IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 8 January 2025, accepted 30 January 2025, date of publication 3 February 2025, date of current version 11 February 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3538170

RESEARCH ARTICLE

Class Imbalance in Network Traffic Classification: An Adaptive Weight Ensemble-of-Ensemble Learning Method

MAHMOUD ABBASI[1], (Member, IEEE), SEBASTIÁN LÓPEZ FLÓREZ[1], AMIN SHAHRAKI[2], (Senior Member, IEEE), AMIR TAHERKORDI[2], (Senior Member, IEEE), JAVIER PRIETO[1], (Senior Member, IEEE), AND JUAN M. CORCHADO[1,3]

[1]BISITE Research Group, University of Salamanca, 37007 Salamanca, Spain
[2]Department of Informatics, University of Oslo, 0373 Oslo, Norway
[3]Department of Electronics, Information and Communication, Faculty of Engineering, Osaka Institute of Technology, Osaka 535-8585, Japan

Corresponding author: Mahmoud Abbasi (mahmoudabbasi@usal.es)

This work was supported by the IoTalentum Project within the Framework of Marie Skłodowska-Curie Actions Innovative Training Networks (ITN)-European Training Networks (ETN), which was funded by European Union Horizon 2020 Research and Innovation Program under Grant 953442.

ABSTRACT Network Traffic Classification (NTC) serves as a crucial element in network management, and the rapid progress in machine learning has inspired the utilization of learning methods to discern network traffic. The inherent characteristics of network traffics result in uneven class distributions when datasets are shaped, creating a phenomenon known as class imbalance. This phenomenon has garnered growing attention across various research fields. Despite encountering performance setbacks attributed to class imbalance, this challenge remains inadequately examined in the realm of network traffic classification. This paper introduces the Adaptive Weight Ensemble-of-Ensemble Learning (AWEE) method as an innovative solution to this challenge. The AWEE integrates multiple ensemble layers with a dynamic weight adjustment mechanism, showcasing the collaborative intelligence of diverse modeling strategies. Using a sliding window-based validation approach, the model enhances adaptability and robustness to address concept drift in dynamic data streams. Experimental studies on benchmark datasets demonstrate the superior performance of AWEE (achieving an outstanding accuracy rate of over 98%), highlighting its effectiveness in handling class imbalance challenges across diverse network traffic scenarios. AWEE, outperforms competitive methods, including algorithmic-level, cost-sensitive, and data-level techniques, showcasing its robustness and superior performance in addressing class imbalance challenges across a wide range of network traffic scenarios.

INDEX TERMS Applications of artificial intelligence, class imbalance, ensemble learning, implemented artificial intelligence, network traffic classification.

I. INTRODUCTION

In an era characterized by the exponential growth of digital interconnectedness, the sheer volume of network traffic has reached unprecedented levels. The classification of network traffic is a key element in the protection and optimization of digital communication [1]. Traffic classification is the fundamental process of categorizing data packets or flows in a network according to specific attributes or behavior. It serves as the digital sentry, tirelessly monitoring the vast data streams that flow through the information superhighways of the modern world. Performing a proper classification of network traffic is not a mere academic exercise; It is the front line of defense against cyber threats, the backbone of network performance optimization, and the compass that guides data-driven decision-making [2].

Considering the importance and impact of traffic classification in different fields, it is evident that network

The associate editor coordinating the review of this manuscript and approving it for publication was Cong Pu.

administrators rely on this classification to ensure that data packets are routed efficiently and securely. Security experts depend on it to detect and mitigate emerging threats, identify vulnerabilities, and respond to incidents in real time [3]. Moreover, data scientists harness the insights gleaned from network traffic classification to extract valuable intelligence, support predictive modeling, and enhance the overall quality of service [4]. Despite these recognized benefits, network traffic classification is far from straightforward. The world of network traffic is not a balanced, evenly distributed ecosystem. Instead, it is riddled with disparities and imbalances that challenge the effectiveness of classification algorithms [5]—a challenge commonly known as *class imbalance*.

Class imbalance [6] refers to the often-unavoidable skew in the distribution of different classes within network traffic data. This phenomenon poses significant challenges to network traffic classifiers [7]. For instance, certain classes, such as normal or benign traffic, frequently dominate the majority of data samples, while others, such as malicious traffic or rare application types, occupy only a fraction of the data landscape [8]. When machine learning algorithms are trained on such imbalanced data, they often become biased toward the majority class, leading to suboptimal accuracy and increased vulnerability to threats from the minority class [9], [10]. The inability to accurately classify malicious traffic, which often represents the minority class, can result in severe security risks.

To address these challenges, various strategies have been proposed. Some methods rely on over- or under-sampling techniques to rebalance the class distribution [11], [12], but these approaches can introduce bias or disrupt the underlying data distribution, potentially producing skewed results. Other techniques employ specialized machine learning algorithms or cost-sensitive learning approaches to assign higher importance to minority classes [13], [14]. While these methods have shown promise in specific scenarios, they often lack the adaptability to accommodate the dynamic and evolving nature of network traffic patterns.

The need for a robust and adaptive approach to handle class imbalance is paramount. In this paper, we propose a novel algorithmic-level method, the Adaptive Weight Ensemble-of-Ensemble Learning (AWEE) Method, to address class imbalance in network traffic classification. This innovative method leverages the power of ensemble learning to create a dynamic and adaptive model, capable of addressing both class imbalance and concept drift in the ever-evolving landscape of network traffic. The AWEE approach integrates multiple ensemble layers with a dynamic weight adjustment mechanism to adaptively respond to changes in data distribution.

A. CONTRIBUTIONS AND SCOPE OF THE STUDY

This paper makes the following contributions:

- **Novel ensemble learning approach:** We propose the AWEE method, which dynamically adjusts the weights of ensemble models to address class imbalance and adapt to evolving network traffic patterns.
- **Enhanced adaptability and robustness:** Using a sliding window-based validation technique, the AWEE model demonstrates robustness in handling concept drift and dynamic data streams.
- **Extensive experimental validation:** The effectiveness of the AWEE method is validated using three benchmark datasets (Cambridge, UNSW-2018 IoT Botnet, and CSE-CIC-IDS2018), achieving outstanding accuracy of over 98% across diverse network traffic scenarios.
- **Generalizable framework for imbalanced datasets:** The proposed methodology introduces a generalized framework that can be applied to other domains facing similar challenges, such as fraud detection and medical diagnostics.

The scope of this study is as follows:

- **Problem focus:** The study focuses on addressing the issue of class imbalance in network traffic classification using a novel ensemble-based approach.
- **Datasets:** The research employs three publicly available benchmark datasets (Cambridge, UNSW-2018 IoT Botnet, and CSE-CIC-IDS2018) to evaluate the performance and robustness of the proposed method.
- **Application domain:** While developed in the context of network traffic classification, the principles of the AWEE method are extendable to other domains with class imbalance challenges.

The proposed approach not only enhances accuracy but also provides a framework for addressing dynamic and evolving network environments. This paper is structured as follows: Section II provides background on traffic classification and ensemble learning. Section III presents the literature review, highlighting existing methods and their limitations. Section IV describes the proposed AWEE methodology and its implementation. Section V details the experimental setup, datasets, and results analysis. Finally, Section VI concludes the paper and discusses future directions.

II. BACKGROUND

This section provides an overview of traffic classification and ensemble learning, the foundational concepts for this study.

A. TRAFFIC CLASSIFICATION

Traffic classification is essential to optimize network management, improve security, and maintain quality of service (QoS). It enables efficient resource allocation, real-time threat detection, and enforcement of network policies across various domains, such as enterprise networks, service providers, and government systems [1], [15], [16].

Port-based classification is one of the simplest methods to categorize traffic based on port numbers. Although it was effective in earlier networking environments, it struggles to accommodate modern applications that often share ports or use dynamic port allocation, diminishing its

reliability [17]. Deep Packet Inspection (DPI) provides more precise traffic identification by examining packet content. However, this method faces increasing challenges due to the prevalence of encrypted traffic, which obscures packet contents. Furthermore, DPI raises privacy concerns, as it involves inspecting the content of communications [18]. Flow-based classification takes a different approach by analyzing flow patterns, such as IP addresses, ports, and protocols, to infer high-level traffic behaviors. This technique is computationally efficient and effective for recognizing broad traffic categories, making it a popular choice for many network management scenarios [19]. Finally, machine learning-based classification uses algorithms to adaptively identify complex and evolving traffic patterns. By training on diverse datasets, these models can recognize emerging traffic behaviors, making them particularly suitable for dynamic and rapidly changing network environments [20].

Despite advancements, traffic classification faces challenges such as class imbalance, encryption, rapid protocol evolution, and traffic variability [21], [22]. Addressing these issues is critical for maintaining accuracy and adaptability.

B. ENSEMBLE LEARNING

Ensemble learning enhances predictive performance by combining multiple models, leveraging the ''wisdom of crowds'' to reduce bias and variance [3]. In network traffic classification, it addresses class imbalance and improves model robustness [23].

Two key ensemble learning methods are particularly notable: Bagging (Bootstrap Aggregating) is designed to reduce variance by training models on bootstrapped subsets of the data and aggregating their predictions. This approach is exemplified by algorithms like Random Forest, which combines the outputs of multiple decision trees to improve accuracy and stability [24]. Boosting, on the other hand, reduces both bias and variance by iteratively focusing on misclassified samples. It constructs a sequence of weak learners, each improving upon the errors of its predecessor. Popular techniques in this category include AdaBoost and Gradient Boosting, which are widely used for their effectiveness in handling complex datasets [25].

Ensemble methods are highly versatile, capable of integrating diverse classifiers to tackle the challenges of imbalanced network traffic data. By mitigating overfitting and enhancing generalization, they are particularly well-suited to addressing the complexities of real-world applications. In our proposed approach, ensemble methods form the backbone of the solution, dynamically combining the strengths of multiple models to effectively manage class imbalance and improve overall performance.

III. LITERATURE REVIEW

Class imbalance in network traffic classification is a significant challenge, as machine learning algorithms often favor majority classes, leading to the misclassification of minority classes [26], [27]. This bias is exacerbated by the sheer volume of benign traffic, which overshadows critical but infrequent classes like malicious traffic [28]. Misclassification of malicious traffic can result in severe security risks, including data breaches and diminished trust in network systems [8].

A. ALGORITHMIC-LEVEL TECHNIQUES

Algorithmic-level techniques address class imbalance by modifying the learning process to emphasize minority classes. These include ensemble learning and hybrid methods, which combine multiple models for better robustness and generalization.

Several studies highlight the effectiveness of ensemble methods. Thakkar and Lohiya [29] proposed a bagging-based approach using Deep Neural Networks (DNNs) as base estimators, incorporating class weights to create balanced subsets. Bi et al. [7] introduced SVUX, which combines Sparse Autoencoders and unbalanced XGBoost optimized with evolutionary algorithms to enhance minority class detection. Jiang et al. [30] developed a dynamic ensemble with LightGBM, incorporating Borderline-SMOTE for synthetic sample generation and dynamic weighting to improve anomaly detection in IoT data streams.

Hybrid methods have also been explored to mitigate the challenges of imbalance. Ren et al. [31] proposed DUEN, integrating dynamic undersampling with Boosting to emphasize boundary samples, while Hartono and Syah [32] combined membership-based and density-based oversampling for improved minority class representation in noisy datasets. These methods demonstrate notable success, but often involve significant computational overhead.

B. COST-SENSITIVE TECHNIQUES

Cost-sensitive techniques address class imbalance by assigning different penalties for misclassifications based on class importance. This approach is especially valuable in contexts like network security, where misclassifying minority classes such as malicious traffic can lead to severe consequences [33].

Telikani et al. [14] developed a cost-sensitive machine learning model integrating multitask learning and stacked autoencoders for feature extraction. By incorporating a cost-sensitive hinge loss function, the model prioritized low-frequency intrusions, significantly improving detection accuracy. Similarly, Telikani et al. [13] extended this work by introducing a Cost-sensitive Stacked Autoencoder (CSSAE) that dynamically adjusts costs during the training process, improving performance on unbalanced datasets. Building on these ideas, they later proposed a cost-sensitive deep learning approach that integrates a dynamically generated cost matrix into the cross-entropy loss function, further refining detection rates for minority classes [13].

Hu et al. [34] proposed GAT-COBO, a cost-sensitive graph neural network that combines Graph Attention

Networks (GAT) with a boosting framework. This method dynamically adjusted misclassification penalties for minority classes, addressing the over-smoothing problem in GNNs and improving fraud detection in highly imbalanced datasets. Gupta et al. [35] introduced a layered architecture, CSE-IDS, combining cost-sensitive deep learning and ensemble techniques to detect anomalies in network traffic. This approach reduced false alarms while achieving high detection accuracy for minority classes.

These methods illustrate the adaptability of cost-sensitive techniques in prioritizing critical minority classes in network traffic classification, although they often require fine-tuning of cost matrices for optimal performance.

C. DATA-LEVEL TECHNIQUES

Data-level techniques directly manipulate datasets to balance class distributions through methods such as oversampling, undersampling, or hybrid approaches [36]. By creating more equitable datasets, these techniques provide a level playing field for machine learning models.

Oversampling methods, such as GMF-SMOTE [37], enhance the traditional SMOTE approach by using Gaussian Mixture models to filter noisy and boundary samples, improving the quality of synthesized minority samples. Yang et al. [38] introduced SPE-ACGAN, which combines auxiliary classifier GANs for oversampling and Self-Paced Ensemble (SPE) for undersampling. This method demonstrated significant improvements in the detection of unbalanced network traffic in datasets such as CICIDS-2017 and CICIDS-2018. Hybrid approaches such as RFSE-GRU [39] integrate SMOTE-based oversampling with Edited Nearest Neighbor (ENN) undersampling, achieving superior accuracy and balance for encrypted traffic classification in large datasets.

Undersampling techniques, while reducing the size of the majority class, often risk losing valuable data. Techniques such as Random Sampling and Cluster Centroids [40] selectively remove redundant majority samples to maintain dataset integrity while enhancing minority class representation. Silva et al. [41] evaluated the performance of several undersampling strategies and found that distance-based methods, such as Cluster Centroids, effectively addressed class imbalance in intrusion detection systems.

Hybrid methods that combine oversampling and undersampling are particularly effective. For example, Qin et al. [42] proposed SD Sampling, which focuses on difficult-to-classify samples while mitigating SMOTE's tendency to overgeneralize. This approach, integrated with hierarchical ensemble models like XGBoost and Random Forest, demonstrated significant improvements in classification accuracy and generalization.

These data-level techniques are invaluable in preprocessing imbalanced datasets, allowing a more accurate and reliable classification. However, they may introduce additional computational overhead, especially in large-scale datasets.

D. SUMMARY

While algorithmic, cost-sensitive, and data-level techniques have shown promise in addressing class imbalance, they are often associated with specific challenges. Algorithmic-level techniques, such as ensemble learning approaches, improve model robustness but often come with increased computational complexity. Cost-sensitive methods are effective in prioritizing minority classes, but require careful fine-tuning of cost matrices, which may not adapt well to dynamic traffic patterns. Data-level techniques, including oversampling and undersampling, balance datasets effectively but can introduce noise or remove valuable information, potentially leading to suboptimal performance.

The need for robust and adaptive solutions is evident, particularly in scenarios where network traffic patterns are dynamic and evolving. The proposed AWEE method addresses these limitations by leveraging ensemble learning with a dynamic weight adjustment mechanism, enabling the model to adapt to both class imbalance and concept drift in network traffic data.

To explicitly differentiate our work from previous studies, a detailed comparison table (Table 1) has been provided. This table highlights the key theoretical and technical contributions of previous studies, their methodologies, and the challenges they address, contrasted with the innovations introduced by the AWEE method. The table provides a clear depiction of how the AWEE method overcomes limitations such as lack of adaptability, computational inefficiency, and suboptimal handling of class imbalance in previous works.

IV. PROPOSED METHODOLOGY

A. OVERVIEW OF THE AWEE METHODOLOGY

This study presents an innovative method, called AWEE, to address the challenge of class imbalance in network traffic classification. Figure 1 illustrates the innovative process of the AWEE method, in which the model undergoes continuous training with dynamic weight adjustments based on real-time validation. Unlike traditional approaches, this flowchart emphasizes the adaptive nature of our model, which incorporates multiple ensemble layers that are continuously updated to optimize performance against class imbalance and concept drift in network traffic data. This approach is a significant departure from conventional static training models, providing a demonstration of the novel contributions of our method. The method consists of five phases. In the *first* step, that is, pre-processing, an appropriate input for the machine learning models (base classifiers) is provided by removing irrelevant information, data normalization, and feature selection. Then, the pre-processed data are partitioned into three distinct sub-datasets: one for training, another for validation, and a third for testing. In the *second* step, known as sub-ensemble development, individual base classifiers

TABLE 1. Comparison of key aspects of existing techniques with AWEE.

Methodology	Key contributions	Challenges addressed
Algorithmic-level techniques (e.g., [30], [31])	Enhanced robustness using ensemble models; dynamic weighting in some methods.	Computational complexity; limited adaptability to evolving traffic patterns.
Cost-sensitive techniques (e.g., [14], [35])	Prioritization of minority classes through cost matrices; improved minority detection rates.	Requires fine-tuned cost matrices; limited adaptability to concept drift.
Data-level techniques (e.g., [38], [41])	Improved dataset balance using oversampling and undersampling; hybrid approaches for noise reduction.	Risk of overfitting or removing valuable information; computational overhead in large datasets.
AWEE method	Dynamic weight adjustment; ensemble-of-ensemble learning for enhanced adaptability and performance.	Simultaneously addresses class imbalance and concept drift; efficient handling of evolving network traffic patterns.

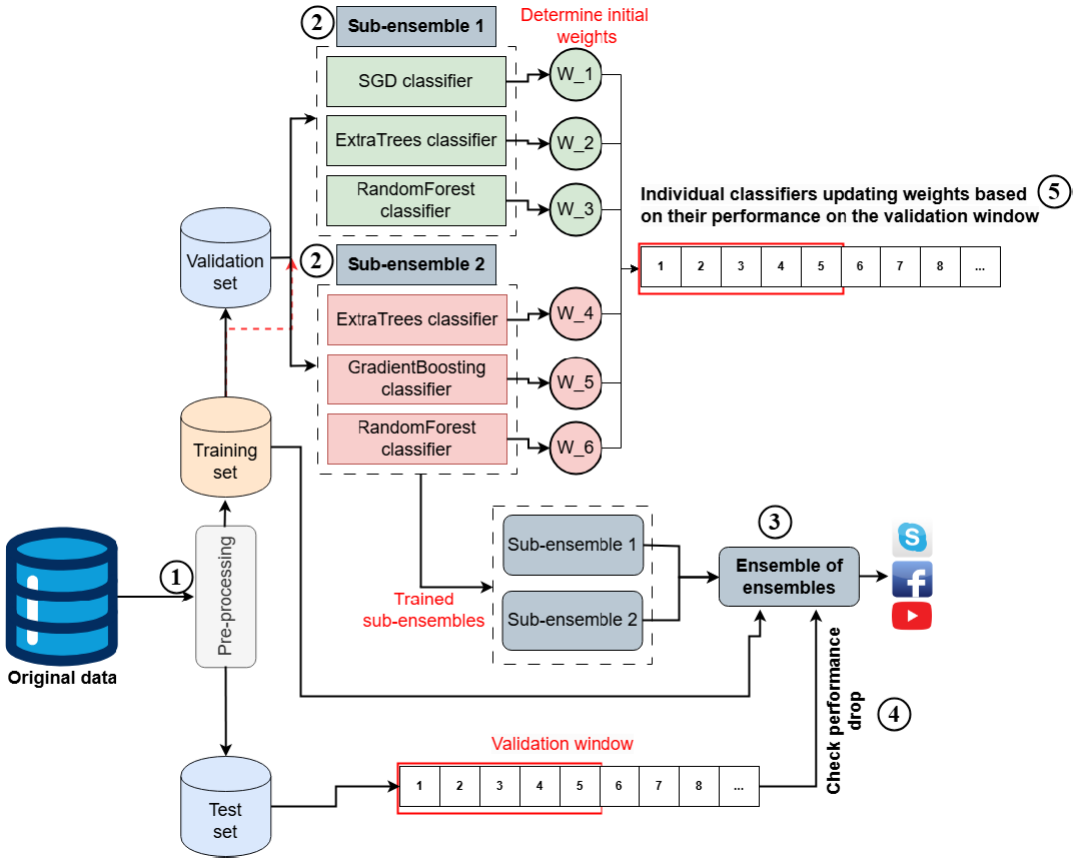


FIGURE 1. Framework of our proposed ensemble-of-ensembles architecture.

are combined to create two distinct sub-ensemble models. In the *third* stage, referred to as ensemble-of-ensemble creation, an ensemble of ensemble models is constructed by aggregating predictions from the sub-ensemble models. In the *forth* step, i.e., performance evaluation, the performance of the ensemble-of-ensemble model is assessed to detect any potential drop in performance compared to previous episodes. Finally, in the *fifth* step, called 'update weights,' the performance of individual classifiers is taken into account for updating their respective weights. In the following, we meticulously explore each step.

Algorithm 1 provides a detailed pseudo-code of the proposed AWEE method, outlining the sequential steps involved in implementing the model. This algorithm encapsulates the entire process, from data preprocessing and sub-ensemble development to the dynamic updating of classifier weights, offering a clear and structured representation of the methodology. In the following sections, we meticulously explore each

Algorithm 1 Adaptive Weight Ensemble-of-Ensemble Learning (AWEE)

Input: Training data D, Validation data V, Initial weights W, Number of ensembles N, Learning rate lr, Discount factor df
Output: Trained ensemble model
Step 1: Data Preprocessing
Clean and normalize the data D
Perform feature selection on D
Step 2: Sub-ensemble Development
for each ensemble i in N **do**
    Train base classifiers on D
    Combine base classifiers into sub-ensembles
end for
Step 3: Ensemble-of-Ensemble Creation
Aggregate predictions from sub-ensembles to create the final ensemble model
Step 4: Performance Evaluation
Evaluate the performance of the ensemble model on V
if Performance drop detected **then**
    Update weights W based on validation accuracy
end if
Step 5: Weights Update
for each classifier j in ensemble **do**
    Update weight W_j using $W_j = W_j + lr \times (Pa - Ca) \times df$
    Normalize weights W
end for
Return the final trained model

step to further elaborate on the underlying mechanisms and the rationale behind the design choices.

B. KEY PHASES OF AWEE

In this subsection, we describe the five key phases of the AWEE methodology, each of which may consist of multiple steps. These phases are data preprocessing, sub-ensemble development, ensemble-of-ensembles creation, performance drop check, and individual weight updating, which together form the foundation of the proposed approach.

1) PHASE 1: DATA PREPARATION AND PRE-PROCESSING

In this study, three imbalanced network traffic datasets, i.e., Cambridge [43], UNSW-2018 IoT Botnet [44], and CSE-CIC-IDS2018 [45], are used to train and evaluate the developed model for imbalance class network traffic classification. We selected these three datasets because they are publicly available datasets and have a substantial history of being widely used as benchmarks in the field of network traffic monitoring and analysis. Table 2 presents the statistics for each of these datasets. The Cambridge dataset consists of 352,059 samples with nine categories of traffic, including "ATTACK," "DATABASE," "FTP-CONTROL," "FTP-DATA," "FTP-PASV," "MAIL," "P2P," "SERVICES," and "WWW." The UNSW-2018 IoT Botnet dataset

TABLE 2. Statistics of the datasets used in this study.

Dataset	Class type	The number of samples	Ratio
Cambridge	ATTACK	1671	0.0047
	DATABASE	2410	0.0068
	FTP-CONTROL	2905	0.0082
	FTP-DATA	4478	0.0127
	FTP-PASV	2645	0.0075
	MAIL	24421	0.0788
	P2P	1755	0.0056
	SERVICES	1893	0.0053
	WWW	309881	0.8801
	Total	**352059**	
UNSW-2018 IoT Botnet	RT	44009	0.024
	DoS	801747	0.449
	DDoS	936264	0.525
	Total	1782020	
CIC-IDS2018	BENIGN	1743179	0.759
	DoS Hulk	231073	0.100
	Infiltration	158930	0.069
	DDoS	128027	0.055
	DoS GoldenEye	10293	0.004
	FTP-Patator	7938	0.003
	SSH-Patator	5897	0.0025
	DoS slowloris	5796	0.0025
	DoS Slowhttptest	5499	0.0023
	Total	2296632	

comprises 1.7 million traffic records. While certain common attack classes such as "DoS," "DDoS," and "Reconnaissance Attack (RT)" are well-represented with a substantial number of instances, there are also rare attack types like "Theft" that have significantly fewer samples. Furthermore, the CSE-CIC-IDS2018 dataset consists of 10 sub-datasets, from which we have specifically chosen 120,000 instances to create a balanced dataset. The new dataset includes five different types of traffic which are: "BENIGN", "DoS GoldenEye", "DoS Slowloris", "FTP-BruteForce", and "SSH-Bruteforce". In general, the Cambridge dataset exhibits a more pronounced data imbalance, with an imbalance ratio of approximately 153.[1] In this context, the CSE-CIC-IDS2018 dataset shows an imbalance ratio of approximately 153, while this figure is 21.2 for the UNSW-2018 IoT Botnet dataset.

In order to conduct our analysis and train our machine learning models, we first need to load the dataset into memory and preprocess it to ensure its suitability for further tasks. Tables 2 and 3 provide information on the size of datasets and selected features, respectively. One of the foundations of our method is effective data pre-processing. To ensure the integrity and reliability of our data, we embark on an extensive data cleaning process. Raw network traffic datasets, originating from diverse sources, often contain inconsistencies, outliers, and incomplete records [46]. Our data cleaning procedures involve identifying and rectifying these issues, thus ensuring the highest data quality for our classification task.

[1]The imbalance ratio is computed as follows: it is the result of dividing the number of samples in the class with the most instances by the number of samples in the class with the fewest instances [9].

TABLE 3. Features used in this study.

Dataset	Selected features
Cambridge	Server Port, Client Port, med data wire, max IAT, max data wire, min data ip, max data ip, mean data control, q3 data control, max data control, var data control, urgent_data_pkts_a b, mss_requested_a b, mss_requested_b a, max_segm_size_b a, avg_segm_size_b a, zero_win_adv_a b, zero_win_adv_b a, avg_win_adv_a b, FFT_all, FFT_all,FFT_all,Effective_Bandwidth, min_data_control_b a, q1_data_control_b a, var_data_ip_b a, min_data_ip_b a, med_data_wire_b a, mean_data_control_a b, q1_data_control_a b, mean_data_wire_a b
UNSW-2018 IoT Botnet	pkSeqID, seq, stddev, N IN Conn P SrcIP, state number, mean, N IN Conn P DstIP, srate, max
CIC-IDS2018	Flow Bwd Pkt Len Min, Flow Pkts/s, Bwd Pkts/s, Pkt Len Min, URG Flag Cnt, Init Fwd Win Byts, Fwd Act Data Pkts, Fwd Seg Size Min

The handling of missing values is a critical aspect of our data cleaning process. Network traffic data frequently features missing or erroneous entries, potentially leading to noise and inaccuracies in the analysis. We adopt a comprehensive strategy to address this challenge, systematically identifying and handling missing values. Any missing values are replaced with appropriate defaults or imputed using established statistical strategies (such as mean, median, or most frequent values) to maintain the overall dataset's integrity. Moreover, the data-preprocessing step involves data normalization using the StandardScaler to ensure uniform feature scaling. Normalization is particularly crucial in the context of machine learning training when the features of a dataset exhibit varying scales. Furthermore, feature selection based on Pearson's correlation coefficient test is performed to extract a subset of relevant features, a key strategy to streamline the classification process and reduce computational complexity (see Table 3).

2) PHASE 2: SUB-ENSEMBLE DEVELOPMENT

In this subsection, we introduce the concept of *sub-ensembles,* a fundamental element of our proposed approach. Sub-ensembles are designed to enhance the predictive capabilities and robustness of our ensemble model by harnessing a diverse array of base classifiers. This subsection outlines the structure, training process, and integration of sub-ensembles within the overarching ensemble framework. Sub-ensembles, as the name suggests, are groups of classifiers that collectively contribute to the final prediction. The sub-ensembles are constructed with the deliberate selection of diverse base classifiers, each chosen for its unique characteristics and strengths. The decision to split the base classifiers into two individual ensembles, rather than incorporating all six base classifiers into a single-layer ensemble, is driven by the need to optimize diversity and robustness of the model. By organizing the base learners into two distinct ensembles, each ensemble can focus on different aspects of the classification task, allowing for specialized learning that captures a broader range of data characteristics. This division helps to reduce both bias and variance in the final model, as each ensemble can adapt to specific challenges presented by the data, such as different patterns of class imbalance. Our methodology features two primary sub-ensembles. Using two subsets provides a balance between diversity and complexity, ensuring that the ensemble benefits from different modeling approaches while avoiding the increased complexity and potential overfitting that could arise from using more than two subsets. Moreover, initial experiments with different numbers of subsets (e.g., three) informed our decision to use two, based on the observed trade-offs in performance and computational efficiency. The first sub-ensemble, denoted as *ensemble_models_1*, consists of a carefully curated selection of base classifiers, including:

- Stochastic Gradient Descent (SGD): The SGD classifier stands out as a versatile and efficient linear model, specifically tailored for the demands of large-scale machine learning tasks characterized by extensive and dynamically evolving datasets. Its adaptability makes it a robust choice, particularly well-suited for scenarios where data patterns shift over time. One of its key strengths lies in its application of the log loss function, a mechanism that proves particularly effective in addressing binary classification challenges. This function enables the model to make incremental adjustments during training, ensuring its responsiveness to the nuanced changes in the dataset [47]. In other words, in the context of our specific problem of imbalanced class classification, the selection of SGD classifier is strategic. Imbalanced datasets often pose challenges where the distribution of classes is skewed, leading to potential bias in model training. SGD, with its adaptability and efficiency, offers advantages in such scenarios. The incremental adjustments facilitated by its log loss function are particularly beneficial for addressing imbalanced classes. By continuously refining its understanding through iterations, SGD can focus on the minority class, making nuanced adjustments that enhance its ability to discern patterns within the imbalanced data landscape.
- Decision trees: Incorporation of decision trees into our ensemble model (*ensemble_models_1*) serves as a pivotal choice. Decision trees are recognized for their inherent capability to discern and represent intricate relationships within the data, making them well-suited for capturing the complexity often associated with imbalanced datasets. In our ensemble configuration, the decision tree classifier is deliberately fine-tuned with a maximum depth set at 20 and a minimum number of samples required to split a node set to two. This

meticulous configuration acts as a strategic measure to strike a delicate balance. The imposition of a maximum depth mitigates the risk of overfitting, preventing the model from becoming overly complex and overly tailored to the training data, thereby promoting better generalization to unseen instances. Simultaneously, by setting constraints on node splitting, we empower the model to craft nuanced decision boundaries without succumbing to the pitfalls of excessive intricacy. This tailored calibration aligns seamlessly with our overarching goal of constructing a robust ensemble model capable of effectively addressing the challenges posed by class imbalance in the domain of network traffic classification.

- Random forest: As a robust solution within our ensemble methodology, the random forest classifier emerges as a collective force, uniting a total of 100 decision trees. This ensemble strategy proves powerful in aggregating predictions from individual trees, mitigating the risk of overfitting, and amplifying the model's ability to generalize effectively. By combining the outputs of multiple decision trees, the random forest inherently introduces diversity into the learning process, ensuring a well-rounded model. This diversity not only bolsters the model's resistance to overfitting but also contributes to its enhanced adaptability in capturing intricate patterns within the data. As an integral component of *ensemble_models_1*, the random forest classifier plays a pivotal role in fortifying the overall robustness of our approach, a crucial aspect in addressing the inherent challenges associated with class imbalance in network traffic classification.

The second sub-ensemble, referred to as *ensemble_models_2*, is designed to complement the first sub-ensemble and diversify the base classifiers further:

- Extra trees: In the intricate landscape of network traffic classification, the inclusion of the extra trees classifier as a fundamental component of *ensemble_models_2* demonstrates a thoughtful consideration of the challenges posed by class imbalance. Functioning as an ensemble of exceptionally randomized decision trees, the extra trees classifier, with its 100 estimators, introduces a remarkable level of variability into the learning process. This deliberate injection of randomness during tree construction sets it apart from traditional decision trees and enriches the diversity within *ensemble_models_2*. The abundance of trees in the ensemble, combined with this extra layer of randomness, empowers the sub-ensemble with a heightened capacity to effectively capture a broad spectrum of data patterns. This unique characteristic becomes instrumental in enhancing the adaptability and comprehensiveness of *ensemble_models_2*, addressing the intricate challenges associated with class imbalance in network traffic classification. By embracing this distinctive approach, the extra trees classifier serves as a valuable asset in our ensemble methodology, strategically contributing to the robustness required for navigating the complexities of imbalanced class scenarios in network traffic classification.
- Random Forest: Similar to *ensemble_models_1*, the random forest classifier is included in *ensemble_models_2* to promote synergy and cross-sub-ensemble comparisons. With 100 decision trees, it continues to play a vital role in improving the overall predictive performance of the methodology.
- Gradient Boosting: As a pivotal component within *ensemble_models_2*, the gradient boosting classifier stands out as a boosting ensemble method renowned for its proficiency in amalgamating the strengths of multiple models. Operating across 100 boosting stages, its primary focus lies in rectifying errors made by preceding stages, thereby incrementally refining the overall model's performance. Distinguished by a sequential learning approach and a boosting mechanism, gradient boosting adds significant value to *ensemble_models_2*. Its iterative strategy, where each stage corrects the deficiencies of its predecessors, contributes to the continual improvement of the model's predictive capabilities. This adaptive learning process, coupled with the boosting mechanism, positions gradient boosting as a valuable asset in our ensemble methodology, effectively addressing the intricate challenges associated with class imbalance in network traffic classification.

Note that these classification algorithms were not chosen arbitrarily; their selection was informed by a combination of literature review and preliminary experiments. For instance, we conducted initial tests comparing the performance of various classifiers on imbalanced network traffic datasets. The results showed that ensemble methods, particularly Random Forests and Gradient Boosting, consistently outperformed individual classifiers in terms of both accuracy and stability. This empirical evidence strongly supports our decision to employ an ensemble-of-ensembles approach, as it allows for the aggregation of strengths across different classifiers, enhancing overall model performance.

a: STEP 1 OF PHASE 2: WEIGHTS INITIALIZATION FOR SUB-ENSEMBLES

The initialization of classifier weights is a foundational step in building and fine-tuning the sub-ensembles, ensuring balanced contributions from all classifiers at the start of the training process. This step is pivotal to the functionality and success of AWEE, as it establishes a baseline for dynamic weight adjustments in later phases. In other words, the training process initiates the learning of each base classifier within the sub-ensembles. These classifiers are the fundamental units responsible for making individual predictions. The choice of classifiers within each sub-ensemble is deliberate and strategic, designed to harness a diverse set of modeling techniques and strengths. These classifiers vary in their

ability to capture different aspects of the data and adapt to different data dynamics. For instance, SGD classifier excels in handling large-scale, dynamically changing data through incremental learning. On the other hand, decision trees and random forests are adept at capturing complex, non-linear relationships within the data. The gradient boosting classifier specializes in boosting the performance of existing models by correcting their errors.

A portion of the training data is set aside for validation. In this case, 30% of the data is allocated for validation. For each classifier in *ensemble_models_2* and *ensemble_models_1*, the training process begins. After each classifier has trained, their predictions in the validation set are made and the validation accuracy of these predictions is computed using the accuracy score. To determine the initial weights for each classifier, the validation scores are taken into account, as shown in Fig.1. In other words, The idea is to assign higher weights to classifiers that perform better in the validation set. The validation scores (validation accuracy) for the classifiers are normalized. This normalization is performed by dividing each score by the sum of all scores in the validation scores (see Eq. 1). The resulting values represent the proportion of each classifier's contribution to the sub-ensemble. Note that this weight assignment process ensures that no single model dominates the ensemble, promoting balance and preventing the ensemble from over-relying on any single classifier.

$$\mathrm{N\_w}_i = \frac{Val\_score_i}{\sum_{j=1}^{n} Val\_score_j} \tag{1}$$

where $\mathrm{N\_w}_i$ refers to the normalized weight for classifier i, Val_score_i is the validation score for classifier i, and $\sum_{j=1}^{n} Val\_score_j$ represents the sum of validation scores for all classifiers in the given sub-ensemble.

b: STEP 2 OF PHASE 2: SUB-ENSEMBLES CREATION

The creation of sub-ensembles involves combining the predictions of individual classifiers, which have been trained and initialized with weights, to make collective decisions. In this process, the predictive capabilities of individual classifiers are harmoniously integrated. This sub-ensemble creation functions as a bridge between the individually trained classifiers and the ensemble-of-ensembles.

The *VotingClassifier* from the scikit-learn library is used for sub-ensemble formation. Here, the ensemble strategy is set to 'soft' voting. The 'soft' voting mechanism considers not only the decisions of classifiers but also the associated probabilities of predictions. This ensures that the confidence levels of the classifiers are appropriately weighted in the decision-making process. In other words, within each sub-ensemble, the weighted voting mechanism comes into play when making predictions. When a prediction is required, each base classifier within the sub-ensemble provides its prediction along with its confidence (as reflected by its weight). These predictions are combined and the final decision is determined by weighting the average of these predictions. By combining predictions from multiple models through a weighted voting system, the sub-ensemble can collectively make more accurate predictions for the minority class. Moreover, the sub-ensembles, by aggregating predictions from multiple models, are less susceptible to the impact of outliers and noisy instances. Outliers may have a stronger influence on a single model, but their impact tends to be diluted when combined with predictions from other models in an ensemble. In addition, if one model exhibits a bias towards the majority class, other models may compensate by providing more accurate predictions for the minority class. *Ensemble_models_1* and *ensemble_models_2* operate in parallel, leveraging the diverse modeling approaches captured by their constituent classifiers. They take into account the weighted input from each classifier to arrive at a collective decision.

It is noteworthy that the final predictions of each sub-ensemble serve as inputs to the ensemble-of-ensembles (see Fig. 1, trained sub-ensembles). This overarching ensemble takes into consideration the collective wisdom of the sub-ensembles, possibly alongside additional ensembles, to provide a comprehensive and robust prediction for the dataset.

3) PHASE 3: ENSEMBLE-OF-ENSEMBLES

The ensemble-of-ensembles is architected to assimilate the collective insights of the sub-ensembles. Each sub-ensemble contributes its weighted decisions, integrating the knowledge distilled from individual base classifiers. The ultimate aim is to harness the synergy of these sub-ensembles to make informed predictions. The core decision-making mechanism in the ensemble-of-ensembles is 'hard' voting. This method aggregates the predictions from the sub-ensembles, with each sub-ensemble represented as an entity within the ensemble structure. When a prediction is required, each sub-ensemble provides its decision, which is predominantly based on the combined wisdom of its constituent classifiers. The 'hard' voting mechanism mandates a majority rule, wherein the most frequently predicted class is chosen as the final prediction. This straightforward yet effective approach leverages the strength of diverse sub-ensembles to make a robust and reliable decision. The Eq. 2 explains the 'hard' voting mechanism.

The ensemble-of-ensembles exemplifies a collaborative approach to decision-making. It builds upon the complementary nature of sub-ensembles. While *ensemble_models_1* and *ensemble_models_2* are designed to capture distinct aspects of the data, the ensemble-of-ensembles aggregates these insights, creating a harmonious fusion of modeling strategies. This collaborative intelligence is a testament to the adaptability and robustness of our methodology. It allows the ensemble of ensembles to excel in scenarios where individual sub-ensembles may exhibit varying degrees of accuracy.

$$FinalDecision = \arg\max_{class} \sum_{sub-ensemble} \mathrm{W} \times \mathrm{Prd}_p \tag{2}$$

where *FinalDecision* represents the ultimate output of ensemble-of-ensembles, $\arg\max_{class}$ denotes the class for which the sum is highest, $\sum_{\text{sub-ensemble}}$ is summation over all sub-ensembles, W represents the weight assigned to each sub-ensemble, and Prd_p is the probability assigned to a particular class.

4) PHASE 4: CHECK THE PERFORMANCE DROP

The robustness and adaptability of our methodology is further enhanced through a dynamic training process within the ensemble-of-ensembles. This phase introduces a training loop (using the training data), which continually refines the predictive capabilities of the ensemble by leveraging new data and adjusting the sub-ensembles' weights. The training loop operates over a predefined number of episodes. Each episode represents a distinct cycle of training in which the ensemble adapts to the dynamic of evolving data. This iterative approach ensures that the ensemble remains up-to-date and capable of accommodating potential concept drift in the dataset. Concept drift, a common occurrence in dynamic data streams, refers to the phenomenon where the statistical properties of the data change over time. These changes can result from various factors, such as evolving user behavior, environmental shifts, or technical alterations. In the context of our methodology, concept drift could manifest itself as shifts in the underlying data patterns, which can potentially affect the performance of the ensemble.

The training process incorporates a sliding window-based validation approach. In each episode, the validation window is updated to capture a fresh subset of the dataset. The size of this window is determined by *window_size*, and it ensures that the ensemble's performance is continually evaluated on a representative portion of the data. This window-based adaptive validation is instrumental in detecting and responding to changes in data patterns. In fact, the training process is tuned to the presence of drift in the concept, where data patterns change over time. If a significant performance drop is detected during training, the methodology can respond effectively. This detection mechanism is activated when the difference in validation accuracy between episodes exceeds a predefined threshold, termed *drift_threshold*. This threshold is a user-defined parameter that quantifies the allowable difference in validation accuracy between consecutive episodes during the training process. When the difference in validation accuracy exceeds this threshold, it is indicative of a significant performance drop, signaling a change in the data distribution. In such cases, the training process can be stopped or adapted to address evolving data patterns.

5) PHASE 5: INDIVIDUAL CLASSIFIERS UPDATING WEIGHTS

The dynamic adaptation of weights assigned to individual classifiers within the sub-ensembles is an important aspect of the AWEE. This process, executed within each training episode, plays a vital role in fine-tuning the ensemble's decision-making capabilities. The adjustment of weights is fundamentally performance-driven. It recognizes that not all classifiers contribute equally to the ensemble's accuracy. Some classifiers may excel in certain scenarios, while others may lag behind. The goal of weight adjustment is to reflect the relative contributions of individual classifiers based on their performance in the validation window. The process begins by evaluating the performance of each individual classifier in the validation window. This evaluation is based on their ability to make accurate predictions. The accuracy of each classifier is assessed by comparing its predictions with the ground truth labels. The accuracy score, computed using the accuracy score function, quantifies the classifier's success in correctly classifying instances within the validation window.

Weight adjustment is executed by updating the weights assigned to each classifier within the sub-ensemble. This update is proportional to the classifier's individual accuracy. The weights are modified using the following formula:

$$\mathrm{U}_w = \mathrm{O}_w + \mathrm{lr} * (\mathrm{P}_a - \mathrm{C}_a) * \mathrm{D}_f \tag{3}$$

where U_w is updated weight, O_w denotes old weight, lr is learning rate, P_a denotes previous accuracy, and C_a and D_f are current accuracy and discount factor, respectively. The learning rate(lr) is a user-defined parameter that controls the rate at which weights are adjusted. It influences the sensitivity of the weight update to performance changes. In other words, the learning rate governs the speed at which weight updates occur. Specifically, it dictates how much of the weight adjustment formula's computed change should be applied in each update. A higher learning rate results in more substantial weight adjustments with each iteration, while a lower learning rate leads to more gradual adjustments.

On the other hand, the discount factor (D_f) is a parameter integral to the weight adjustment process within our methodology. It controls the magnitude of weight updates for individual classifiers. This parameter introduces a level of adaptability to the weight adjustment process and its influence is significant in maintaining the robustness and stability of the ensemble. The discount factor functions as a modulator for weight updates. It determines how much importance should be placed on the difference between the previous accuracy of a classifier and its current accuracy. In essence, the discount factor gauges the impact of performance fluctuations on weight adjustments. A higher discount factor implies a more pronounced response to performance changes, whereas a lower discount factor results in more conservative updates.

To ensure that the weights remain valid and proportional, a normalization step follows the weight adjustment process. The normalization scales the weights so that they sum to 1, representing a distribution of influence. This normalization prevents any single classifier from dominating the decision-making process and fosters a balanced contribution from all classifiers within the sub-ensemble.

The weight adjustment mechanism acknowledges the evolving nature of data patterns and the changing performance of individual classifiers. It allows the ensemble to

adapt to shifting dynamics and maximize the impact of high-performing classifiers while mitigating the influence of underperforming ones. This adaptability ensures that the ensemble remains resilient and responsive to the complexities of real-world data.

V. EXPERIMENTAL RESULTS

In this section, we assess the effectiveness of the proposed ensemble method through a comprehensive set of experiments conducted on three distinct imbalance datasets. Subsequently, we provide a thorough discussion of the results obtained.

A. EVALUATION METRICS

To evaluate the effectiveness of the proposed method in handling imbalanced datasets, we utilize a range of performance metrics. These metrics encompass accuracy, F1-score, precision, recall, and G-Mean. Indeed, due to the potential for accuracy to provide misleading insights in imbalanced datasets, various alternative metrics are employed to assess the performance of the AWEE in such scenarios. To compute these metrics, we rely on four crucial terms: TP (True Positive), FP (False Positive), TN (True Negative), and FN (False Negative). We assume that the reader is already familiar with these four terms.

Accuracy represents the fraction of accurately classified cases (TP and TN) relative to the total number of cases.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{4}$$

F1-score is the harmonic mean of precision and recall. When its value is high and approaches that of accuracy, it signifies a superior classification performance.

$$F\text{-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

where:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7}$$

Finally, the G-Mean, or geometric mean, is a performance metric used in classification tasks, particularly in situations where there is a class imbalance. The G-Mean is designed to give a balanced measure of a model's performance across both the majority and minority classes. It combines the TP rate (sensitivity or recall) and the TN rate (specificity) into a single metric. The formula for the G-Mean is:

$$\text{G-Mean} = \sqrt{\text{Sensitivity} \times \text{Specificity}} \tag{8}$$

B. EXPERIMENT SETUP AND HYPERPARAMETERS SELECTION

In this section, we detail the experimental setup for our method. Given the inherent class imbalance in network traffic data, we addressed this challenge employing our proposed AWEE method, which adaptively assigns weights to different components of the ensemble to mitigate the effect of imbalance. The ensemble-of-ensembles approach is the main focus of our investigation, which comprises multiple ensemble layers designed to collectively improve classification accuracy. To ensure the replicability of our results, all experiments were repeated multiple times, and we reported the average performance metrics. The experiments were conducted using four core AMD Ryzen 7 5800H CPU@3.20GHz processor, and 16.00 GB of RAM. The experiments were carried out using the Python programming language, using libraries such as scikit-learn and XGBoost.

The hyperparameter selection process, often referred to as parameter tuning, is a pivotal stage in building machine learning-based network traffic classifiers and significantly influences the classification results. In Table 4, a comprehensive list of hyperparameter descriptions and their corresponding values is provided for each model, encompassing all hyperparameters used in our experiments. The selection of hyperparameters for the AWEE method was performed using a trial-and-error approach, where different combinations of hyperparameter values were iteratively tested to identify the configuration that yielded the best performance on our validation data. The test set was strictly reserved for final model evaluation and was not used during the hyperparameter tuning process, ensuring that the model's performance metrics are unbiased and reflect its generalization capability.

Although more systematic approaches like grid search or random search are often employed, the trial-and-error method was chosen due to specific constraints of our experimental setup and the inherent complexity of the model. These constraints included the computational resources available and the high dimensionality of the parameter space, which made exhaustive search methods impractical. Despite these challenges, the selected hyperparameters were rigorously validated, and the final model was tested only after the tuning phase, demonstrating its robustness across various network traffic scenarios.

TABLE 4. Hyperparameters values of the different classifiers.

Model	Hyperparameters	Values
SGD	loss	'log_loss'
	max_iter	100
Decision tree	max_depth	20
	min_samples_split	2
Random forest	n_estimator	100
	max_depth	20
Extra trees	max_n_estimator	100
Gradient boosting	max_n_estimator	100
All models	window_size	200
	drift_threshold	0.7
	learning_rate	0.1
	discount_factor	0.6

C. RESULTS AND ANALYSIS

In this section, we delve into a comprehensive examination of the performance of the AWEE method in the context

of our study. This section aims to provide insights into the effectiveness of the AWEE for the task at hand, along with comparisons against baseline methods and competing approaches. Morespecifically, in this study, we selected a range of general machine learning techniques as baseline methods for comparison. The rationale behind this choice is rooted in the broad applicability and established performance of these techniques across various domains, including NTC. While these methods are not exclusively designed for NTC, they are widely used as benchmarks in the machine learning community. This allows us to establish a solid baseline and assess the relative performance of our proposed AWEE method. The choice to use these conventional methods was also influenced by their ability to handle class imbalance. Although not network traffic-specific, MOTEs, although not network traffic-specific, are recognized for their effectiveness in rebalancing datasets and have been widely adopted in numerous studies dealing with imbalanced data. Through these analyzes, our objective is to offer a comprehensive understanding of the performance of the AWEE method, its comparative effectiveness against baseline methods, and its competitive position within the broader landscape of existing approaches.

It is worth noting that while the primary focus of our study is improving classification accuracy in the presence of class imbalance, it is essential to acknowledge the computational cost associated with the Given that AWEE incorporates multiple layers of ensemble learning and dynamic weight adjustments, there is an inherently higher computational demand compared to more straightforward models. In other words, the increased computational complexity arises from several factors, including the need to train and validate multiple base classifiers within each sub-ensemble, the iterative nature of the weight adjustment process, and the ensemble-of-ensembles architecture itself. These components collectively contribute to a more resource-intensive training process, which can result in longer training times and higher computational requirements. However, it is crucial to note that the computational overhead is an expected trade-off for achieving the enhanced performance and robustness demonstrated by the AWEE method. The computational cost is justified by the significant improvements in accuracy, precision, recall, and overall model stability, particularly in handling highly imbalanced datasets. Moreover, the design of AWEE, with its adaptive learning mechanism, ensures that the method remains efficient in online or partial online scenarios, where real-time updates and adaptability are critical.

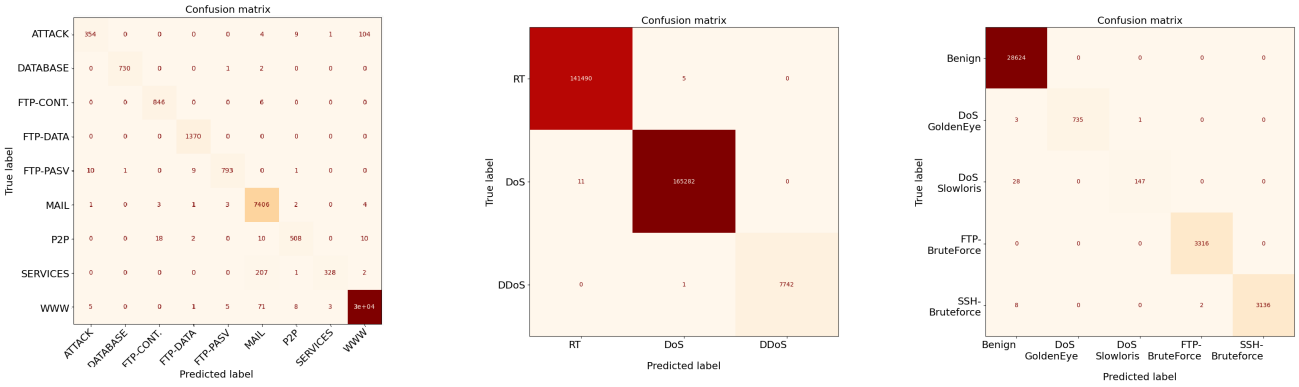
1) PERFORMANCE ANALYSIS OF THE AWEE

This subsection presents an accuracy comparison of our proposed method, demonstrating its performance across three distinct datasets. Table 5 displays the classification performance of the proposed ensemble-of-ensemble approach as evaluated in the test data. The proposed AWEE method demonstrates strong performance in the Cambridge dataset. The achieved accuracy of 98.9% with a minimal variation of ± 0.0032 indicates a high level of correctness in classifying instances. Precision and recall, both reported at 98.9% with small variations (± 0.0031 and ± 0.0032, respectively), highlight the model's ability to maintain a balance between correctly identifying positive instances and minimizing false positives. The F1-score, at 98.8% with a small variation of ± 0.0032, reinforces the model's effectiveness in handling imbalanced datasets. Moreover, the AWEE method exhibits outstanding performance in the UNSW-2018 dataset, showcasing a remarkable accuracy of 99.9% with an extremely small variation. Precision, recall, and F1-score, all reported at 99.9% with the same minimal variation, emphasize the model's consistent and exceptional classification capabilities. Finally, on the CSE-CIC-IDS2018 dataset, the AWEE method performs impressively, achieving an accuracy of 98.9% with a small variation of ± 0.0032. The precision and recall values, both reported at 98.9%, highlight the balanced approach of the model in correctly classifying positive instances while minimizing false positives. Although the precision variation is slightly larger at ± 0.0034, the overall performance suggests that the AWEE method effectively handles the characteristics of the CSE-CIC-IDS2018 dataset.

In addition, Figure 2 illustrates the confusion matrices of our AWEE method for imbalance traffic classification in the benchmark datasets, respectively. In Figure 2, the class imbalance is evident in the datasets. For example, consider the UNSW-2018 dataset, in which the third class (class ''DDoS''), has fewer instances compared to classes ''DoS'' and ''RT''. For the other datasets, similar patterns are observed in terms of class imbalance, reflecting challenges similar to those seen in the UNSW-2018 dataset. In each case, the minority class tends to have fewer instances compared to the majority classes, potentially impacting the model's ability to generalize effectively to the less-represented class. The results for the imbalance classification demonstrate the model's effectiveness in identifying instances of the minority class, with a reasonable number of false positives and false negatives. In the UNSW-2018 dataset, for example, the minority class exhibits 5 false positives and 5 false negatives. However, in the CSE-CIC-IDS2018 dataset, the model has some difficulty in correctly classifying instances of the minority class, resulting in a moderate number of false negatives. Regarding the Cambridge dataset, the diagonal elements represent the true positives for each class, indicating that the model is generally effective in classifying instances across all classes, including minority classes. More specifically, the model exhibits a reasonable performance on minority classes, as indicated by the high true positive values for class ''ATTACK'', class ''DATABASE'', and class ''SERVICES''. The model's ability to correctly classify instances in these minority classes suggests that it is not biased toward the majority classes and can generalize well across different classes.

TABLE 5. Performance of the AWEE on the benchmark datasets.

Dataset	Accuracy	Precision	Recall	F1-score	G-Mean
Cambridge	98.9 ±0.0032	98.9 ±0.0031	98.9 ±0.0032	98.8 ±0.0032	0.98
UNSW-2018	99.9 ±3.6e-6	99.9 ±3.6e-6	99.9 ±3.6e-6	99.9 ±3.6e-6	0.99
CSE-CIC-IDS	98.9 ±0.0032	98.9 ±0.0034	98.9 ±0.0019	98.8 ±0.0032	0.99



(a) Cambridge (b) UNSW-2018 (c) CSE-CIC-IDS2018

FIGURE 2. Confusion matrix of the AWEE for traffic classification on the benchmark datasets.

TABLE 6. Comparison of the AWEE on the baseline methods on the Cambridge dataset.

Method	Accuracy	Precision	Recall	F1-score	G-Mean
AWEE	98.9 ±0.0032	98.9 ±0.0031	98.9 ±0.0032	98.8 ±0.0032	0.98
Decision trees	94.4 ±0.051	61.4 ±0.418	61.4 ±0.418	63.4 ±0.387	0.72
Naive Bayes	84.7 ±0.044	85.4 ±0.042	84.9 ±0.047	83.3 ±0.03	0.85
Gradient Boosting	37.5 ±0.36	84.4 ±0.11	37.5 ±0.36	40 ±0.37	0.49
AdaBoost	93.8 ±0.025	88.6 ±0.044	93.8 ±0.025	91.1 ±0.037	0.91
Bagging	58.7 ±0.39	93.2 ±0.05	58.7 ±0.39	60 ±0.37	0.74

TABLE 7. Comparison of the AWEE on the baseline methods on the UNSW18 dataset.

Method	Accuracy	Precision	Recall	F1-score	G-Mean
AWEE	99.9 ±3.6e-6	99.9 ±3.6e-6	99.9 ±3.6e-6	99.9 ±3.6e-6	0.99
Decision trees	99.9 ±0.001	99.9 ±0.001	99.9 ±0.001	99.9 ±0.001	0.99
Naive Bayes	70.5 ±0.001	75.8 ±0.004	70.5 ±0.001	68.8 ±0.002	0.73
Gradient Boosting	99 ±0.003	99 ±0.003	99 ±0.003	99 ±0.003	0.99
AdaBoost	78.4 ±0.03	80 ±0.02	78.4 ±0.03	77.6 ±0.03	0.79
Bagging	99.9 ±0.0	99.9 ±0.0	99.9 ±0.0	99.9 ±0.0	0.99

TABLE 8. Comparison of the AWEE on the baseline methods on the CSE-CIC-IDS2018 dataset.

Method	Accuracy	Precision	Recall	F1-score	G-Mean
AWEE	98.9 ±0.0032	98.9 ±0.0034	98.9 ±0.0019	98.8 ±0.0032	0.99
Decision trees	91 ±0.0002	86.5 ±0.005	91 ±0.0003	88.2 ±0	0.88
Naive Bayes	79.5 ±0	62.9 ±0.005	79.5 ±0.005	70 ±0.007	0.70
Gradient Boosting	89 ±0.005	90 ±0.005	88.9 ±0.007	88.3 ±0.004	0.89
AdaBoost	83.3 ±0.0	81.7 ±0.0	83.3 ±0.0	81.7 ±0.0	0.82
Bagging	91.1 ±0.0	85.5 ±0.0	91.1 ±0.0	87.9 ±0.0	0.88

D. PERFORMANCE COMPARISON WITH BASELINE METHODS

In this sub-section, we present a comprehensive evaluation of AWEE by comparing its performance against established baseline methods. The selection of baseline methods encompasses decision trees, Naive Bayes, Gradient Boosting, AdaBoost, and Bagging, each chosen for its relevance in addressing class imbalance in network traffic classification. Through a rigorous comparison, our aim is to assess the effectiveness of the AWEE in achieving superior classification accuracy and robustness in the face of class imbalance challenges.

Table 6 represents the performance results on the Cambridge dataset, in which the AWEE exhibits superior performance, with an accuracy of 98.9 ±0.0032, precision of 98.9 ±0.0031, recall of 98.9 ±0.0032, and F1-score of 98.8 ±0.0032. Compared to baseline methods, such as Decision Trees, Naive Bayes, Gradient Boosting, AdaBoost, and Bagging, AWEE consistently outperforms them across all metrics. Notably, Decision Trees and Naive Bayes show lower accuracy, precision, recall, and F1-score, while Gradient Boosting and Bagging exhibit weaknesses in certain aspects. AWEE stands out as a robust solution for handling class imbalance challenges in network traffic classification in the Cambridge dataset.

Furthermore, Table 7 shows the remarkable performance of AWEE in the UNSW18 dataset, achieving an exceptional accuracy, precision, recall, and F1-score of 99.9 ±3.6e-6. In comparison, the baseline methods present varied results. Decision Trees and Gradient Boosting perform competitively with the AWEE, while Naive Bayes and AdaBoost lag behind in accuracy, precision, recall, and F1-score. Bagging also performs well, aligning closely with the AWEE. The results emphasize AWEE's capability to excel in network traffic classification tasks, particularly in scenarios with varying degrees of class imbalance, in this case a dataset with low imbalance-degree. Moreover, the analysis report in Table 8 reveals the effectiveness of AWEE in the CSE-CICIDS2018 dataset. With an accuracy of 98.9 ± 0.0032, precision of 98.9 ± 0.0034, recall of 98.9 ±0.0019, and F1-score of 98.8 ± 0.0032, the AWEE consistently outperforms the baseline methods. Decision Trees, Naive Bayes, Gradient Boosting, AdaBoost, and Bagging, while exhibiting reasonable performance, fall short of AWEE's metrics.

Table 9 presents a class-wise performance comparison based on the Cambridge dataset. It can be demonstrated that the AWEE consistently exhibits strong performance across multiple class types, striking a balance between precision and recall. It outperforms other models in terms of F1-score in several instances. All models, except Naive Bayes and AdaBoost, perform reasonable for ''DATABASE'', ''WWW'', ''SERVICES'', ''FTP-CONTROL'', ''P2P'', and ''FTP-PASV'' classes, with AWEE achieving high precision, recall, and F1-score. The AWEE model demonstrates well-balanced performance for the 'MAIL' class, outperforming other classifiers in all performance measures. It is worth noting that Decision Trees show superior performance in terms of both precision (0.97) and recall (0.99) compared to other classifiers for the 'DATABASE' class. Additionally, this classifier demonstrates enhanced recall (0.95) for the 'FTP-PASV' class when compared to its counterparts. Totally, the AWEE demonstrates robust performance across diverse class types, especially the minority classes (e.g., ''ATTACK'', ''P2P'', and ''SERVICES''), reinforcing its adaptability to handle challenges associated with class imbalance.

The results for the UNSW18 dataset, as presented in Table 10, demonstrate strong performance across all classes by the classifiers. The UNSW18 dataset, being less imbalanced, does not exhibit challenges often associated with imbalanced datasets. Indeed, the number of instances for the minority class, i.e., ''RT'', in the UNSW-2018 IoT Botnet dataset is relatively substantial compared to other datasets with imbalanced classes. With 44,009 instances, it provides a reasonable amount of data from which the classifier can learn, despite being a minority class. Another plausible explanation could be that our feature representations are exceptionally discriminative for the UNSW18 dataset, thereby enabling robust segregation of minority class instances from the rest of the classes. Models, including AWEE, showcase high performance across various classes without notable unbalanced data-related biases.

Table 11 presents a comparative analysis of evaluation results for imbalance traffic classification on the CSE-CIC-IDS2018 dataset. Most models, including the AWEE, Decision trees, and Bagging, perform well for ''DoS Slowloris'', achieving high precision, recall, and F1-score. Other models, including Naive Bayes, AdaBoost, and Boosting, struggle with this class. The AWEE achieves perfect and stable performance for ''DoS GoldenEye''. Other models, particularly Naive Bayes, demonstrate challenges in precision and F1-score. In the 'FTP-BruteForce' class, both the AWEE and Bagging outperform other models, achieving perfect precision, recall, and F1-score (1.00). In particular, other classifiers exhibit less consistent performance. The analysis highlights the robust performance of the AWEE in various classes in the CSE-CIC-IDS2018 dataset. Decision Trees and Bagging also show consistent strength, except for the ''SSH-Bruteforce'' class, while other models exhibit variations in performance across different classes.

A general comment for all results up to now is that the imbalance in class distribution can heavily influence model training and performance. In this case, the significant majority of samples in one class, for example ''BENIGN'' in the CSE-CIC-IDS2018 dataset, class may dominate the learning process, making it challenging for models to learn patterns from minority classes (e.g., ''SSH-Bruteforce'' class). Moreover, different classification algorithms have varying capabilities to handle class imbalance. Some methods, such as Naive Bayes, may inherently struggle with imbalanced datasets, leading to poorer performance for minority classes. At the same time, the AWEE method incorporates techniques such as ensemble learning and adaptive weight to address class imbalance, enabling it to better learn from minority class instances.

E. COMPARISON WITH COMPETING METHODS

As previously highlighted, studies focused on addressing the class imbalance issue in the literature can be broadly classified into three distinct groups: algorithmic-level approaches, cost-sensitive methodologies, and data-level techniques. In this section, a series of experiments has been undertaken to assess the performance of AWEE compared to various competitive methods. The selection of these methods is based

TABLE 9. Performance comparison of the models on the Cambridge dataset (P = Precision, R = Recall, and F1. = F1-score).

	AWEE			Decision trees			Naive Bayes			G. Boosting			AdaBoost			Bagging		
Class type	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.
ATTACK	**0.90**	**0.74**	**0.81**	0.51	0.72	0.60	0.00	0.00	0.00	0.36	0.72	0.49	0.00	0.00	0.00	0.58	0.51	0.56
DATABASE	0.97	0.99	**0.98**	**0.97**	**0.99**	0.98	0.00	0.00	0.00	0.50	0.41	0.41	0.00	0.00	0.00	0.74	0.82	0.73
FTP-CON.	**0.95**	**0.95**	**0.95**	0.30	0.74	0.30	0.17	0.00	0.00	0.58	0.80	0.80	0.00	0.00	0.00	0.13	0.19	0.11
FTP-DATA	**0.96**	**0.93**	**0.94**	0.49	0.50	0.50	0.50	0.00	0.00	0.70	0.79	0.73	0.25	0.20	0.21	0.60	0.60	0.50
FTP-PASV	**0.95**	0.91	**0.92**	0.91	**0.95**	0.90	0.26	0.00	0.00	0.89	0.95	0.92	0.27	0.37	0.26	0.61	0.61	0.57
MAIL	**0.94**	**0.89**	**0.90**	0.90	0.18	0.22	0.30	0.32	0.18	0.77	0.52	0.51	0.59	0.75	0.61	0.55	0.46	0.45
P2P	**0.95**	**0.93**	**0.93**	0.41	0.91	0.41	0.02	0.36	0.03	0.29	0.59	0.27	0.00	0.00	0.00	0.35	0.76	0.25
SERVICES	**0.98**	**0.96**	**0.96**	0.71	0.99	0.81	0.00	0.00	0.00	0.60	0.89	0.70	0.00	0.00	0.00	0.60	0.60	0.57
WWW	**1.00**	**0.99**	**1.00**	0.98	0.78	0.78	0.73	0.78	0.77	0.68	0.55	0.53	0.80	0.80	0.79	0.76	0.57	0.59

TABLE 10. Performance comparison of the models on the UNSW18 dataset (P = Precision, R = Recall, and F1. = F1-score).

	AWEE			Decision trees			Naive Bayes			G. Boosting			AdaBoost			Bagging		
Class type	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.
RT	**1.00**	**1.00**	**1.00**	1.00	1.00	1.00	0.87	0.44	0.58	1.00	1.00	1.00	0.87	0.61	0.71	1.00	1.00	1.00
DoS	**1.00**	**1.00**	**1.00**	1.00	1.00	1.00	0.66	0.96	0.78	1.00	1.00	1.00	0.74	0.92	0.81	1.00	1.00	1.00
DDoS	**1.00**	**1.00**	**1.00**	1.00	1.00	1.00	0.79	0.22	0.34	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

TABLE 11. Performance comparison of the models on the CSE-CIC-IDS2018 dataset (P = Precision, R = Recall, and F1. = F1-score).

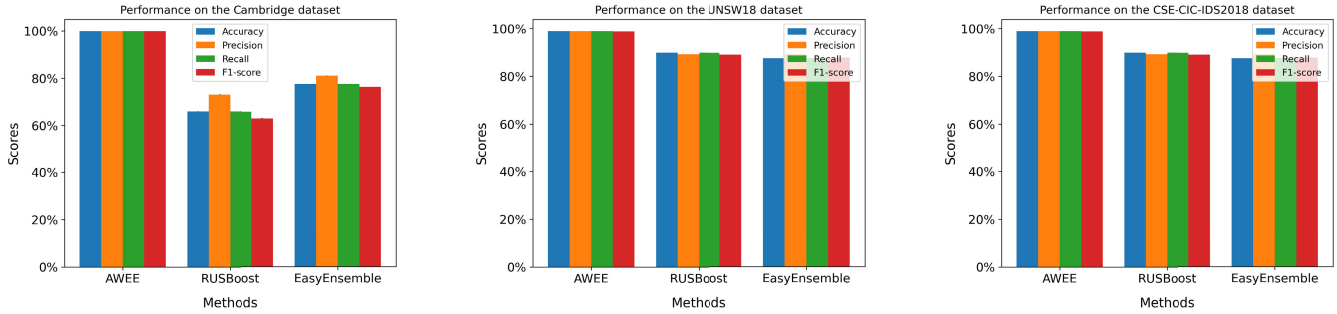
	AWEE			Decision trees			Naive Bayes			G. Boosting			AdaBoost			Bagging		
Class type	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.	P	R	F1.
BENIGN	**1.00**	1.00	**1.00**	1.00	1.00	1.00	0.16	0.20	0.18	0.89	0.99	0.94	0.94	0.91	0.93	0.95	1.00	0.97
DoS GoldenEye	**1.00**	0.99	**1.00**	0.32	**1.00**	0.48	0.00	0.00	0.00	1.00	0.46	0.63	0.25	0.96	0.40	0.32	1.00	0.49
DoS Slowloris	**0.99**	0.83	0.91	0.98	0.89	0.93	0.00	0.00	0.00	0.42	0.86	0.56	0.00	0.00	0.00	0.99	**0.98**	**0.98**
FTP-BruteForce	**1.00**	**1.00**	**1.00**	0.67	1.00	0.80	0.00	0.00	0.00	1.00	0.50	0.66	0.67	1.00	0.80	1.00	1.00	1.00
SSH-Bruteforce	**1.00**	**1.00**	**1.00**	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.51	0.68	0.00	0.00	0.00	0.00	0.00	0.00

on their distinct approaches to addressing the challenge of class imbalance.

1) ALGORITHMIC-LEVEL TECHNIQUES

Algorithmic-level approaches represent a category of methods designed to tackle class imbalance by modifying the learning algorithm itself. In this study, we include two prominent algorithmic-level techniques: Random Under-Sampling Boosting Classifier (RUSBoostClassifier) [48] and EasyEnsemble [49]. These methods operate by altering the training process to better accommodate imbalanced class distributions. The RUSBoostClassifier integrates under-sampling of the majority class within each boosting iteration, dynamically adjusting the sample weights to give more emphasis to the minority class. On the other hand, EasyEnsemble adopts an ensemble learning strategy, creating multiple balanced subsets through random under-sampling and training individual classifiers on each subset. In this section, we provide a comprehensive evaluation of the performance of these algorithmic-level approaches and compare them with our proposed AWEE method to address the challenges posed by class imbalance in the context of network traffic classification. The performance results on the Cambridge dataset are visually presented in Figure 3a. In particular, AWEE consistently outperforms both RUSBoost and EasyEnsemble on all metrics evaluated. This observation underscores the superior efficacy of the AWEE in addressing the challenges associated with class imbalance within the Cambridge dataset. Detailed analysis reveals that RUSBoost exhibits the least favorable performance among the three methodologies, particularly in terms of accuracy and precision. However, EasyEnsemble demonstrates better performance than RUSBoost but falls short of AWEE on multiple metrics, including accuracy, precision, and F1-score. The reason behind this is that EasyEnsemble employs a simple ensemble learning technique in which multiple classifiers are trained in different subsets of the data, with each subset containing a balanced distribution of minority class samples achieved through random undersampling. In contrast, AWEE utilizes an ensemble-of-ensembles architecture, coupled with dynamic weight adjustment and sliding window-based validation. This architecture enables the AWEE to adapt to evolving data dynamics and maintain high accuracy by dynamically adjusting the weights assigned to base learners and validating the model performance using a sliding window approach. Figures 3b and 3c show the performance results on the UNSW18 and CSE-CIC-IDS2018 datasets, respectively. AWEE outperforms both RUSBoost and EasyEnsemble across all metrics in the datasets.

EasyEnsemble exhibits the lowest performance among the three methods. RUSBoost performs better than EasyEnsemble but falls short of the AWEE in terms of accuracy, precision, recall, and F1-score. This analysis suggests that the AWEE is a highly effective method of addressing class imbalance in the context of network traffic classification.



(a) Cambridge (b) UNSW18 (c) CSE-CIC-IDS2018

FIGURE 3. Comparison of the AWEE with the algorithmic methods on the benchmark datasets.

TABLE 12. Performance comparison of the competing methods on the Cambridge dataset (P = Precision, R = Recall, F1. = F1-score, and G. = G-Mean).

	AWEE				RUSBoost				EasyEnsemble			
Class type	P	R	F1.	G.	P	R	F1.	G.	P	R	F1.	G.
ATTACK	**0.90**	**0.74**	**0.81**	**0.74**	0.09	0.23	0.07	0.23	0.00	0.00	0.00	0.00
DATABASE	**0.97**	**0.99**	**0.98**	**0.99**	0.37	0.68	0.37	0.43	0.00	0.00	0.00	0.00
FTP-CON.	**0.95**	**0.95**	**0.95**	**0.98**	0.03	0.24	0.06	0.24	0.00	0.00	0.00	0.00
FTP-DATA	0.96	0.93	0.94	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.25	0.25	0.25	**1.00**
FTP-PASV	**0.95**	**0.91**	**0.92**	**0.96**	0.57	0.34	0.41	0.38	0.00	0.00	0.00	0.00
MAIL	**0.94**	**0.89**	**0.90**	**0.91**	0.54	0.20	0.27	0.19	0.00	0.00	0.00	0.00
P2P	**0.95**	**0.93**	**0.93**	**0.92**	0.42	0.09	0.10	0.09	0.00	0.00	0.00	0.00
SERVICES	**0.98**	**0.96**	**0.96**	**0.74**	0.05	0.43	0.10	0.42	0.00	0.00	0.00	0.00
WWW	**1.00**	**0.99**	**1.00**	**0.99**	0.88	0.62	0.70	0.62	0.67	**1.00**	0.80	**1.00**

TABLE 13. Performance comparison of the competing methods on the UNSW18 dataset (P = Precision, R = Recall, F1. = F1-score, and G. = G-Mean).

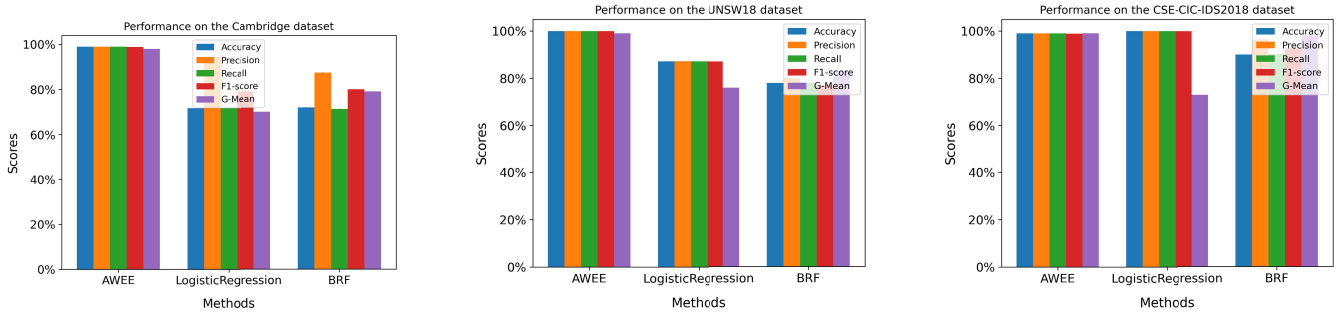
	AWEE				RUSBoost				EasyEnsemble			
Class type	**P**	**R**	**F1.**	**G.**	**P**	**R**	**F1.**	**G.**	**P**	**R**	**F1.**	**G.**
RT	**1.00**	**1.00**	**1.00**	**1.00**	0.60	0.64	0.71	0.84	0.91	0.58	0.70	0.56
DoS	**1.00**	**1.00**	**1.00**	**1.00**	0.78	0.50	0.58	0.48	0.73	0.94	0.82	0.94
DDoS	**1.00**	**1.00**	**1.00**	**1.00**	0.91	0.97	0.93	0.97	**1.00**	**1.00**	**1.00**	0.99

TABLE 14. Performance comparison of the competing methods on the CSE-CIC-IDS2018 dataset (P = Precision, R = Recall, F1. = F1-score, G = G-Mean).

	AWEE				RUSBoost				EasyEnsemble			
Class type	**P**	**R**	**F1.**	**G**	**P**	**R**	**F1.**	**G**	**P**	**R**	**F1.**	**G**
BENIGN	**1.00**	**1.00**	1.00	**1.00**	**1.00**	0.99	0.99	0.99	**1.00**	0.96	0.98	0.98
DoS GoldenEye	**1.00**	0.99	1.00	0.99	0.27	**1.00**	0.43	1.00	0.32	**1.00**	0.48	0.99
DoS Slowloris	**0.99**	0.83	0.91	0.91	0.01	0.11	0.02	0.33	0.05	**0.94**	0.10	0.97
FTP-BruteForce	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
SSH-Bruteforce	**1.00**	**1.00**	**1.00**	**1.00**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Furthermore, to improve the evaluation of our method, a detailed analysis of performance per class is presented in Tables 12, 13, and 14. From the observations in Table 12, it is evident that our model consistently demonstrates high performance across all classes, as evidenced by the high precision, recall, and F1-scores. In contrast, RUSBoost faces challenges in effectively addressing class imbalance, resulting in disparate performance levels across different classes. While RUSBoost excels in certain classes (e.g., FTP-DATA), it encounters difficulties in others (e.g., ATTACK and FTP-CON). Furthermore, EasyEnsemble exhibits suboptimal performance in all classes, and several metrics report zero scores. This suggests a significant struggle in correctly classifying instances across various classes, highlighting limitations in its ability to handle class imbalance effectively.

Table 13 shows the results for the UNSW18 dataset. The AWEE demonstrates excellent performance across all classes, achieving perfect precision, recall, and F1-scores for each class label. In contrast, RUSBoost displays relatively lower performance compared to the AWEE, especially in the RT and DoS classes. EasyEnsemble performs well in the RT and DDoS classes, but exhibits lower precision and F1-scores in the DoS class. Concerning the CSE-CIC-IDS2018 dataset, the AWEE demonstrates consistent and strong performance across various attack types, with perfect scores in several classes. RUSBoost performs well in some



(a) Cambridge (b) UNSW18 (c) CSE-CIC-IDS2018

FIGURE 4. **Comparison of the AWEE with the cost-sensitive methods on the benchmark datasets.**

classes but struggles in others, particularly with lower recall in certain attack types. Meanwhile, EasyEnsemble shows competitive performance but has limitations in certain attack categories, such as SSH-Bruteforce.

2) COST-SENSITIVE TECHNIQUES

In the pursuit of addressing the challenge of class imbalance, a specialized category of methodologies, known as cost-sensitive techniques, emerges as a promising avenue. Cost-sensitive techniques focus on adjusting the misclassification costs associated with different classes. This section delves into a comparative analysis between the proposed AWEE method and two notable cost-sensitive techniques, including cost-sensitive LogisticRegression and BalancedRandomForestClassifier (BRF). Cost-sensitive LogisticRegression augments traditional logistic regression by introducing a cost-sensitive mechanism. By assigning varying misclassification costs to different classes, this approach provides a tailored solution to imbalanced class distributions. On the other hand, BRF extends the popular random forest algorithm to accommodate imbalanced datasets. By adjusting class weights during training, this technique ensures equal consideration for minority and majority classes, fostering a more balanced learning process.

Figure 4 illustrates the average performance of both the cost-sensitive approaches and AWEE on the benchmark datasets. Specifically, Figure 4a shows the performance of the proposed method along with the selected cost-sensitive approaches in the Cambridge dataset. As can be seen, AWEE outperforms both cost-sensitive LogisticRegression and BRF in accuracy (98.9%), precision (98.9%), recall (98.9%), and F1 score (98.9%). The cost-sensitive LogisticRegression demonstrates high precision at 94.36%, surpassing that of BRF (87.3%). However, this improvement comes at the expense of a lower recall rate, which is 71.73%. Considering BRF, this cost-sensitive classifier achieves a balance between precision and recall, but with slightly lower overall performance compared to LogisticRegression. Concerning the UNSW18 dataset (see Figure 4b), AWEE continues to exhibit its dominance by consistently delivering superior performance and stability. Although Cost-sensitive Logistic Regression and BRF yield acceptable results, they show variability. Additionally, BRF consistently falls behind both AWEE and Logistic Regression across all metrics. The outcomes on the CSE-CIC-IDS2018 dataset indicate that both the AWEE and LogisticRegression deliver exceptional performance, with LogisticRegression holding a slight advantage in performance measures (A, P, R, and F1 = 99.9%). While BRF performs well (A = 90%, P = 96.4%, R = 90.2%, F1 = 92.6%), it exhibits lower accuracy compared to the AWEE and LogisticRegression.

All in all, the superior performance of the AWEE method compared to cost-sensitive techniques can be justified by several factors. More specifically, AWEE leverages an ensemble-of-ensembles architecture, which inherently promotes model diversity and robustness. Unlike cost-sensitive LogisticRegression, which adjusts misclassification costs to handle class imbalance, and BRF, which adjusts class weights during training, AWEE combines multiple base learners trained on different subsets of the data using various algorithms and hyperparameters. This diverse ensemble enables AWEE to capture a broader range of patterns in the data, improving its ability to generalize well to unseen samples and improving its overall performance.

In a manner analogous to the preceding experiment, to augment the evaluation of our approach, we provide a comprehensive analysis of the performance for each class. Table 15 demonstrates the results for the Cambridge dataset. As can be seen, AWEE achieves consistently high precision, recall, and F1-score across most classes, such as "ATTACK", "DATABASE", "FTP-CON.", and "WWW". Cost-sensitive LogisticRegression shows good precision for certain classes, including "FTP-DATA", "FTP-PASV", and "MAIL". Generally, this classifier provides stable results, as evidenced by moderate standard deviations. In contrast, BRF demonstrates diminished precision and F1-scores for certain classes, such as "P2P," "ATTACK," "DATABASE," and "FTP-CON," when juxtaposed with AWEE.

AWEE remains the top-performing method in the UNSW18 dataset (see Table 16), showcasing perfect scores and exceptional stability. While both Cost-sensitive LogisticRegression and BRF offer competitive results,

TABLE 15. Performance comparison of the cost-sensitive methods on the Cambridge dataset (P= Precision, R= Recall, F1.= F1-score, and G.= G-Mean).

	AWEE				Cost-sensitive LogisticRegression				BRF			
Class type	P	R	F1.	G.	P	R	F1.	G.	P	R	F1.	G.
ATTACK	**0.90**	0.74	**0.81**	0.74	0.07	**0.84**	0.13	**0.9**	0.14	0.73	0.24	0.8
DATABASE	**0.97**	**0.99**	**0.98**	**0.99**	0.57	0.99	0.72	0.97	0.11	0.89	0.21	0.79
FTP-CON.	**0.95**	**0.95**	**0.95**	**0.98**	0.79	0.97	0.87	0.96	0.47	0.9	0.51	0.74
FTP-DATA	0.96	0.93	0.94	**0.99**	**0.96**	**0.99**	**0.97**	**0.99**	0.65	0.96	0.8	**0.99**
FTP-PASV	0.95	0.91	0.92	**0.96**	**0.95**	**0.96**	**0.95**	**0.96**	0.44	0.32	0.33	0.48
MAIL	0.94	0.89	0.90	**0.91**	**0.94**	**0.89**	**0.91**	0.9	0.76	0.5	0.51	0.72
P2P	**0.95**	**0.93**	**0.93**	**0.92**	0.32	0.79	0.40	0.82	0.0	0.0	0.0	0.00
SERVICES	**0.98**	0.96	**0.96**	0.74	0.10	**0.98**	0.20	**0.96**	0.56	0.76	0.64	0.93
WWW	**1.00**	**0.99**	**1.00**	**0.99**	0.99	0.63	0.76	0.7	0.79	0.79	0.8	0.85

TABLE 16. Performance comparison of the cost-sensitive methods on the UNSW18 dataset (P = Precision, R = Recall, F1. = F1-score, and G. = G = Mean).

	AWEE				Cost-sensitive LogisticRegression				BRF			
Class type	P	R	F1.	G.	P	R	F1.	G.	P	R	F1.	G.
RT	**1.00**	**1.00**	**1.00**	**1.00**	0.84	0.90	0.87	0.92	0.89	0.58	0.70	0.74
DoS	**1.00**	**1.00**	**1.00**	**1.00**	0.90	0.86	0.88	0.90	0.74	0.93	0.82	0.94
DDoS	**1.00**	**1.00**	**1.00**	**1.00**	0.85	0.65	0.74	0.78	0.56	0.99	0.71	0.97

TABLE 17. Performance comparison of the cost-sensitive methods on the CSE-CIC-IDS2018 dataset (P = Precision, R = Recall, F1. = F1-score, and G. = G-Mean).

	AWEE				Cost-sensitive LogisticRegression				BRF			
Class type	P	R	F1.	G.	P	R	F1.	G.	P	R	F1.	G.
BENIGN	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.98	0.90	0.94	0.92
DoS GoldenEye	**1.00**	0.99	**1.00**	0.99	0.99	**1.00**	**1.00**	**1.00**	0.26	0.99	0.41	0.89
DoS Slowloris	**0.99**	0.83	0.91	0.91	0.91	**0.99**	**0.95**	**0.99**	0.13	0.74	0.22	0.79
FTP-BruteForce	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.98	0.87	0.92	0.90
SSH-Bruteforce	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.99	0.98	0.99	0.96

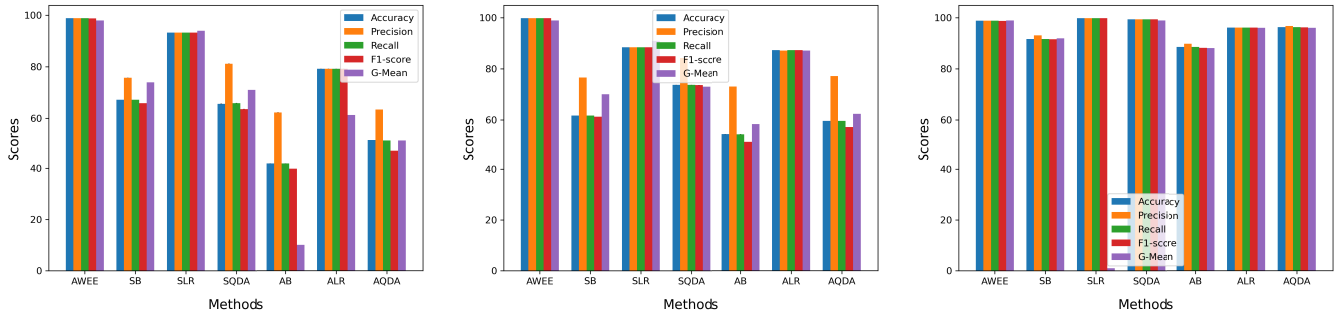
they lag behind the AWEE. Moreover, BRF is comparable to cost-sensitive LogisticRegression but slightly less precise.

Last but not least, AWEE and Cost-sensitive LogisticRegression achieve perfect precision, recall, and F1-score on the CSE-CIC-IDS2018 dataset (see Table 17). To be more specific, the cost-sensitive LogisticRegression attains the highest recall and F1-score for the "DoS GoldenEye" and "DoS Slowloris" classes. Conversely, BRF encounters difficulties in attaining perfect precision, particularly in the case of "SSH-Bruteforce."

3) DATA-LEVEL TECHNIQUES

In this subsection, we explore two prominent data-level techniques: Synthetic Minority Over-sampling Technique (SMOTE) [50] and Adaptive Synthetic Sampling Approach for Imbalanced Learning (ADASYN) [51]. These techniques aim to alleviate the challenge of class imbalance by generating synthetic instances for the minority class, fostering a more balanced and representative training dataset. To thoroughly assess the effectiveness of these data-level techniques, we incorporate them into three distinct classifiers: Naive Bayes, Logistic Regression, and Quadratic Discriminant Analysis. This integration results in six classifiers, denoted as follows: SMOTE Byesian (SB), SMOTE Logistic Regression (SLR), SMOTE Quadratic Discriminant Analysis (SQDA), ADASYN Byesian (AB), ADASYN Logistic Regression (ALR) and ADASYN Quadratic Discriminant Analysis (AQDA). The subsequent experiments provide a nuanced understanding of the interaction between data-level modifications and classifier performance, offering valuable insights into effective strategies for handling class imbalance. The proposed AWEE method is also included in this comparative analysis to discern its efficacy in relation to these established approaches.

Figure 5 illustrates the performance comparison between the data-level techniques and the AWEE. As for Figure 5a where the classifiers are validated in the Cambridge dataset, the AWEE and SLR classifiers exhibit high accuracy, surpassing 90%, where the SB, ALR, and AQDA classifiers show moderate accuracy, ranging between 65% and 80%. In addition, the SQDA and AB classifiers display comparatively lower accuracy, around 40% to 60%. AWEE, SLR, SQDA and ALR classifiers achieve high precision values, indicating a low false positive rate. The AB and AQDA classifiers exhibit the lowest performance among the six classifiers evaluated on the Cambridge dataset. These classifiers demonstrate lower accuracy, precision, recall, and F1-score compared to the other methods.



(a) Cambridge (b) UNSW18 (c) CSE-CIC-IDS2018

FIGURE 5. **Comparison of the AWEE with the data-level methods on the benchmark datasets.**

Similarly to the trend observed in the Cambridge dataset, the AWEE exhibits outstanding performance in all metrics in the UNSW18 dataset. Accuracy, precision, recall, and F1-score reaching almost 100%. The SB and SQDA classifiers show relatively lower performance compared to the AWEE. Although precision is moderate, recall and F1-score are noticeably lower, indicating potential challenges in capturing positive instances and achieving a balance between precision and recall. SLR exhibits good accuracy, precision, recall, and F1-score. The results suggest that combining SMOTE with Logistic Regression yields a well-balanced performance on the dataset. The AB classifier demonstrates moderate performance in all metrics, while in contrast, ALR achieves good accuracy, precision, recall, and F1-score, showcasing a balanced performance on the dataset. Finally, AQDA demonstrates moderate precision, but recall and F1-score are lower. This suggests that there may be difficulties in effectively capturing positive instances and achieving a balanced trade-off between precision and recall.

The analysis of classifier performance on the CSE-CIC-IDS2018 dataset reveals that the SLR exhibits excellent performance, with accuracy, precision, recall, and F1 score all approaching 100%. Furthermore, the SQDA classifier demonstrates exceptional performance across all metrics, similar to the AWEE. AQDA demonstrates high performance across all metrics, similar to ALR, with accuracy, precision, recall, and F1-score that exceed 96%. The SB classifier shows good performance. Although recall and F1-score are slightly lower, overall, SB performs well in capturing positive instances and achieving a balance between precision and recall. Lastly, it is worth noting that AB shows the lowest performance, mirroring its performance on other datasets.

In validation of the aforementioned outcomes, both SMOTE and ADASYN aim to address the challenge of class imbalance by generating synthetic instances for the minority class, thus creating a more balanced and representative training dataset. This augmentation helps classifiers trained on such datasets to better learn the underlying patterns and improve their performance on minority class instances. However, the success of these techniques depends on their ability to generate synthetic samples that accurately represent the minority class distribution without introducing excessive noise or bias. When comparing the performance of classifiers incorporating SMOTE and ADASYN with the AWEE method, the propoted model diversity in AWEE method enhances its ability to generalize well to unseen samples and improves its overall performance compared to classifiers trained solely on augmented data.

VI. CHALLENGES, LIMITATIONS, AND FUTURE WORK

The development and application of the AWEE method posed several challenges, highlighting opportunities for further refinement and exploration. The AWEE method addresses complex issues in network traffic classification, but the process presented several challenges. First, the datasets used in this study exhibited significant class imbalance, requiring robust mechanisms to ensure that minority classes were adequately represented during training and classification. Second, the dynamic nature of network traffic, including concept drift and evolving data patterns, necessitated the development of an adaptive weight adjustment mechanism to maintain model accuracy over time. Finally, the ensemble-of-ensembles architecture introduced substantial computational overhead, particularly due to iterative weight adjustments and sliding window-based validation, which demanded significant resources during training and evaluation.

Despite its demonstrated effectiveness, the AWEE method has certain limitations that must be acknowledged. The computational demands of the ensemble-of-ensembles framework may limit its scalability, particularly in scenarios involving extremely large datasets or strict real-time constraints. Furthermore, while the method was tested on three benchmark datasets, its generalizability to other domains, such as healthcare or financial fraud detection, remains unverified [52]. Furthermore, the effectiveness of the AWEE method is heavily based on careful hyperparameter tuning, which can be time-consuming and may require substantial domain expertise.

Building on the challenges and limitations encountered, several avenues for future research are proposed to extend the applicability and impact of the AWEE method. First, optimizing the AWEE framework for real-time applications

by developing lightweight ensemble models or distributed training techniques could enhance its scalability and efficiency. Second, expanding the evaluation of AWEE to other domains, such as cybersecurity, medical diagnostics, and financial fraud detection, would provide valuable insights into its robustness and adaptability. Third, integrating AWEE with deep learning models could improve feature extraction and classification accuracy, particularly in scenarios with complex or high-dimensional datasets. Finally, automating hyperparameter optimization using systematic techniques, such as Bayesian optimization or genetic algorithms, could reduce manual intervention, streamline the training process, and further improve the method's practicality.

VII. CONCLUSION

In this paper, the AWEE method has been proposed as a robust and effective solution for addressing the class imbalance issue in network traffic classification. The architecture of the *ensemble of ensembles*, coupled with dynamic weight adjustment and sliding window-based validation, allows AWEE to adapt to evolving data dynamics and maintain high precision. Experimental results on benchmark datasets illustrate the consistent superiority of AWEE over baseline methods, including Decision Trees, Naive Bayes, Gradient Boosting, AdaBoost, and Bagging. The comparison with algorithmic-level techniques further emphasizes the efficacy of AWEE, outperforming RUSBoost and EasyEnsemble across various metrics and datasets. The comprehensive evaluation demonstrates AWEE's adaptability, resilience, and superior performance, positioning it as a valuable tool for network traffic classification in real-world scenarios characterized by imbalanced data.

REFERENCES

[1] M. Abbasi, A. Shahraki, and A. Taherkordi, "Deep learning for network traffic monitoring and analysis (NTMA): A survey," *Comput. Commun.*, vol. 170, pp. 19–41, Mar. 2021.

[2] G. Miao, G. Wu, Z. Zhang, Y. Tong, and B. Lu, "SPN: A method of few-shot traffic classification with out-of-distribution detection based on Siamese prototypical network," *IEEE Access*, vol. 11, pp. 114403–114414, 2023.

[3] A. Shahraki, M. Abbasi, A. Taherkordi, and M. Kaosar, "Internet traffic classification using an ensemble of deep convolutional neural networks," in *Proc. 4th FlexNets Workshop Flexible Netw. Artif. Intell. Supported Netw. Flexibility Agility*, Aug. 2021, pp. 38–43.

[4] Z. Chen, G. Cheng, Z. Wei, D. Niu, and N. Fu, "Classify traffic rather than flow: Versatile multi-flow encrypted traffic classification with flow clustering," *IEEE Trans. Netw. Service Manage.*, vol. 21, no. 2, pp. 1446–1466, Apr. 2024.

[5] M. Zheng, K. Ma, F. Wang, X. Hu, Q. Yu, L. Guo, and F. Chen, "Which standard classification algorithm has more stable performance for imbalanced network traffic data?" *Soft Comput.*, vol. 28, no. 1, pp. 217–234, Jan. 2024.

[6] K. Ghosh, C. Bellinger, R. Corizzo, P. Branco, B. Krawczyk, and N. Japkowicz, "The class imbalance problem in deep learning," *Mach. Learn.*, vol. 113, no. 7, pp. 4845–4901, Jul. 2024.

[7] J. Bi, Z. Guan, H. Yuan, J. Yang, and J. Zhang, "Network anomaly detection with stacked sparse shrink variational autoencoders and unbalanced XGBoost," *IEEE Trans. Sustain. Comput.*, early access, 2024.

[8] A. Abdelkhalek and M. Mashaly, "Addressing the class imbalance problem in network intrusion detection systems using data resampling and deep learning," *J. Supercomput.*, vol. 79, no. 10, pp. 10611–10644, Jul. 2023.

[9] A. Telikani, J. Shen, J. Yang, and P. Wang, "Industrial IoT intrusion detection via evolutionary cost-sensitive learning and fog computing," *IEEE Internet Things J.*, vol. 9, no. 22, pp. 23260–23271, Nov. 2022.

[10] M. Abbasi, A. Taherkordi, and A. Shahraki, "FLITC: A novel federated learning-based method for IoT traffic classification," in *Proc. IEEE Int. Conf. Smart Comput. (SMARTCOMP)*, Jun. 2022, pp. 206–212.

[11] A. Saber, B. Fergani, and M. Abbas, "Encrypted traffic classification: Combining over-and under-sampling through a PCA-SVM," in *Proc. 3rd Int. Conf. Pattern Anal. Intell. Syst. (PAIS)*, Oct. 2018, pp. 1–5.

[12] J. Liu, Y. Gao, and F. Hu, "A fast network intrusion detection system using adaptive synthetic oversampling and LightGBM," *Comput. Secur.*, vol. 106, Jul. 2021, Art. no. 102289.

[13] A. Telikani, A. H. Gandomi, K. R. Choo, and J. Shen, "A cost-sensitive deep learning-based approach for network traffic classification," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 1, pp. 661–670, Mar. 2022.

[14] A. Telikani, N. E. Rudbardeh, S. Soleymanpour, A. Shahbahrami, J. Shen, G. Gaydadjiev, and R. Hassanpour, "A cost-sensitive machine learning model with multitask learning for intrusion detection in IoT," *IEEE Trans. Ind. Informat.*, vol. 20, no. 3, pp. 3880–3890, Mar. 2024.

[15] S. Krishnaveni and S. Prabakaran, "Ensemble approach for network threat detection and classification on cloud computing," *Concurrency Comput., Pract. Exper.*, vol. 33, no. 3, p. 5272, Feb. 2021.

[16] H. Sadr, M. N. Soleimandarabi, M. Pedram, and M. Teshnelab, "Unified topic-based semantic models: A study in computing the semantic relatedness of geographic terms," in *Proc. 5th Int. Conf. Web Res. (ICWR)*, Apr. 2019, pp. 134–140.

[17] A. Azab, M. Khasawneh, S. Alrabaee, K.-K.-R. Choo, and M. Sarsour, "Network traffic classification: Techniques, datasets, and challenges," *Digit. Commun. Netw.*, vol. 10, no. 3, pp. 676–692, Jun. 2024.

[18] C. Yu, J. Lan, J. Xie, and Y. Hu, "QoS-aware traffic classification architecture using machine learning and deep packet inspection in SDNs," *Proc. Comput. Sci.*, vol. 131, pp. 1209–1216, Jan. 2018.

[19] T.-L. Huoh, Y. Luo, P. Li, and T. Zhang, "Flow-based encrypted network traffic classification with graph neural networks," *IEEE Trans. Netw. Service Manage.*, vol. 20, no. 2, pp. 1224–1237, Jun. 2022.

[20] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Comput.*, vol. 24, no. 3, pp. 1999–2012, Feb. 2020.

[21] M. S. Sheikh and Y. Peng, "Procedures, criteria, and machine learning techniques for network traffic classification: A survey," *IEEE Access*, vol. 10, pp. 61135–61158, 2022.

[22] J. Zhao, X. Jing, Z. Yan, and W. Pedrycz, "Network traffic classification for data fusion: A survey," *Inf. Fusion*, vol. 72, pp. 22–47, Aug. 2021.

[23] A. Shahraki, M. Abbasi, and Ø. Haugen, "Boosting algorithms for network intrusion detection: A comparative evaluation of real AdaBoost, gentle AdaBoost and modest AdaBoost," *Eng. Appl. Artif. Intell.*, vol. 94, Sep. 2020, Art. no. 103770.

[24] M. Elnawawy, A. Sagahyroon, and T. Shanableh, "FPGA-based network traffic classification using machine learning," *IEEE Access*, vol. 8, pp. 175637–175650, 2020.

[25] I. D. Mienye and Y. Sun, "A survey of ensemble learning: Concepts, algorithms, applications, and prospects," *IEEE Access*, vol. 10, pp. 99129–99149, 2022.

[26] A. Nitish, J. Hanumanthappa, S. P. S. Prakash, and K. Krinkin, "Class imbalance and concept drift invariant online botnet threat detection framework for heterogeneous IoT edge," *Comput. Secur.*, vol. 141, Jun. 2024, Art. no. 103820.

[27] R. S. Rao, S. Dewangan, A. Mishra, and M. Gupta, "A study of dealing class imbalance problem with machine learning methods for code smell severity detection using PCA-based feature selection technique," *Sci. Rep.*, vol. 13, no. 1, p. 16245, Sep. 2023.

[28] F. Ullah, S. Ullah, G. Srivastava, and J. C.-W. Lin, "IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic," *Digit. Commun. Netw.*, vol. 10, no. 1, pp. 190–204, Feb. 2024.

[29] A. Thakkar and R. Lohiya, "Attack classification of imbalanced intrusion data for IoT network using ensemble learning-based deep neural network," *IEEE Internet Things J.*, vol. 10, no. 13, pp. 11888–11895, Jun. 2023.

[30] J. Jiang, F. Liu, Y. Liu, Q. Tang, B. Wang, G. Zhong, and W. Wang, "A dynamic ensemble algorithm for anomaly detection in IoT imbalanced data streams," *Comput. Commun.*, vol. 194, pp. 250–257, Oct. 2022.

[31] H. Ren, Y. Tang, W. Dong, S. Ren, and L. Jiang, "DUEN: Dynamic ensemble handling class imbalance in network intrusion detection," *Expert Syst. Appl.*, vol. 229, Nov. 2023, Art. no. 120420.

[32] Hartono and R. B. Y. Syah, "Hybrid approach with membership-density based oversampling for handling multi-class imbalance in Internet traffic identification with overlapping and noise," *ICT Exp.*, vol. 10, no. 5, pp. 1094–1102, Oct. 2024.

[33] A. Fernández, "Cost-sensitive learning," in *Learning From Imbalanced Data Sets*. Amsterdam, The Netherlands: Elsevier, 2018, pp. 63–78.

[34] X. Hu, H. Chen, J. Zhang, H. Chen, S. Liu, X. Li, Y. Wang, and X. Xue, "GAT-COBO: Cost-sensitive graph neural network for telecom fraud detection," *IEEE Trans. Big Data*, vol. 10, no. 4, pp. 528–542, Aug. 2024.

[35] N. Gupta, V. Jindal, and P. Bedi, "CSE-IDS: Using cost-sensitive deep learning and ensemble algorithms to handle class imbalance in network-based intrusion detection systems," *Comput. Secur.*, vol. 112, Jan. 2022, Art. no. 102499.

[36] S. Fotouhi, S. Asadi, and M. W. Kattan, "A comprehensive data level analysis for cancer diagnosis on imbalanced data," *J. Biomed. Informat.*, vol. 90, Feb. 2019, Art. no. 103089. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1532046418302302

[37] Z. Xu, D. Shen, Y. Kou, and T. Nie, "A synthetic minority oversampling technique based on Gaussian mixture model filtering for imbalanced data classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 3, pp. 3740–3753, Mar. 2022.

[38] H. Yang, J. Xu, Y. Xiao, and L. Hu, "SPE-ACGAN: A resampling approach for class imbalance problem in network intrusion detection systems," *Electronics*, vol. 12, no. 15, p. 3323, Aug. 2023.

[39] M. Dener, S. Al, and G. Ok, "RFSE-GRU: Data balanced classification model for mobile encrypted traffic in big data environment," *IEEE Access*, vol. 11, pp. 21831–21847, 2023.

[40] R. Zuech, J. Hancock, and T. M. Khoshgoftaar, "Detecting web attacks using random undersampling and ensemble learners," *J. Big Data*, vol. 8, no. 1, pp. 1–20, Dec. 2021.

[41] B. Silva, R. Silveira, M. Silva Neto, P. Cortez, and D. Gomes, "A comparative analysis of undersampling techniques for network intrusion detection systems design," *J. Commun. Inf. Syst.*, vol. 36, no. 1, pp. 31–43, 2021.

[42] J. Qin, X. Han, C. Wang, Q. Hu, B. Jiang, C. Zhang, and Z. Lu, "Network traffic classification based on SD sampling and hierarchical ensemble learning," *Secur. Commun. Netw.*, vol. 2023, pp. 1–16, May 2023.

[43] A. W. Moore and D. Zuev, "Internet traffic classification using Bayesian analysis techniques," in *Proc. ACM SIGMETRICS Int. Conf. Meas. Model. Comput. Syst.*, Jun. 2005, pp. 50–60.

[44] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Gener. Comput. Syst.*, vol. 100, pp. 779–796, Nov. 2019.

[45] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *ICISSp*, vol. 1, pp. 108–116, Jan. 2018.

[46] P. Wang, F. Ye, X. Chen, and Y. Qian, "DataNet: Deep learning based encrypted network traffic classification in SDN home gateway," *IEEE Access*, vol. 6, pp. 55380–55391, 2018.

[47] A. Sharma, "Guided stochastic gradient descent algorithm for inconsistent datasets," *Appl. Soft Comput.*, vol. 73, pp. 1068–1080, Dec. 2018.

[48] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Trans. Syst., Man, Cybern., A, Syst. Hum.*, vol. 40, no. 1, pp. 185–197, Jan. 2010.

[49] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 39, no. 2, pp. 539–550, Apr. 2009.

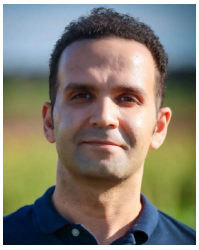
[50] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.

[51] W. Chen, K. Yang, Z. Yu, Y. Shi, and C. L. P. Chen, "A survey on imbalanced learning: Latest research, applications and future directions," *Artif. Intell. Rev.*, vol. 57, no. 6, pp. 1–51, May 2024.

[52] Z. A. Saberi, H. Sadr, and M. R. Yamaghani, "An intelligent diagnosis system for predicting coronary heart disease," in *Proc. 10th Int. Conf. Artif. Intell. Robot. (QICAR)*, Feb. 2024, pp. 131–137.

MAHMOUD ABBASI (Member, IEEE) received the B.Eng. degree from the Department of Computer Engineering, Islamic Azad University of Birjand, and the M.Sc. degree from the Department of Computer Engineering, Islamic Azad University of Mashhad. He is currently pursuing the Ph.D. degree in the IoTalentum with the BISITE Research Group, University of Salamanca. His current research interests include the general area of communication systems and networks and ML, the Internet of Things, and blockchain.

SEBASTIÁN LÓPEZ FLÓREZ received the degree in mechatronics engineering and the M.Sc. degree in electrical engineering from the Universidad Tecnológica de Pereira, with his final project focusing on computer vision for scene detection from object semantics. His interests lie in developing flexible, interpretable, and scalable machine learning models, particularly in deep learning, Gaussian processes, and kernel learning. He aims to create impactful solutions while understanding the mathematical foundations behind them, with a focus on interpretable models like probabilistic generative models, physics-inspired methods, and Bayesian approaches, applied to areas, such as time series, vision, NLP, public policy, and medicine.



AMIN SHAHRAKI (Senior Member, IEEE) was born in Mashhad, Iran, in 1988. He received the bachelor's degree in computer software engineering and the master's degree in computer networks, in 2009 and 2012, respectively, and the Ph.D. degree from the University of Oslo, in 2020. He is currently a Research Scientist of the industrial IoT with ABB, Germany. His current research interests include the Internet of Things, machine learning for network management, cognitive communications and networking, network behavior analysis, time series analysis, clustering, QoS, and self-healing networks. He has been a member of the National Elite Foundation of Iran since 2011.

AMIR TAHERKORDI (Senior Member, IEEE) received the B.Sc. degree in computer engineering, the M.Sc. degree in information technology engineering, and the Ph.D. degree from the University of Oslo, advised by Prof. Frank Eliassen, with a thesis on adaptive models for wireless sensor networks. He is currently a Professor of computer science with the Networks and Distributed Systems Group, Department of Informatics, University of Oslo. His research interests include resource-efficiency, scalability, adaptability, and dependability in software systems for emerging technologies, such as the IoT, fog/edge/cloud computing, and cyber-physical systems. He also explore AI-based data modeling for sustainability models, such as the circular economy.

JAVIER PRIETO (Senior Member, IEEE) received the Telecommunication Engineering degree, the degree in marketing research, and the Ph.D. degree in information and communication technologies from the University of Valladolid, in 2008, 2010, and 2012, respectively. He has worked at various public and private research centers, including CEDETEL, University of Valladolid, and MIT, as a Visiting Researcher. He is currently an Associate Professor with the University of Salamanca and a member with the Institute of Biomedical Research of Salamanca. Since 2023, he has been serving as the Chair for the IEEE Spain Signal Processing and Communications Joint Chapter. He was an Associate Editor of the IEEE Communications Letters (2016–2024) and the coordinator of the IEEE Spain Blockchain Local Group (2020–2024).

JUAN M. CORCHADO received the Ph.D. degree in computer science from the University of Salamanca, and the Ph.D. degree in artificial intelligence from the University of the West of Scotland. From 2013 to 2017, he was Vice-Rector for Research and the Director of the Science Park, University of Salamanca. He was also twice elected as the Dean of the Faculty of Sciences. Since 2000, he has been leading the Bioinformatics, Intelligent Systems, and Educational Technology (BISITE) Research Group. He is currently a Professor with the University of Salamanca.