabTransformer	84.59	69.59	**84.92**	84.13	0	**26.35**	**69.97**	0	38.61	0	0	98.14
GBDT2NN	70.36	55.42	77.87	25.93	3.25	22.30	58.38	**5.50**	26.47	0	0.67	92.63
CatBoost	**91.22**	70.13	**85.16**	84.02	0.34	1.11	68.28	0	**40.39**	0	31.19	98.05
FLAGB	91.04	**70.17**	84.87	**85.01**	5.24	8.61	68.93	0	**40.14**	**43.75**	39.44	98.15
SPE	90.95	56.55	73.76	61.46	**10.66**	24.00	60.79	**8.84**	33.31	7.07	11.57	97.80

Bold values represent top-2 results.

Based on the results, we can conclude that GTF can provide the best performance in the multiclass case with the comprehensive consideration of overall and individual metrics. It demonstrates that the proposed GTF can not only learn efficiently from tabular data but also mitigate the imbalance classification problem.

4.2.3. Overall Metric in the Binary Case among Different IRs. Due to the limited space, we only show the overall metric of the binary case, that is, MCC, in Figure 3. The results indicate that the classifiers' performance generally shows a downward trend as the IR increases, and GTF outperforms other methods under all IRs. The performance of TabTransformer is the worst as it cannot distinguish Attack records from the Normal ones at all. The performance of CatBoost is very unstable, which is also lower than the baseline in most cases. FLAGB achieves good performance when IR = 50, but its performance drops dramatically when IR ≥ 100. Both GTF and SPE outperform others regardless of the value of IR. From the figure, we also observe that GTF's performance is more stable (approximately within 20% range) for different IRs, compared to SPE. It is worth mentioning that SPE consumes about twice as much training time as GTF does in our experiments. Therefore, we can conclude that GTF can perform much better than other methods in the binary case while being fast enough.

4.2.4. Ablation Study. As shown in Tables 4 and 5 and Figure 3, neither TabTransformer nor GBDT2NN can achieve the best performance across all cases. What is worse is that, in some cases, these two models perform way worse than the baseline in terms of overall and individual metrics. On the contrary, in both multiclass and binary cases, GT(F) can beat TabTransformer and GBDT2NN on most metrics. For example, in Table 5, GT(F) can achieve good performance on $F1_7$ and $F1_8$, while both TabTransformer and GBDT2NN yield 0. Even though when GT(F) leads to worse performance compared to TabTransformer or GBDT2NN, such as $F1_4$ and $F1_5$ in Table 5, the achieved performance tends to be close to the best ones or above averages. This indicates that the combination of TabTransformer and GBDT2NN can improve the overall performance and prevent performance degradation in several cases.

Now we want to understand the impact of Focal Loss on the classification model by comparing GTF and GT(F). It is obvious to see that GTF outperforms GT(F) on almost all metrics, especially on the metrics to detect minority classes (e.g., $F1_3$ in Table 4 and $F1_7$ and $F1_8$ in Table 5). In some cases, such as $F1_4$ in Table 4, the Focal Loss leads to more than 100% improvement of the score. Another two interesting data points are $F1_2$ and $F1_3$ in Table 4. In these two cases, GT(F) is clearly underperforming compared to TabTransformer and GBDT2NN. But there is a big performance boost when the Focal Loss is added in GTF. For the overall

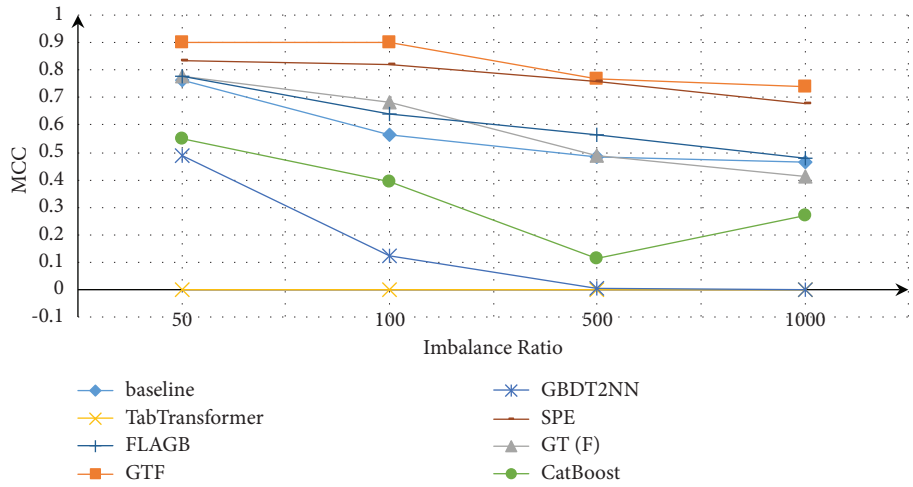

Figure 3: Overall metric in the binary case among different IRs.

Table 6: Computational complexity of GTF.

# of parameters	Batch size	MACs (M)	Inference time (milliseconds)
355,731	1	0.36	12.32
	1024	364.27	119.72

metrics in the binary case, we also observe more than 30% performance improvement on MCC when *IR* is 1000, as shown in Figure 3.

To summarize, all our results illustrate the effectiveness and performance improvement brought by GTF, which is a combination of TabTransformer, GBDT2NN, and Focal Loss to focus on minority classes.

4.3. Computational Complexity. Because edge devices usually have limited computation resources and memory, we also evaluate the complexity of GTF using (1) the number of multiply-accumulate operations (MACs, also known as MADDs) performed per inference, (2) the number of parameters, and (3) the inference time. We consider two batch sizes: 1 and 1024. The obtained results are displayed in Table 6. From these numbers, we can see that the number of parameters is less than one million, which is much less than some complex deep learning models. The MACs grow linearly with the batch size, but the inference time grows at a much slower rate. For instance, we notice that increasing the batch by 1000 times (from 1 to 1024) will only incur about a ten-time rise for inference time. When the batch size is 1 (single sample), the inference time can be as short as 12.32 ms. Thus, we can conclude that GTF is feasible to be deployed on constrained edge devices.

5. Conclusion

In this paper, we described the challenges of tabular input and class imbalance which exist in the nature of the NAD task. Based on the analysis of data characteristics, we propose a new method named GTF that combines the advanced cost-sensitive algorithm and tabular learning strategy. Specifically, our proposal utilizes TabTransformer and GBDT2NN to handle categorical and numerical features, respectively. It also applies Focal Loss in the learning process to reduce the bias towards the majority classes. Powered by these components, GTF could gain powerful learning capability on tabular data while maintaining the ability to handle imbalance classification tasks. Compared to existing well-known models, our comprehensive experiments demonstrate that GTF can learn more effectively with tabular data and adapt to different imbalanced datasets in both multiclass and binary cases. Moreover, our implementation also shows that GTF is effective enough to deploy on constrained edge devices for NAD purposes.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (no. XDC02030000), Jiangsu Planned Projects for Post-doctoral Research Funds (2021K402C), and Jiangsu Provincial Double-Innovation Doctor Program (JSSCBS20211035).

References

[1] A. R. Javed, M. Usman, S. U. Rehman, M. U. Khan, and M. Sayad Haghighi, "Anomaly detection in automated vehicles using multistage attention-based convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4291–4300, 2021.

[2] A. Mohiyuddin, A. R. Javed, C. Chakraborty, M. Rizwan, M. Shabbir, and J. Nebhen, *Secure Cloud Storage for Medical Iot Data Using Adaptive Neuro-Fuzzy Inference System*, Springer, New Yok, NY, USA, 2021.

[3] H. Gao, Xi Qin, R. J. D. Barroso, W. Hussian, Y. Xu, and Y. Yin, "Collaborative learning-based industrial iot api recommendation for software-defined devices: the implicit knowledge discovery perspective," *IEEE Transactions on Emerging Topics in Computational Intelligence*, pp. 1–11, 2020.

[4] Y. Huang, H. Xu, H. Gao, X. Ma, and W. Hussian, "Ssur: an approach to optimizing virtual machine allocation strategy based on user requirements for cloud data center," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 2, pp. 670–681, 2021.

[5] Federal Bureau of Investigation (Fbi), *Internet Crime Report 2020*, 2021, https://www.ic3.gov/Media/PDF/AnnualReport/2020_IC3Report.pdf.

[6] Y. Zhu, W. Zhang, Y. Chen, and H. Gao, "A novel approach to workload prediction using attention-based lstm encoder-decoder network in cloud environment," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, pp. 1–18, 2019.

[7] Y. Yin, Z. Cao, Y. Xu, H. Gao, R. Li, and Z. Mai, "Qos prediction for service recommendation with features learning in mobile edge computing environment," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1136–1145, 2020.

[8] H. Gao, K. Xu, M. Cao, J. Xiao, Q. Xu, and Y. Yin, "The deep features and attention mechanism-based method to dish healthcare under social iot systems: an empirical study with a hand-deep local-global net," *IEEE Transactions on Computational Social Systems*, pp. 1–12, 2021.

[9] J. Xiao, H. Xu, H. Gao, M. Bian, and Y. Li, "A weakly supervised semantic segmentation network by aggregating seed cues: the multi-object proposal generation perspective," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 17, no. 1s, pp. 1–19, 2021.

[10] Y. Xu, Y. Wu, H. Gao, S. Song, Y. Yin, and X. Xiao, "Collaborative apis recommendation for artificial intelligence of things with information fusion," *Future Generation Computer Systems*, vol. 125, pp. 471–479, 2021.

[11] H. F. Jerome, "Greedy function approximation: a gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, 2001.

[12] A. Kadra, M. Lindauer, F. Hutter, and J. Grabocka, "Regularization is all you need: simple neural nets can excel on tabular data," 2021, https://arxiv.org/abs/2106.11189.

[13] L. Katzir, E. Gal, and R. El-Yaniv, "Net-dnf: effective deep modeling of tabular data," in *Proceedings of the International Conference on Learning Representations*, Addis Ababa, Ethopia, May 2020.

[14] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," *Advances in Neural Information Processing Systems*, vol. 31, pp. 6638–6648, 2018.

[15] S. Ö. Arik and T. Pfister, "Tabnet: attentive interpretable tabular learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 6679–6687, Vancouver, Canada, February 2021.

[16] S. Popov, S. Morozov, and A. Babenko, "Neural oblivious decision ensembles for deep learning on tabular data," in *Proceedings of the International Conference on Learning Representations*, Jakarta, Indonesia, September 2019.

[17] X. Huang, A. Khetan, M. Cvitkovic, and Z. S. Karnin, "Tabtransformer: tabular data modeling using contextual embeddings," 2020, https://arxiv.org/abs/2012.06678.

[18] V. Sugumaran, V. Muralidharan, and K. I. Ramachandran, "Feature selection using decision tree and classification through proximal support vector machine for fault diagnostics of roller bearing," *Mechanical Systems and Signal Processing*, vol. 21, no. 2, pp. 930–942, 2007.

[19] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems?" *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3133–3181, 2014.

[20] T. Chen and C. Guestrin, "XGBoost," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 2016.

[21] S. S. Meriem Amina, B. Abdolkhalegh, N. Kim, and C. Mohamed, "Featuring real-time imbalanced network traffic classification," in *Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, IEEE, Halifax, Nova Scotia, Canada., july 2018.

[22] H. Guo, Y. Li, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data: review of methods and applications," *Expert Systems with Applications*, vol. 73, pp. 220–239, 2017.

[23] A. Dal Pozzolo, G. Boracchi, C. Olivier, C. Alippi, and G. Bontempi, "Credit card fraud detection: a realistic modeling and a novel learning strategy," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 8, pp. 3784–3797, 2018.

[24] D. Gamberger, N. Lavrac, and C. Groselj, "Experiments with noise filtering in a medical domain," *ICML*, vol. 99, pp. 143–151, 1999.

[25] H. He and Y. Ma, *Imbalanced Learning: Foundations, Algorithms, and Applications*, Wiley-IEEE Press, Hoboken, NJ, USA, 1st edition, 2013.

[26] K. Guolin, Z. Xu, J. Zhang, B. Jiang, and T.-Y. Liu, "DeepGBM," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ACM, Anchorage, AK, USA, July 2019.

[27] T.-Yi Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, October 2017.

[28] K. Guolin, M. Qi, and T. Finley, "Lightgbm: a highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems*, vol. 30, pp. 3146–3154, 2017.

[29] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: gradient boosting with categorical features support," 2018, https://arxiv.org/abs/1810.11363.

[30] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, Springer series in statistics, New York, NY, USA, 2001.

[31] Sergey Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the International conference on machine learning*, pp. 448–456, PMLR, Lille, France, July 2015.

[32] I. Mani and I. Zhang, "Knn approach to unbalanced data distributions: a case study involving information extraction," in *Proceedings of the workshop on learning from imbalanced datasets*, vol. 126, ICML United States, Washingdon, DC, USA, August 2003.

[33] I. Tomek, "Two modifications of CNN," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 11, pp. 769–772, 1976.

[34] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[35] H. He, B. Yang, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, IEEE, Hong Kong, China, June 2008.

[36] S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining*, March 2009.

[37] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: improving prediction of the minority class in boosting," in *Knowledge Discovery in Databases: PKDD 2003*vol. 107–119, Berlin, Germany, Springer, 2003.

[38] Xu-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory under-sampling for class-imbalance learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 539–550, 2009.

[39] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Advances in Neural Information Processing Systems*, pp. 5998–6008, Springer, New York, NY, USA, 2017.

[40] Yu Guo, Z. Li, Z. Li, G. Xiong, M. Jiang, and G. Guo, "FLAGB: Focal loss based adaptive gradient boosting for imbalanced traffic classification," in *Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN)*, July 2020.

[41] KDD Cup 1999 Data, 2021, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[42] N. Moustafa and Jill Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS)*, November 2015.

[43] Z. Liu, W. Cao, Z. Gao et al., "Self-paced ensemble for highly imbalanced massive data classification," in *Proceedings of the 2020 IEEE 36th International Conference on Data Engineering (ICDE)*, April 2020.