ME-IDS: AN ENSEMBLE TRANSFER LEARNING FRAMEWORK BASED ON MISCLASSIFIED SAMPLES FOR INTRUSION DETECTION SYSTEMS

by

Arka Ghosh

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
November 2023

To my beloved parents,

this thesis is dedicated to your unwavering support and encouragement throughout my educational journey. Your firm belief in my decisions, even when I may have doubts, has always been a tremendous source of motivation to pursue my goals with determination.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

IoT ..... Internet-of-Things

IDS ..... Intrusion Detection Systems

ML ..... Machine Learning

DL ..... Deep Learning

TL ..... Transfer Learning

CNN ..... Convolutional Neural Network

LSTM ..... Long Short Term Memory

RNN ..... Recurrent Neural Network

SA ..... Simulated Annealing

TPE ..... Tree Parzen Estimator

RS ..... Random Search

XAI ..... Explainable Artificial Intelligence

GAN ..... Generative Adversarial Networks

SHAP ..... Shapley Additive Explanations

LIME ..... Local Interpretable Model-agnostic Explanations

PDP ..... Partial Dependence Plot

Abstract

In our digitally interconnected world, the demand for robust security measures has become increasingly apparent, given the escalating threat of cyberattacks on the Internet. Intrusion Detection Systems (IDS) have emerged as vital safeguards for Internet network infrastructure. Despite the significant advancements in IDS over the past decades, there remains much room for improvement, especially with recent advances in machine learning and deep learning. In this paper, we propose a Misclassified sample based Ensemble transfer learning framework for IDS (ME-IDS) in order to effectively detect malicious intrusions. Technically, ME-IDS employs frequency encoding to handle categorical features and utilizes a feature selection method to mitigate the curse of dimensionality. In addition, it leverages three hyper-parameter-tuned variants of a transfer learning model in its ensemble learning stage, ultimately resulting in high detection accuracy. Our experimental results based on a publicly available IDS dataset, UNSW-NB15, indicate that ME-IDS leads to an impressive accuracy of 99.72%, significantly outperforming the state-of-the-art detection systems.

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor, Dr. Qiang Ye, for his invaluable guidance, mentorship and unwavering support throughout the process of completing this thesis. His commitment to excellence and constructive feedback have been a major instrument in shaping me into a better researcher. I am truly grateful to have had the opportunity to work with and learn from such an exceptional and motivated mentor.

I am deeply thankful to my beloved parents, Ajoy Ghosh and Lucky Ghosh, my sister, Rittwika Ghosh, for showering me with unconditional love and support in every situation, be it during moments of joy or in the face of challenges. Their tireless efforts and boundless care are the gifts that I can never truly repay. I am forever grateful for their presence in my life. I am also thankful to my cousins, Sutrishna Nandy and Mithun Mallick, who made me feel at home during the last two years in Canada while I was far away from my own. Lastly, I extend my sincere gratitude to my fellow lab mates, whose timely comments and valuable insights have served as a motivating force in making various improvements and refinements in this thesis

Chapter 1

Introduction

The emergence of the Internet has facilitated extensive connectivity among a myriad of conventional computing devices, spanning from servers to clients. However, in recent years, the landscape has evolved with the widespread integration of Internet-of-Things (IoT) devices, ranging from gaming consoles to smartwatches. This influx of IoT devices has notably escalated both the quantity and variety of traffic traversing over the Internet, posing challenges to the security paradigm for IoT devices. Despite the security breaches associated with IoT, it has gained widespread acceptance and integration across a multitude of sectors, encompassing critical domains such as healthcare, agriculture, transportation, energy grids, and many more, signifying its profound impact on diverse industries. IoT is composed of numerous devices, which have limited storage, computing capabilities, and communication abilities [2]. These devices are equipped with a wide array of sensors and actuators to collect and share sensitive data over the traditional internet [4]. Projections indicate that the IoT market is poised to generate substantial revenue, starting at $2 billion in 2020 and expected to reach $8.131 trillion by 2030 [10]. This exponential potential for growth has attracted the attention of various stakeholders including suppliers, corporations, manufacturers, to invest in this revolutionary technology [2, 15].

The interconnection of devices and the rapid transmission of sensitive data across IoT devices have elevated security to a paramount concern, primarily due to the continuously growing number of diverse network anomalies over the internet [3]. In the realm of computer security, a variety of methods, including firewalls, access control, antivirus software and network segmentation, are previously used to enhance the security and control cyber-attacks over internet due to their ability to filter content, prevent data leakage, and raise alerts to thwart malicious activities [41]. However, a limitation of these techniques is their occasional inability to identify novel and intricate types of attacks, as well as their incapacity to distinguish between legitimate and

harmful network traffic. To address this, Intrusion Detection Systems (IDS) can offer an effective security solution by continuously monitoring traffic and identifying novel anomalies at the entry points of IoT devices. IDS is designed to scrutinize network flows, utilizing features to distinguish between legitimate and malicious traffic flows with a strong emphasis on high accuracy and minimal false positive rate. While the concept of IDS was first coined in 1980 [21], many IDS frameworks have continuously evolved to meet the demands of ever-changing threats over the internet. Nevertheless, the substantial technological advancements have led to an increase in network scale, interconnect devices and data volume which creates the ever-growing necessity of the developing existing IDS systems to cope with these advancements. The detection methodology of IDS can be categorized into three sections, as illustrated in Figure 1.1.

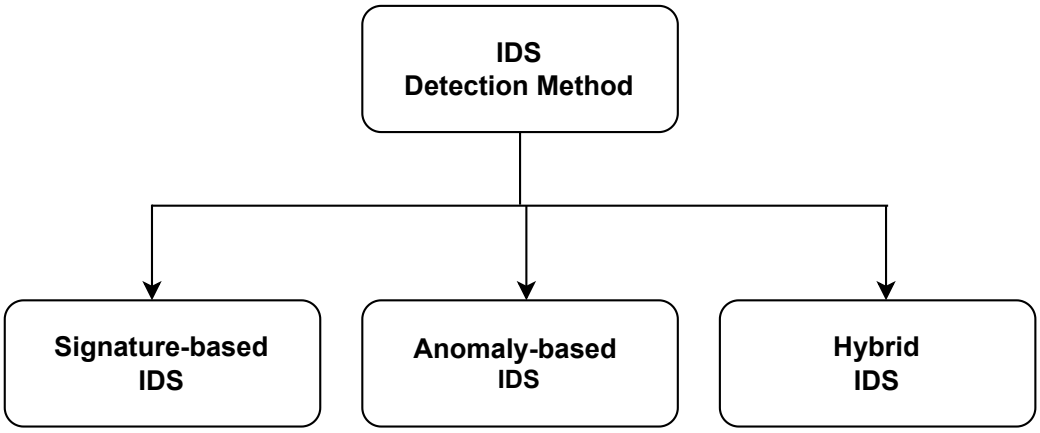

Figure 1.1: Taxonomy Chart of IDS Detection Methodology.

The first generation IDS are mostly signature-based [50]. Signature-based IDS involves searching for pre-defined patterns or signature associated with known attacks. These patterns may encompass packet content, source IP addresses, or distinct headers. When the IDS identifies an attack with a matching signature from its repository, it raises an alert. However, signature based IDSs necessitate regular updates to address the newly arisen threats since it rely on pre-existing signature of the attack patterns [81]. Any outdated information can hinder these IDSs from detecting emerging attacking, including zero-day attacks. However, anomaly-based IDS has gained the attention of numerous researchers because of their potential to overcome the limitations of signature-based IDS [36]. Anomaly-based IDS creates a reference model for

a computer system's typical operations using machine learning, statistical analysis or knowledge-based approaches [14]. Whenever there is a substantial deviation between the observed behavior and this model, the flow is marked as an anomaly, potentially indicating an intrusion. A key advantage of anomaly-based IDS is its capability to identify zero-day attacks, as it is not dependent on a signature database to spot unusual user actions [8]. On the other hand, hybrid IDS is designed by combining both signature-based and anomaly-based IDS to detect both known and unknown attacks [37]. In such scenario, two IDSs operate concurrently, and the ultimate decision of the hybrid system is determined by assigning a weight to each IDS's output [17]. While hybrid IDSs offer clear benefits over signature-based and anomaly-based IDSs, their construction demands substantial computation resource and effort as ensuring efficient interoperability between two IDS setups is comparatively a challenging task. As a result, within the research community, there is still a predominant emphasis on the study of anomaly based IDS.

The rapid and notable progress and expansion of various Machine Learning (ML) and Deep Learning (DL) frameworks have ushered in a multitude of different ML and DL-based IDS, effectively showing impressive results in different classification tasks. This transformative wave has extended its reach into the realm of cyber-security as well, encompassing fields like intrusion and malware detection. Notably, in the context of IDS, various ML and DL based IDS have become an integral part in the establishment of anomaly-based IDS due to their inherent simplicity, straightforwardness, efficiency and impressive performance. As these frameworks continue to advance, they bring new possibilities and capabilities by enhancing and opening new innovative doors to further innovations in cyber-security.

1.1 Motivation

Many prior researches on IDS have predominantly focused on utilizing machine learning (ML) techniques. Due to the considerable diversity in normal network traffic, ML-based IDS face challenges in establishing consistent definitions of normal flows, resulting into lower accuracy on unforeseen network flows [84]. Moreover, ML-based IDS require manual feature engineering and extraction, which can be challenging in terms of both time and expertise required to identify relevant features for the

different kinds of network flows. Deep Learning (DL) methods have addressed the limitations of ML by identifying unusual patterns, thus resulting in better detection accuracy [24]. DL-based IDS excel when a significant amount of training dataset is available. However, within IoT environments, there is a notable lack of substantial labeled datasets, particularly when it comes to the discovering unknown attack types, i.e., zero day attacks [66]. Acquiring new data in such situation can be a both time-consuming and resource-intensive process, and it may not be accessible at all in some cases. Furthermore, if a new intrusion is discovered, the process of re-training a DL model from scratch involves a significant investment in both computational resources and time. As a consequent, DL-based IDSs face tremendous hurdles in IoT networks, where datasets are limited, often imbalanced, and the devices possess constrained computational capabilities.

The emergence of Transfer Learning (TL) addresses the problem of DL, providing a powerful solution for some of the potential challenges involved in training deep neural networks from scratch [66]. Fundamentally, TL enables the utilization of leveraging the knowledge and expertise gained from previously trained models to improve performance on novel and varied tasks. This approach has shown significant potential in alleviating the resource intensive and data-hungry nature of deep learning, making it more accessible and efficient for a wide range of applications in real world. Some of the famous pre-trained models like VGG16, VGG19, InceptionV3, Resnet, and Xception have consistently produced remarkable results in the area of computer vision (CV) over the years. Another notable technique in classification tasks is the ensemble of classifiers, referred to as an ensemble learning, which has gathered significant attention in cyber-security domain, and the field of IDS is no different [76]. Ensemble learning utilizes the strength of several classifiers to make a more robust classifier, which often outperforms a single classifier in terms of predictive accuracy. In the purview of IDS, ensemble of multiple classifiers has demonstrated superior performance compared to individual classifiers due to statistical, computational and representational considerations [60].

Although transfer learning from pre-trained Convolutional Neural Network (CNN)

based models is a well-established approach, its application in network intrusion detection, especially when incorporated with ensemble learning, remains largely unexplored. Inspired by these findings, the objective of this paper is to establish a framework for IDS that combines the principles of transfer learning and weighted ensemble learning to create a robust IDS system capable of effectively differentiating attack flows from normal flows. The weight of the ensemble stage within the proposed framework is dependent on the performance of the base classifiers, particularly the total number of samples misclassified by each base classifier.

1.2 Thesis Contribution

In this thesis, we proposed a Misclassified sample based Ensemble transfer learning framework for IDS (ME-IDS), designed to detect attack and normal network flows. It leverages a lightweight pre-trained transfer learning model in combination with a weighted ensemble learning approach. The issue of the curse of dimensionality is also addressed through the use of a validated categorical feature encoding technique and a feature selection method. Additionally, unlike other methods that rely on tabular datasets, this research utilizes an image transformation approach to make the dataset compatible with the requirements of CNN-based pre-trained models. The major contributions of this paper are as follows:

- ME-IDS employs frequency encoding to encode categorical features and incorporates random noise with a minimal mean and standard deviation to mitigate the tied frequency counts among categories.

- ME-IDS utilizes a filter-based feature selection method, using a p-value threshold, to select critical features to enhance model efficiency, and bolster the IDS's resilience to noise and irrelevant data.

- ME-IDS adopts a novel ensemble scheme that is based on three hyper-tuned variants of a lightweight pre-trained TL model. It utilizes misclassified samples to assign appropriate weights to base classifiers.

- Extensive experiments based on the widely adopted dataset, UNSW-NB15 [52], are carried out in our research. Our experimental results indicate that ME-IDS

outperforms the state-of-the-art schemes.

1.3 Thesis Outline

The rest of the paper is organized into four sections. Chapter II offers an exploration of prior notable researches in the fields of ML, DL, and TL as applied to IDS. Moving on to Chapter III, it provides a comprehensive overview of the ME-IDS framework. This includes a detailed description of the dataset used and data pre-processing, the architecture of the pre-trained Convolutional Neural Network (CNN) model, and the precise configurations of hyper-parameters for three optimization techniques. Additionally, this chapter explains the fundamental concepts underpinning the weight assignment process in the weighted ensemble learning approach. Chapter IV conducts an in-depth analysis of the ME-IDS's performance. It employs four distinct evaluation metrics to assess its effectiveness and also presents a comparative analysis, allowing for a comparison with three other benchmark ensemble learning approaches. Lastly, Chapter V serves as the conclusion, summarizing the research and outlining potential directions for future work.

Chapter 2

Related Work

In this chapter, we first explore the utilization of traditional ML-based IDS, including the methods and models employed to detect and mitigate security threats across various IDS dataset. Following this, we shift our focus towards the integration of Neural Networks and other DL frameworks that have shown promising improvement within the IDS domain. In the final section, we discuss several recently published works on TL strategies such as domain adaption and pre-trained models in the context of IDS, offering insights into their performance and potential to improve the accuracy and efficiency of IDS. Due to the fact that ML/DL based IDS are well-established research topics, a large number of research papers have been published over the years. As a result, it is impractical to conduct an extensive review of each and every published work. Instead, we will highlight a number of noteworthy and exemplary works in this discipline.

2.1 Machine Learning Based Intrusion Detection Systems

By employing statistical principles as the underlying framework for constructing mathematical models, machine learning algorithms are primarily designed to extract meaningful insights from provided data samples. This process typically involves two key phases: a training phase and a testing phase, as depicted in Figure 2.1. Over the last two decades, the field of machine learning-based IDS has experienced significant growth in response to the continually evolving nature of various cyber-attacks [33]. In order to address the dynamic landscape of cyber threats, numerous enhancements have been introduced to assist machine learning-based IDS models in adapting to changing attack patterns while minimizing computational complexity. While the long-term goal remains the development of a fully automated and highly intelligent cyber defense system, network and security operators can already derive substantial benefits from the utilization of machine learning-based IDS [62]. While ML-based

IDS come with inherent drawbacks and limitations, they have nonetheless paved the way for numerous innovative research opportunities in the field of intrusion detection over the years. The recognition of these limitations has, in fact, served as a catalyst, motivating researchers to explore and contribute to the improvement of different IDS infrastructures.

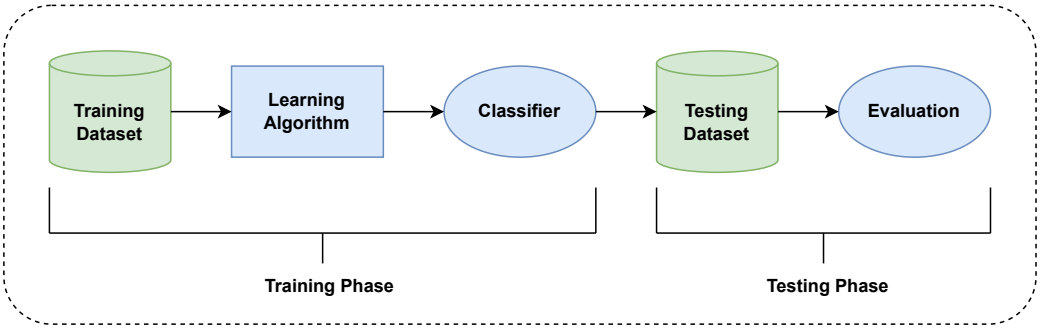

Figure 2.1: Fundamental Learning Process of Machine Learning Algorithms

Moutafa et al. [51] introduced an IDS framework to classify attack families within the UNSW-NB15 dataset. In their initial work, the authors employed Association Rule Mining for feature selection and utilized both Expectation Maximization (EM) and the Naïve Bayes Algorithm for classification. However, the performance of these classifiers was not particularly notable, with Naïve Bayes achieving an accuracy of 78.06% and EM achieving 58.88%. Building upon their earlier research [53], the authors extended their approach by incorporating feature selection techniques based on correlation coefficient and gain ratio. Additionally, they experimented with five different classification algorithms applied to the UNSW-NB15 dataset. Among these classifiers, the Decision Tree (DT) algorithm yielded the most promising results. It achieved an accuracy score of 85%, accompanied by a False Alarm Rate (FAR) of 15.75%. Othman et al. [56] proposed Spark-Chi-SVM model for IDS, involving Chi-square for feature selection and building an IDS using the Support Vector Machine (SVM) on the Apache Spark Big Data platform. They carried out their experiment with KDD99 dataset, and compared their results with two other ML classifier without feature selection. The results indicated that the Spark-Chi-SVM model excels in terms of performance and reduces training time.

Tree-based ML classifiers, particularly Decision Tree and Random Forest, have undergone extensive investigation in the realm of IDS research [6]. Within these

classifiers, the process of partitioning the tree divides the training data into multiple subsets, with each subsequent split progressively augmenting the model's complexity to perform the designated task better [31]. In this context, Al-Omari et al. [6] introduced a tree-based IDS approach based on Decision Tree, with Gini Index to select the important features from the UNSW-NB15 Dataset. During the testing phase, their proposed method attained an overall binary classification accuracy of 96.72%. Another Decision Tree based binary classifier model was proposed in [26], which utilized entropy for feature selection. The authors trained and tested their approach on both the NSL-KDD and CICIDS2017 datasets, where their approach yielded impressive results, achieving an overall accuracy of 99.42% for NSL-KDD and 98.80% for the CICIDS2017 dataset. In addition to that, Chen et al. [16] proposed a method by utilizing Adaptive Synthetic Sampling (ADASYN) to handle the data imbalance problem, coupled with training the IDS using Random Forest. The study, conducted on the CICIDS2017 dataset, demonstrated superior performance of the ADASYN with Random Forest approach compared to Random-Under Sampling (RUS) with Random Forest and Synthetic Oversampling Technique (SMOTE) with Random Forest, achieving the highest precision score of 98.505% and an f1 score of 95.303%. Moreover, another approach is proposed in [87] to address the problem of data imbalance in IDS dataset by combining an enhanced random forest model and synthetic oversampling technique (SMOTE) algorithm. The authors initially augment the minor samples using a hybrid K-means clustering along with SMOTE approach, followed by an enhanced random forest and similarity matrix to refine the prediction results. Their proposed approach achieved a high training accuracy of 99.72%, with 78.47% accuracy on the testing dataset while evaluated on the NSL-KDD dataset. Several other notable ML-based IDS have been proposed over the years, including IDS based on semi-supervised learning utilizing k-Nearest Neighbors (kNN) with hyperparameter optimization [85], a two-phase IDS employing Naïve Bayes with elliptic envelop method [83], an IDS utilizing the Nesterov-Accelerated Adaptive Moment Estimation–Stochastic Gradient Descent (HNADAM-SGD) algorithm [69], and an IDS based on LightGBM with the ADASYN technique [44].

In recent years, ensemble learning is extensively applied in the field of IDS, combining multiple ML models to create a more resilient and effective IDS. Tama et

al. [75] proposed a two stage IDS framework, named TSE-IDS, where the authors employed a hybrid feature selection technique using three methods, namely particle swarm optimization, ant colony algorithm, and genetic algorithm to reduce the feature dimension of training datasets UNSW-NB15 and NSL-KDD. Next, the authors introduced a two-level classifier ensemble utilizing two meta-learners, Rotation Forest and Bagging, which demonstrated an accuracy of 91.27% on the UNSW-NB15 dataset and 85.8% accuracy on the NSL-KDD dataset. Another dual ensemble approach, titled Dual-IDS, is proposed in [46], where the authors combined two existing ensemble learning techniques, namely bagging and gradient boosting decision tree. When evaluating the approach on three datasets, namely NSL-KDD, UNSW-NB15, and HIKARI2021, the proposed method attained an accuracy score of 94.66% and a precision score of 92.21% on the UNSW-NB15 dataset. Thockchom et al. [78] proposed a stacked ensemble learning-based IDS using lightweight ML models, such as Gaussian Naïve Bayes, Logistic Regression, and Decision Tree as the base classifiers, with Stochastic Gradient Descent serving as the meta-classifier. The performance of this proposed ensemble learning approach is trained and evaluated across three IDS datasets, namely KDD Cup 1999, UNSW-NB15, and CICIDS2017. This proposed stacked ensemble learning framework achieved an overall accuracy score of 93.88% for binary classification and 80.96% for multi-class classification on the UNSW-NB15 dataset.

2.2 Deep Learning Frameworks in Intrusion Detection Systems

DL based frameworks have shown notable performance enhancements compared to certain ML algorithms, which is why these frameworks are increasingly applied in different domains including safeguarding IoT infrastructure [29]. In many scenarios, DL frameworks has demonstrated significant improvements by outperforming ML-based frameworks, particularly with extensive labelled data [57]. Furthermore, DL-based algorithm can effectively handle new features, offering enhancing solutions that do not require extensive human intervention. Typically, DL architectures are characterized by their intricate network structures, where numerous hidden layers connect input layer to the final output layer. The hidden layers play a crucial part in extracting features and representing data by enabling the network to extract complicated

patterns and hierarchical features from the input data. Based on the feature learnt and processed by the hidden layers, the final output layer makes the final prediction of outcome of the network. Figure 2.2 depicts a Deep Neural Network architecture that establishes connections from the input layer to the output layer through the integration of multiple hidden layers.

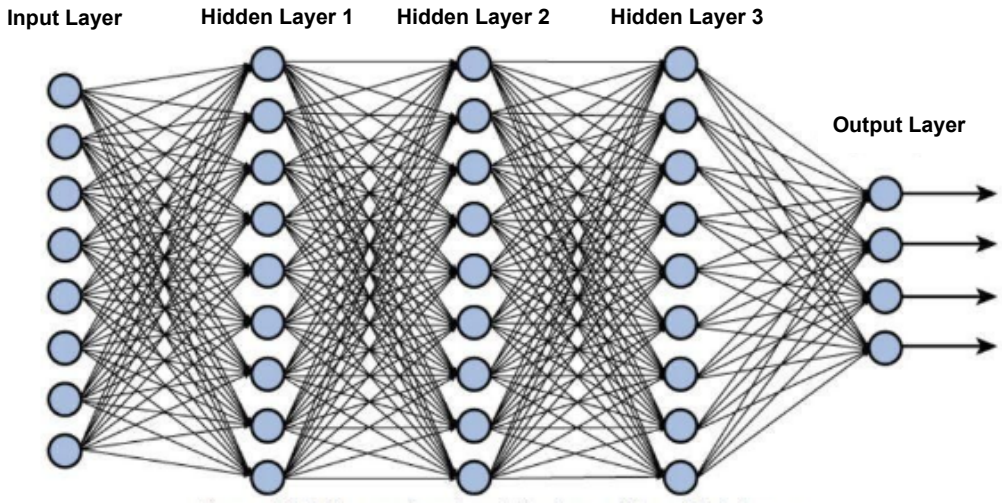

Figure 2.2: Illustration of Generic Deep Neural Network Architecture

Norouzian et al. [54] employed a standard neural network, comprising of a two-layer multi-layer perceptron model with back propagation learning method. Their proposed approach was evaluated on categorizing six different types of attacks using the KDDCUP99 dataset. Another DL-based distributed threat detection system for IoT networks was proposed in [20] where the authors conducted a comparative analysis between their proposed DL model and a shallow model. During performance evaluation, they utilized the NSL-KDD dataset and obtained a precision score of 99.20% and 98.27% for binary and multiclass classification, respectively. Otoum et al. [57] proposed a DL-IDS framework integrating spider money optimization (SMO) algorithm and the stacked-deep polynomial network (SDPN), where SMO being responsible for selecting the most relevant features and SDPN being responsible for categorizing the data into normal or malicious instances. Their approach, when evaluated on NSL-KDD dataset for optimal feature selection of classification of various attack classes, achieved an accuracy of 99.02%. Ravi et al. [63] introduced a method,

which utilized hidden layer features from recurrent models, applied Kernel-based Principal Component Analysis (KPCA) for feature selection, and employed an ensemble meta-classifier for prediction. This approach achieved a remarkable 99% accuracy in detecting network attacks and 97% accuracy in classifying them using the SDN-IoT dataset.

Within the domain of IDS, there is a growing trend of DL-based hybrid frameworks utilizing the fusion of Convolutional Neural Network and Recurrent Neural Network (RNN). These integrated architectures leverage the strengths of CNNs for extracting spatial features and RNNs for sequential information analysis [1]. Halbouni et al. [28] proposed a hybrid CNN-LSTM based IDS that is established around three blocks for feature extraction, with each block consisting of a sequence of CNN layer, Maxpooling, Batch Normalization, followed by LSTM layer and concluded with dropout regularization to reduce overfitting. Following the three feature extraction blocks, their architecture is followed by a fully connected layer employing softmax activation function to make the final prediction. In their evaluation, the authors trained and tested their hybrid architecture on three different dataset, namely CIC-IDS2017, UNSW-NB15, and WSN-DS, and achieved the highest accuracy of 94.53% and 82.41% for binary and multi-class classification, respectively, on the UNSW-NB15 dataset. Another hybrid DL-based IDS named DCNNBiLSTM was proposed in [29], combining 1D CNN and Bidirectional Long Short-Term Memory (BiLSTM). The proposed architecture was investigated for multi-class classification on CICIDS2018 and Edge_IIOT datsets, achieving an impressive accuracy of 100% and 99.64% respectively on both the datasets. Yao et al. [90] introduced an innovative hybrid IDS framework, combining CNN and LSTM architectures through cross-layer feature fusion. In this proposed framework, the CNN component is dedicated to identifying regional features, contributing to the derivation of comprehensive global features. Simultaneously, the LSTM component focuses on capturing periodic features through its memory function. In essence, the proposed hybrid architecture leverages the strengths of CNN for spatial feature extraction and LSTM for capturing temporal dependencies. The CNN-LSTM-based Cross-Layer Feature Fusion approach was evaluated on two benchmark datasets: KDDCup99 and NSL-KDD. Remarkably, the results demonstrated impressive performance with an accuracy score of 99.95%

for the KDDCup99 dataset and 99.75% for the NSL-KDD dataset.

Several of the IDS research studies have incorporated image-based approaches, utilizing the power of CNN for its exceptional feature extraction capabilities, spatial hierarchies, and robust pattern recognition. Al-Turaiki et al. [7] employed a two-step preprocessing step to generate 2D grayscale images from tabular data, which involves Principal Component Analysis (PCA) with feature engineering using Deep Feature Synthesis (DFS). They introduced two CNN models, namely BCNN and MCNN, for binary and multi-class classification, respectively, using five CNN layers followed by four fully connected layers. When evaluated on two IDS datasets including UNSW-NB15, their proposed approach achieved the highest accuracy score of 95.71% and 80.51% for binary and multiclass classification, respectively, on UNSW-NB15 dataset. Several other research studies on image-based IDS utilizing Deep CNNs, as proposed in [35] and [3], have been conducted where the authors utilize network spectrogram images generated from tabular dataset using short-time Fourier transformation. Both of these approaches, when evaluated on two different IDS datasets, demonstrated improved accuracy and reduced false alarm rates compared to other traditional DL-based IDS frameworks. Kim et al. [38] proposed a DL-based IDS framework using multiple image transformer methods that involves several dimensionality reduction algorithms, namely t-Distributed Stochastic Neighbor Embedding (t-SNE), Uniform Manifold Approximation and Projection (UMAP), Principal Component Analysis (PCA), Kernal Principal Component Analysis (kPCA) with Radial Basis Function (RBF), and Kernal Principal Component Analysis (kPCA) with sigmoid to transform the normalized pre-processed tabular data to a 120x120x1 grayscale image first. The proposed scheme creates grayscale images for different algorithms, trains a CNN, and computes the F1-score for training data. The top three algorithms with the highest F1-scores are chosen. Their grayscale images are then mapped to the RGB channels, forming a three-channel RGB image. The authors generated RGB images from ISCXIDS2012 and CSE-CIC-IDS2018 datasets and employed a four-layer dual concatenated CNN model for training and classification with the generated 2D RGB images. Upon evaluation on the testing samples, the proposed IDS framework was able to achieve an overall accuracy of 95.60% on ISCXIDS2012 dataset and an overall accuracy of 95.50% on CSE-CIC-IDS2018 dataset.

2.3 Pre-trained Transfer Learning Models for Intrusion Detection

In transfer learning, when a given source domain D_S and learning task T_S, along with a target domain D_T and learning task T_T, the objective is to enhance the learning of the target predictive function $\mathrm{y_T}$ in D_T by utilizing the knowledge of D_S and T_S, where $D_\mathrm{S} \neq D_\mathrm{T}$ or $T_\mathrm{S} \neq T_\mathrm{T}$ [58]. In various practical scenario, the dimension of D_S is significantly larger than D_T. Transfer learning serves as a strategic approach to enhance the performance of a classification model in specific domains by leveraging knowledge extracted from related yet distinct source domains. The primary goal is to reduce the reliance on extensive amounts of data from the target domain during model development, thereby facilitating learning in new and task-specific contexts. As emphasized in the comprehensive study by Zhuang et al. [93], this strategy proves particularly advantageous when addressing novel tasks or domains where collecting substantial data is challenging. The transfer learning process typically involves migrating a model's architecture and learned weights from a source domain, which boasts a large dataset and ample computational resources, to a target domain characterized by a smaller dataset and limited computational capabilities. Figure 2.3 illustrates the procedure of transferring of knowledge from a resource-rich source domain to a resource-constrained target domain. By leveraging insights gained from the source domain, transfer learning aims to improve the generalization and effectiveness of models, especially in situations where collecting abundant target domain data is impractical or resource-intensive.

Considering the efficiency of DL-based frameworks in precisely addressing security issues in IoT, some researchers have suggested the use of Deep Transfer Learning methods to create an IDS framework leveraging the knowledge of pre-trained model on a source dataset to a target dataset. CNN based pre-trained models have gained recognition for their efficiency in image processing tasks, and to accelerate the performance of these models, network traffic data, which are generally available in numeric and categorical form need, to be transformed into image format. In this regard, Hussain et al. [30] proposed a chunk based methodology to convert network traffic flows into RGB images with the remaining 60 features out of 80 features after data cleaning from the CICDDoS2019 dataset. The authors iteratively utilized 180 data samples from the tabular dataset to form a square shaped RGB image with dimensions of

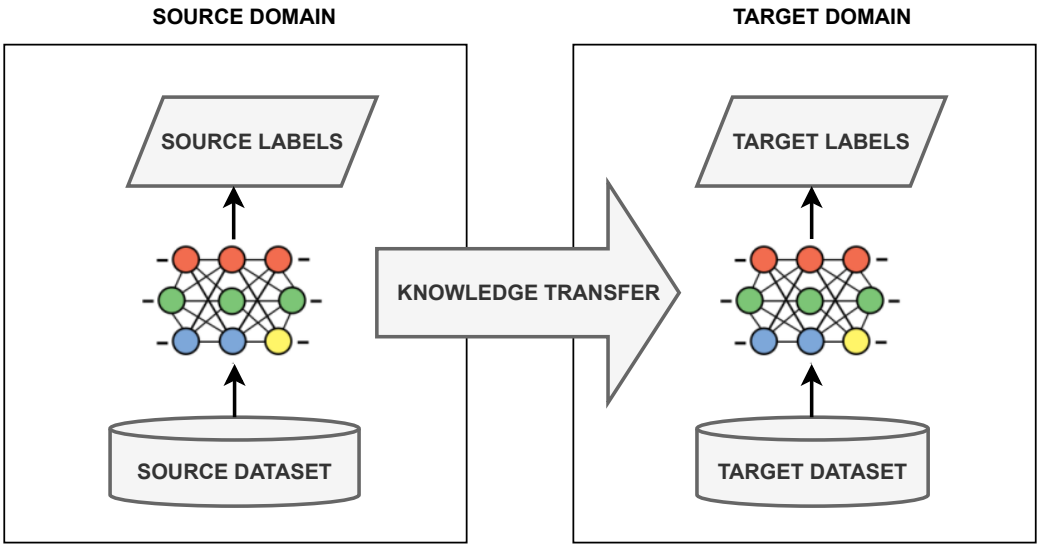

Figure 2.3: Process of Transfer Learning.

60x60x3, with each channel of the image comprising 60 network flow data. They trained a state-of-the-art pre-trained CNN based model named ResNet18 with their converted data, achieving an average precision score of 87% in detection of eleven types of different DoS and DDoS attacks. Masum et al. [48] proposed TL-NID framework based on VGG16 architecture, in which the authors presented an alternative technique for transforming tabular data to image using the NSL-KDD dataset. Upon encoding the categorical features, the number of the feature attributes expanded from 41 to 121. The authors then transformed these attributes into a grayscale image with a 2D array size of 11x11 first. Given that greyscale images consist of just one color channel, a channel augmentation was carried out during the conversion process by replicating the grayscale images to create three-channel RGB images. The TL-NID framework was assessed on the KDDtest+ and KDDTest-21 datasets, resulting in an overall accuracy of 89.30% and 70.97%, respectively.

Haddaji et al. [27] proposed a deep transfer learning based intrusion detection in-vehicle model named TRLID using CAN Bus Protocol for Internet of Vehicle where the authors utilized a data preparation stage to clean the CAN bus dataset and adopted a similar approach as proposed in [30] to transform the network flows of CAN bus dataset into RGB images to meet the data input requirement of the transfer learning model. In this approach, a six layer CNN based model followed by a flattening

and two dense layers was utilized as a source domain for training. In the target domain, the weights of the CNN base layers trained in the source domain were kept frozen, followed by a flattening and four dense layers, among which the last dense layer being responsible for the final classification tasks of the network flows. The authors reported a validation accuracy of 99.97% in the validation phase in the source domain, with a slightly lower validation accuracy of 99.87% in the target domain. Dhillon et al. [19] utilized deep TL techniques where the authors transferred the knowledge gained from a well-sourced source domain to a less-resourced target domain. The authors also integrated tree-based extra-trees classifier into their approach to assess the importance of features, determining a scoring value for each feature to evaluate the signifiance of the feature in the classification of traffic flows. The proposed approach utilized a CNN-LSTM architecture in both the source and target domain, achieving 98.30% accuracy in the source domain and an improved score of 98.43% in the target domain, using the UNSW-NB15 dataset.

Yang et al. [89] introduced IDS framework Internet of Vehicle (IoV) systems, emploting transfer learning and ensemble learning IDS using five different pre-trained CNN-based TL models, including VGG16, VGG19, Xception, Inception, Inception Resnet in conjunction with hyper-parameter optimization techniques. The authors utilized a chunk based approach, similar to the one proposed in [30], to generate the RGB images from the tabular data. Notably, they opted for quantile transformation to normalize the network flow before converting them into RGB images instead of min-max normalization due to its greater resilience to outliers. They proposed two ensemble learning methods, namely Confidence Averaging and Concatenation Ensemble, utilizing the top three best models out of five pre-trained transfer learning models. These approaches achieved high accuracy on both the Car-Hacking and CICIDS2017 datasets. In a related approach outlined in [89], an IDS framework, titled Efficient-Lightweight Ensemble Transfer Learning (ELETL-IDS), is introduced in [55]. In this framework, the authors choose and utilize the top three best-performing models from a selection of five pre-trained CNN-based transfer learning models during the ensemble stage. These models are trained on generated 2D RGB images derived from the tabular form of two IDS datasets. They utilized a confidence averaging ensemble learning method. When evaluating their proposed ensemble transfer learning based

IDS framework on the CIC-IDS2017 and CSE-CIC-IDS-2018 datasets, the ELETL-IDS framework achieved an overall accuracy of 100% on the CIC-IDS-2017 dataset and 99% on the CSE-CIC-IDS2018 dataset.

Although substantial research has been conducted in the field of Intrusion Detection Systems utilizing ML and DL frameworks, there remains significant untapped potential for enhancement. A specific area of opportunity lies in the integration of the principles of transfer learning and ensemble learning. The proposed framework aims to fill this gap by integrating a lightweight, pre-trained transfer learning model with weighted ensemble learning. The weights are optimized based on the performance of each base classifier, enhancing the framework's ability to discriminate between normal and malicious network traffic flows, ultimately creating a more robust and intelligent IDS framework.

Chapter 3

ME-IDS: A Novel Framework For Intrusion Detection

We begin this chapter by providing a comprehensive system overview of the proposed ME-IDS framework, accompanied by a schematic block diagram for clarity of the steps involved in the framework. Subsequently, we dive into the dataset employed within this framework, discussing the process of its generation and summarizing the distribution of data across both the training and testing subset. Our exploration continues with a detailed examination of the data pre-processing techniques applied. We then shift out focus towards the transformation of tabular data into RGB images using a chunk-based method, followed by an in-depth explanation of the pre-trained transfer learning model, discussing its architecture and how its different layers are tailored to align with our specific dataset requirements, supported by a corresponding schematic diagram. We further delve into the three hyper-parameter optimization techniques utilized in the ME-IDS framework, providing comprehensive tables for more detailed presentation of the optimal values for parameters derived from each of hyper-parameter optimization technique. Finally, we explore the weighted ensemble learning approach, shedding light on the methodology behind assigning weights to each base classifier utilized in the framework.

3.1 System Overview

The purpose of this thesis is to provide a comprehensive framework for the development of an intelligent IDS capable of effectively and precisely detecting both normal and malicious network flows. Ensuring the security of network infrastructure is of paramount importance in today's digital age due to ever increasing interconnected devices, and this framework is designed to enhance the accuracy and efficiency of the detection process. The system overview for the proposed ME-IDS framework is illustrated in Figure 3.1, providing a visual representation of the sequential steps involved in building this framework.

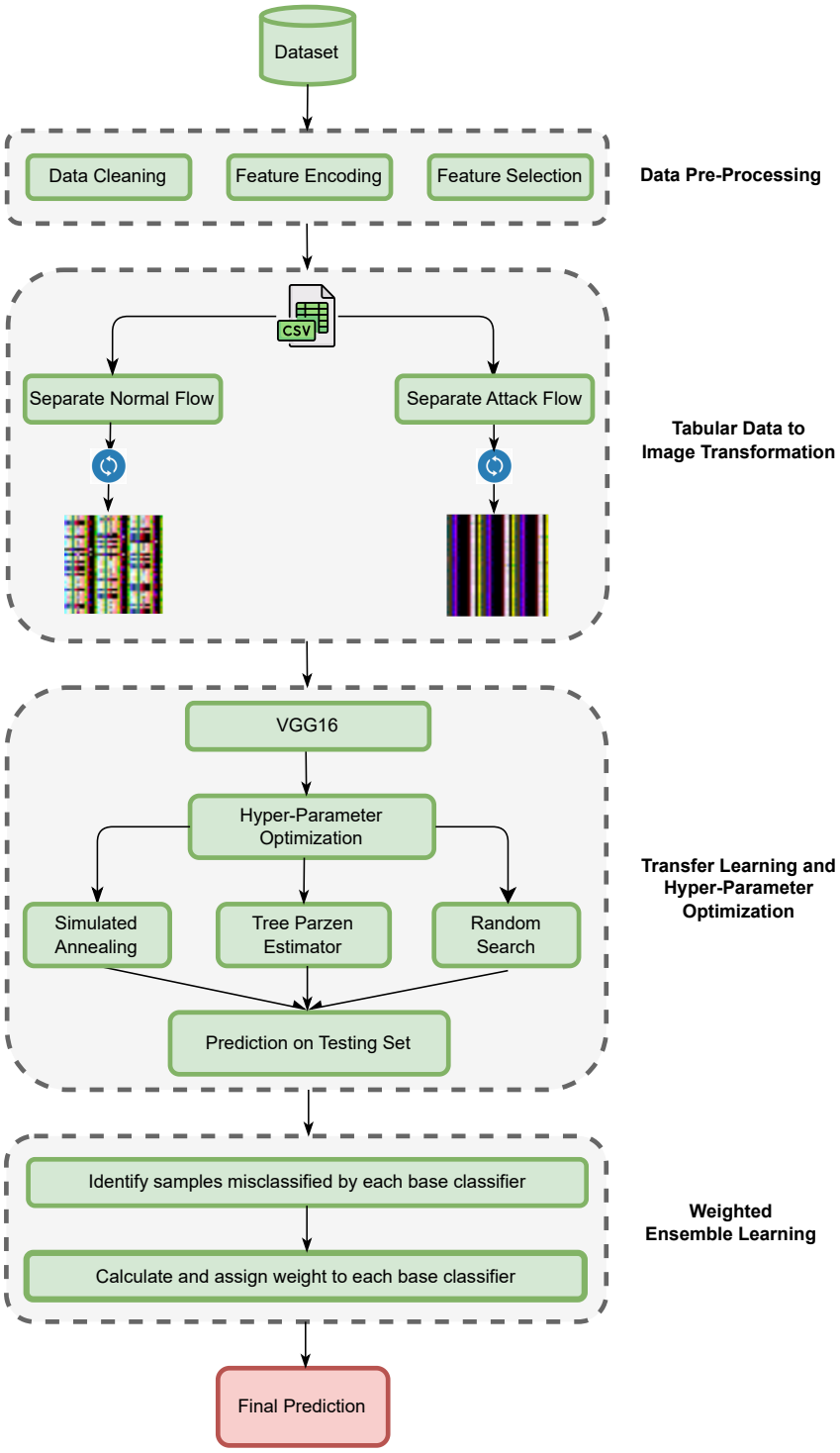

Figure 3.1: System Overview of Proposed ME-IDS Framework.

Following the collection of a real-world IDS dataset, ME-IDS enters a critical data pre-processing phase, which serves three major objectives. This data pre-processing phase commences by eliminating redundant data to streamline the dataset. It then proceeds to encode categorical features to prepare the data in suitable way to be fed into the classifier model. Additionally, the pre-processing stage concludes with a feature selection step, employing a filter-based method to identify and retain the most important and relevant attributes for further analysis. This final step of the pre-processing stage effectively and strategically reduces the dimensionality of the feature set. Considering the well-established proficiency of CNN-based Transfer Learning models in handling image data, ME-IDS undergoes a transformation phase at the next stage to transform the pre-processed tabular dataset into squared shape RGB images. This RGB image transformation from tabular data is executed through a chunk-based method, involving the segmentation of the tabular data into distinct chunks that are subsequently converted into corresponding image segments. Additionally, the transformation incorporates a quantile transformation, adding a layer of refinement to the dataset representation. At the next stage, a lightweight CNN based TL model is fine-tuned using three different hyper-parameter optimization techniques, resulting in three model variants. These three fine-tuned variants are then trained using the transformed images from the previous stage and subsequently used to make predictions on the testing dataset after the training phase. Following this, a weighted ensemble learning approach is constructed, bringing together the predictions of the three fine-tuned versions of the CNN-based TL models. The structure of this ensemble model is designed in such a way that the classifier with the superior performance, in terms of minimizing misclassifications during the testing phase, is given greater significance by assigning more weight. Upon the establishment of the ensemble, the ultimate prediction is executed using these optimized weights. This weighted ensemble learning stage of the proposed ME-IDS framework leverages the combined knowledge and predictive capabilities of the individual based classifiers to detect both malicious and normal network flows with an elevated level of precision, reliability, and intricate understanding.

3.2 Dataset Description: UNSW-NB15

3.2.1 Dataset Generation

In this work, we used a publicly available dataset named UNSW-NB15, curated through a combination of live network traffic, encompassing both regular and attack patterns at Cyber Range Lab of Australian Centre for Cyber Security (ACCS). This dataset was generated utilizing the IXIA Perfect Storm tool, which includes a collection of both new attack types and publicly known security patterns and exposures listed in the Common Vulnerabilities and Exposure (CVE) database. Three virtual servers were configured with IXIA traffic generator, with servers 1 and 3 for normal traffic distribution and server 2 designed for generating abnormal or malicious network activities, as shown in Figure 3.2. Tcpdump tool was employed to capture network packet traces, involving hours of compilation for 100 GB of data divided into 1000 MB pcap files, from which features were extracted using Argus and Bro-IDS on a Linux system.

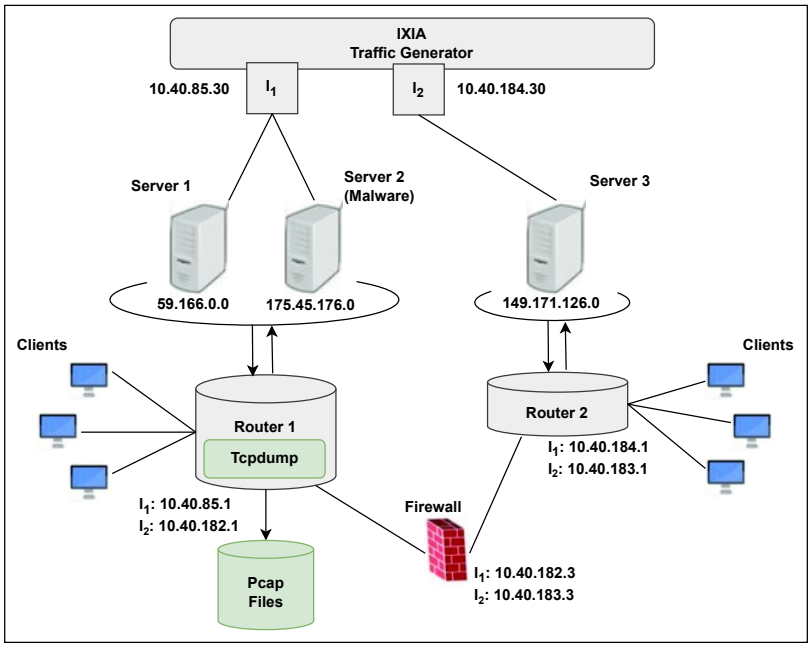

Figure 3.2: Testbed Architecture for Generating UNSW-NB15 Dataset [52].

3.2.2 Dataset Composition

The UNSW-NB15 dataset contains modern network traffic compared to other publicly available IDS dataset, encompassing both normal and malicious attack flows, which also include present-day low footprint attacks [40]. The creation of this dataset is driven by the shortcomings of the KDDCUP 99 [77] and NSLKDD [77] datasets, including the issues of significant presence of redundant records in training set, existence of numerous missing records, and the dataset's inadequacy in offering a comprehensive representation of modern low foot print attack environment [52, 82].

The UNSW-NB15 dataset, comprising of 2,57,673 records, has been partitioned into a training and testing set, with 1,75,341 records in the training set and 82,332 records in the testing set. Each individual record within the UNSW-NB15 dataset is characterized by a total of 43 features, including 40 numeric and 3 categorical features. Additionally, the dataset includes two distinct labels as feature attribute for classification purpose. The first label comprises 9 distinct types of attack classes, each class representing a specific type of modern day malicious attack, alongside a class for normal network flow. The other label feature is binary, '0' denotes instances classified as normal, while '1' signifies instances identified as the malicious network flow. As the major purpose of IDS is to differentiate attack flows from normal flows, we have restricted our work to binary classification only. Additionally, the distribution records between the attack and normal classes in UNSW-NB15 dataset is relatively more balanced compared to other NIDS datasets such as KDD99, NSL-KDD [5]. The distribution of the normal and attack flows in the training and testing subset of UNSW-NB15 dataset has been illustrated in Table 3.1.

Table 3.1: Data Distribution of UNSW-NB15 Dataset

Network Type	**Training Set**		**Testing Set**	
	Number of Records	**Percentage (%)**	**Number of Records**	**Percentage (%)**
Normal Flow	56,000	31.94	37,000	44.94
Attack Flow	119,341	68.06	45,332	55.06

3.3 Data Pre-Processing

3.3.1 Data Cleaning

The data cleaning phase commenced with a comprehensive analysis of both the training and testing subset to identify and address data instances of missing or malformed data. In this initial step, we systemically examined which samples contained missing values or inadequate entries such as NaN. It is important to analyze the presence and handle missing or 'nan' values within a dataset as these values don't provide any meaningful contribution to the prediction process. Notably, in contrast to other intrusion detection datasets, it is observed that there are no such instances of missing or 'nan' values present both in the training and testing subset of UNSW NB15 dataset.

3.3.2 Categorical Feature Encoding

In almost all the real-time datasets, categorical attributes are often prevalent which generally exhibit high cardinality [18]. Since many machine learning and deep learning algorithms necessitate features in numerical form, it is crucial to convert the categorical features into numerical form before feeding these features into any classification algorithm. Among various categorical feature encoding techniques, one-hot encoding and label encoding are widely employed, although both of the methods come with their associated drawbacks. One-hot encoding represents each category with a binary vector indicating presence (1) or absence (0) of the feature's value [80], while label encoding assigns an arbitrary numeric integer to each categorical feature categories [39]. One-hot encoding tends to lead to high dimensionality and increased computational complexity, while label encoding introduces unintended ordinal relationships among categories, potentially impacting the model's performance [39]. This is why we have adapted the concept of frequency encoding that is used to replace each attribute of the categorical features with the total frequency or count [59] within the dataset. Figure 3.3 illustrates an example of a categorical feature encoding where each category is replaced by the number of times it occurs in that feature column. The main problem with frequency encoding arises when there exists two or more categories with the same frequency as this leads to no distinction between

those categories. To address this issue, a random noise using a normal distribution with mean 0 and standard deviation of 0.01 is added to the categorical features to mitigate the issues arising from tied frequency counts between two or more categories.

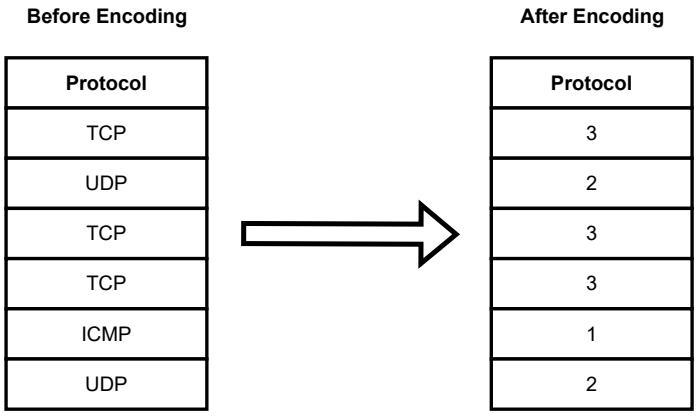

Figure 3.3: Illustration of Frequency Encoding for Categorical Feature.

3.3.3 Feature Selection

Tackling the challenge of high-dimensional data in most of the real-world IDS datasets, often associated with curse of dimensionality, is crucial task in developing a precise predictive model [61]. Effective feature selection methods can enhance the efficiency of learning models, improve predictive accuracy and simultaneously reduce the complexity of both high-dimensionality and training complexity. The feature selection techniques commonly used in classification tasks can be categorized into three groups, namely filters, wrappers and embedded methods. Among these, wrapper and embedded methods are classifier dependent and are relatively slower, making them more susceptible to overfitting compared to filer methods [61], which is why filter-based method is employed to eliminate the redundant features. Before applying the feature selection methods on UNSW-NB15 dataset, we have dropped one feature "id" which represents the unique identifier for each record in the dataset and does not provide any meaningful insight. In filter based methods, features are evaluated through statistical tests assess their correlation with the class, and those score below a specific threshold are subsequently eliminated. Among the filter based methods, we have selected chi-square test that quantifies the disparity between the observed frequency in

each category and the anticipated frequency as per the null hypothesis, which posits no association between the variables. The chi-square statistic can be calculated by:

$$x^2 = \sum \frac{(O_\mathrm{i} - E_\mathrm{i})^2}{E_\mathrm{i}} \tag{3.1}$$

where O_i is the number of observed value of a class and E_i in the number of expected values. After computing the chi-square statistic for each feature, the p-value can be determined by employing the chi-square distribution with degrees of freedom which signifies the likelihood the encountering a chi-sqaure statistic as extreme as the computed one, assuming no association between the variables. Features characterized by lower p-values are deemed to exhibit a strong association with the target variable and are chosen for modeling. In this method, we have a chosen a significance threshold of p-value ≤ 0.05. The p-value threshold of 0.05 serves as a benchmark and has been incorporated into diverse scientific disciplines, thus motivating its selection within the proposed framework. Among the computed p-values for the features of UNSW-NB15 dataset illustrated in Figure 3.4, three features, namely 'ackdat', 'trans_depth' and'sloss', exceeded this threshold, resulting in their exclusion from the feature set. Consequently, the remaining 39 features are retained for further analysis.

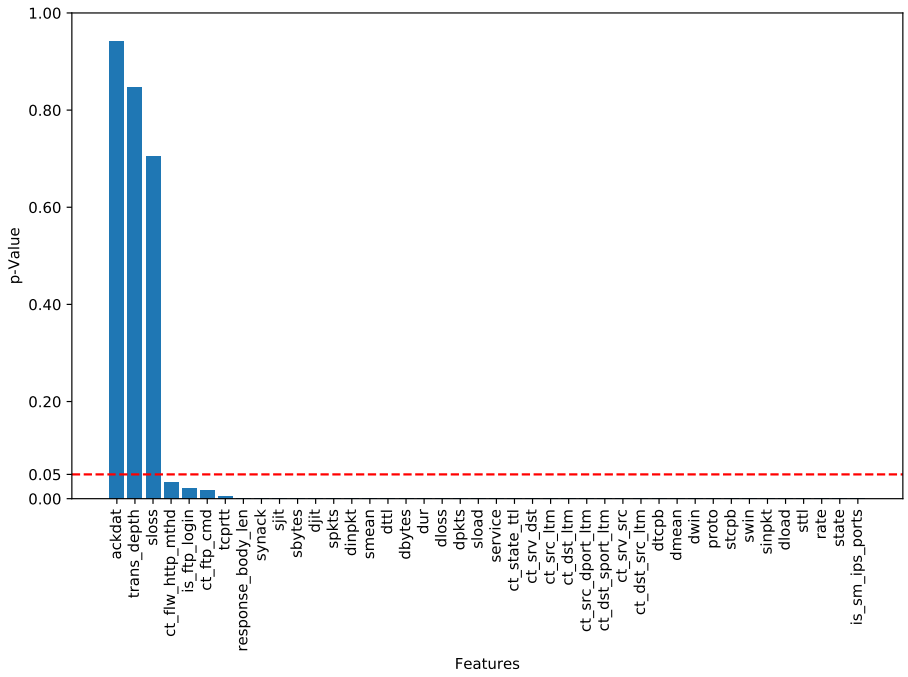

Figure 3.4: Feature Significance Analysis Using Chi-Square p-Value.

3.4 Tabular Data to Image Transformation

The process of transforming tabular data into images is a crucial step in adapting the network traffic data to the requirements of pre-trained TL model which begins with the separation of normal and attack flows into two distinct data frames. To keep the data features on a consistent scale, enabling the fair comparison and meaningful interpretation of data, feature scaling is required. Min-max and standardization are two of most common scaling methods used to normalize the features. While both min-max normalization and standardization are sensitive to the presence of outliers [45], and standardization may transform negative pixel values that are typically not suitable for image representation, quantile transformation is employed in this research which is used to transform the feature's distribution into a specific uniform distribution. This transformation involves remapping the original data values to new feature values that align with specific quantiles within the desired uniform distribution. Notably, quantile transformation is also considered to be an effective transformation strategy that enhances robustness against outliers [64].

As 39 important features has been determined through chi-square p value, a chunk of 39 x 3 = 117 rows from the tabular data has been selected iteratively to form a square image after data normalization. The transformation process involves arranging the first 39 feature rows of each chunk into an image matrix for channel 1, the next 39 feature rows into an image matrix for channel 2, and the final 39 samples into an image matrix for channel 3. These channel-specific matrices are then combined and converted into a 39x39x3 RGB image using the OpenCV library which is a well-known python library for handling and manipulating image dataset. Afterwards, labels are assigned for each RGB images to specify whether they belong to the normal class or malicious/attack class. This process is adhered to for both training and testing dataset until all the data samples are transformed into images. In Figure 3.5, a singular representation from both the normal and attack classes is depicted, revealing pronounced disparities in the feature patterns distinguishing the normal from the malicious class. The visual illustration underscores substantial variations, highlighting the noticeable differences between the two classes in terms of their characteristic feature patterns.

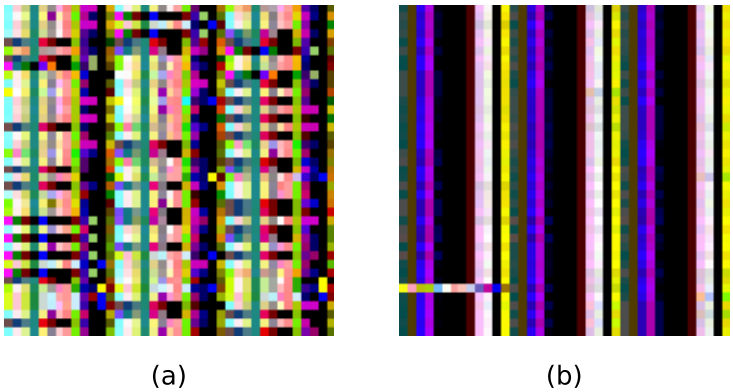

Figure 3.5: Generated image samples from tabular data: (a) Normal Flow; (b) Attack Flow.

3.5 Transfer Learning

CNN-based transfer learning architecture has become intensively used in image classification tasks [72] that involves the method of adapting a pre-trained model, originally trained on a large dataset, to a new task by utilizing the model's learned knowledge and features instead of training the neural network from scratch [91]. The lower layers of a CNN based pre-trained models typically capture general patterns applicable across various tasks enabling the lower layers to be readily applied to different tasks, while the upper layers specialize in dataset-specific features. To enhance the performance of CNN based transfer learning approaches, most of the lower layers are kept fixed with unchanged weights, but a subset of the top layers is adjusted using hyper parameter tuning to help in adapting the model to a new dataset [43].

Within the ME-IDS framework, VGG16 [70] is used as base architecture, chosen for its advantageous blend of relative lightweight characteristics compared to other CNN-based transfer learning architectures. This choice is underpinned by the compelling feature of VGG16, which not only exhibits a streamlined design but also showcases remarkable accuracy across diverse image classification tasks. The architectural composition of VGG16 comprises a comprehensive total of 16 layers, encompassing 13 convolutional layers seamlessly followed by 3 additional fully connected layers. The convolutional layers are organized into five blocks, with the first two blocks having two convolution layers each with 64 filters of size 3x3 in the first block and 128 filters of size 3x3 in the second block. Both of these blocks concluded with a max pooling

layer of size 2x2. In the third CNN block, there are three convolutional layers with 256 filters of size 3x3 in each layer, followed by a max-pooling layer of size 2x2. The fourth and fifth CNN blocks are similar, with each having a number of three convolutional layers in each block with 512 filters of size 3x3. Both blocks conclude with a max-pooling layer of size 2x2. The initial evaluation of the VGG16 dataset has been performed on the ImageNet dataset which is a large-scale image dataset consisting of millions of labelled image data. This large-scale imagery dataset serves as a robust benchmark for assessing the capabilities of the VGG16 architecture, allowing for a comprehensive examination of its proficiency in handling diverse and intricate visual information. In our proposed methodology, we have used the five initial feature extractor CNN blocks of VGG16 architecture. Following these CNN blocks, we have integrated a flattening layer, followed by a fully connected layer with dropout regularization technique. The final layer in our architecture is a sigmoid activation layer with single neuron, tailored specially for the binary classification task. As VGG16 has an input dimension of 224x224x3, the RGB images generated in Section III-C are resized to 224x224x3 from 39x39x3 to match the model's requirement. To enhance adaptability for the UNSW-NB15 dataset, the weights of the fifth CNN block are kept unfroze and three different hyper-parameter tuning techniques are employed to find the optimal number of frozen layer, fully connected layer's filter size, and dropout percentage.

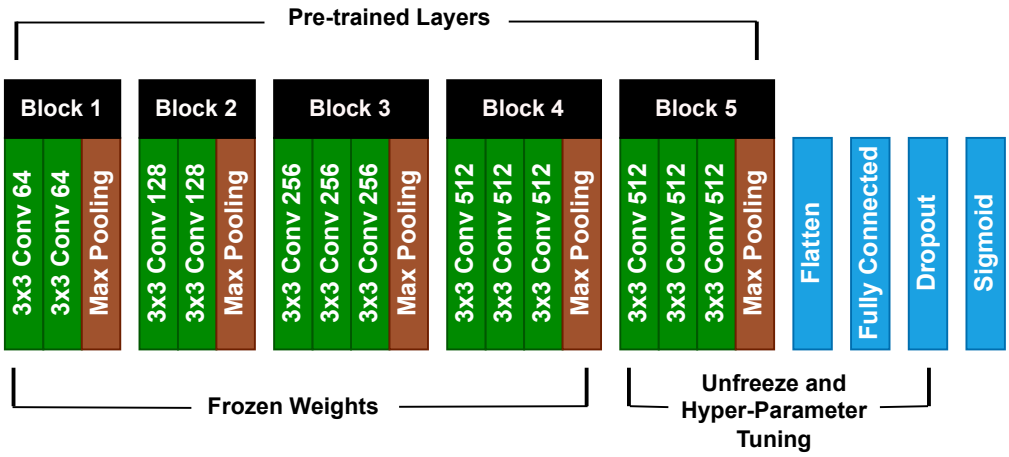

Figure 3.6: Proposed VGG16 Architecture adapting to the UNSW-NB15 dataset.

3.6 Hyper-Parameter Optimization

Hyper-parameter optimization in DL aims to determine the optimal combination of parameters for a neural network to increase its predictive performance on a particular task. Hyper-parameters are parameters that are not learnt during the training process but are set with specific search space prior to the training [9]. Hyperparameter optimization involves discovering a set of hyperparameters that leads to an optimal model, minimizing a predetermined loss function and consequently enhancing accuracy on independent datasets. It is a crucial phase in both the model development and training, as the selection of hyperparameters can have a direct influence on the performance of the model. The optimization of hyperparameters can be denoted in equation form as:

$$x^* = \underset{x \in X}{\operatorname{argmin}} f(x) \tag{3.2}$$

The objective function, denoted as *f(x)* represents a measure to minimize, such as the error rate assessed on the validation dataset. x^* signifies the set of hyper-parameters that leads to the lowest score and x can be any value within a specific search range within this domain. In simpler terms, the goal is to find the optimal values of the hyperp-parameters that result in the highest performance score on the validation dataset.

For CNN-based transfer learning models, hyper-parameter tuning is critical as it enables the optimization of the key parameters by finding the right combination that ensures the model's adaptability to specific task and dataset, thus maximizing its predictive capabilities. The hyper-parameter chosen for fine-tuning in our framework can be divided into two major categories. The first category encompasses parameters related to the model's architecture, which includes number of frozen layers, filter numbers in fully connected layers, and the dropout rate. The second category pertains to parameters involved in the training process, such as learning rate, patience for early stopping, and the number of epochs for training. In the ME-IDS framework, three different hyper-parameter optimization techniques, namely Simulated Annealing, Tree Parzen Estimator (TPE), and Random Search, are utilized to find the optimal values for the above-mentioned hyper-parameters used in building and training the VGG16 model.

3.6.1 Simulated Annealing

Simulated Annealing (SA) is built on the principles of metallurgical annealing [65], adapting the process to optimize complex systems by employing a probabilistic exploration of solution spaces to escape local optima. The algorithm initiates by randomly selecting a set of hyper-parameters within the specified range of values provided by the user. The quantitative key of this optimization method is Boltzamann distribution that gives the probability of any actual state of x

$$p(x) = e^{\frac{-\Delta f(x)}{kT}} \tag{3.3}$$

where $f(x)$ represents the energy configuration, while k stands for Boltzmann's constant and T denotes temperature. SA optimization starts with a randomly chosen initial solution, x_0, and assesses its objective function. The algorithm then sets the temperature parameter, T, which influences convergence speed. It iteratively explores neighboring solutions, accepting them probabilistically based on temperature and gradually reduces the temperature to consider deteriorating moves when it is high.

3.6.2 Tree Parzen Estimator

Tree Parzen Estimator (TPE) employs a tree-structured technique that uses Bayes' rule to handle hyper-parameters, modeling $p(x|y)$ (x represents the suggested hyper-parameter set, while y corresponds to the actual value of the objective functions using hyper-parameters x) and $p(y)$ instead of directly representing $p(y|x)$ [11]. TPE does not specify a predictive distribution but sets up two hierarchical processes, $l(x)$ and $g(x)$, which act as generative models for all domain variables, characterizing their performance when the objective function is either below or above a specified quantile y^* which can be denoted by:

$$p(x|y) = \begin{cases} l(x) & \text{if } y < y^* \\ g(x) & \text{if } y \geq y^* \end{cases} \tag{3.4}$$

Thus, it selects a quantile γ of observed y values, where γ is larger than the best observed $f(x)$, to determine y^*, enabling the algorithm to scale linearly in the size of observed variables and the number of dimensions being optimized without requiring a

specific model for $p(y)$. As TPE is structured in a tree-like arrangement, this enables to preserve the specified conditional dependencies by accommodating the specified conditional hyper-parameters, leading to a time complexity of $O(n \log n)$ [88].

3.6.3 Random Search

Random Search (RS), introduced in [12], was proposed as an improvement over Grid Search. Unlike Grid Search, which evaluates all values within a search space, RS takes a different approach. It randomly picks a predetermined number of samples from the range defined by upper and lower bounds to serve as potential hyper-parameter values. Subsequently, it trains these selected candidates until the allocated budget is used up. The underlying concept of RS relies on the idea that when the configuration space is sufficiently extensive, it becomes possible to identify global optima, or at least close approximations to them. Even with a constrained budget, RS manages to explore a more extensive search space compared to Grid Search [12]. Given that the total number of evaluations in RS is predetermined as n prior to commencing the optimization process, the computational complexity of RS is expressed as $O(n)$ [86].

Tables 3.2 and 3.3 provide a concise overview of hyperparameter configurations for VGG16 on image datasets generated from 39 and 42 features, respectively. These tables encompass parameters, their search spaces with step sizes, and the optimal values achieved through the application of three optimization methods: Simulated Annealing (SA), Tree-structured Parzen Estimators (TPE), and Random Search (RS).

Table 3.2: Hyper-parameter configuration space for VGG16 on image dataset generated using 39 features

Hyper-Parameter	**Search Space**	**Step Size**	**Optimal Value**		
			SA	**TPE**	**RS**
Frozen Layer	15 - 18	1	17	16	16
Filter	128 - 512	128	256	384	256
Dropout Rate	0.1 - 0.5	0.1	0.2	0.5	0.2
Learning Rate	0.001 - 0.005	0.001	0.004	0.003	0.002
Epoch	10 - 30	5	15	10	20
Patience	2 - 10	1	7	8	8

Table 3.3: Hyper-parameter configuration space for VGG16 on image dataset generated using 42 features

Hyper-Parameter	**Search Space**	**Step Size**	**Optimal Value**		
			SA	**TPE**	**RS**
Frozen Layer	15 - 18	1	15	16	17
Filter	128 - 512	128	128	128	384
Dropout Rate	0.1 - 0.5	0.1	0.5	0.2	0.4
Learning Rate	0.001 - 0.005	0.001	0.001	0.001	0.002
Epoch	10 - 30	5	15	20	30
Patience	2 - 10	1	3	10	6

3.7 Weighted Ensemble Learning

A weighted ensemble learning approach is utilized in this framework where the three fine-tuned variants of VGG16 are treated as base classifiers and the predictions of these three base classifiers are combined together with specific optimized weight to make the final prediction. The weighted ensemble approach can be denoted by the following formula:

$$y_{\text{ensemble}} = \sum_{i=1}^{n} W_{\text{i}} F_{\text{i}}(x) \tag{3.5}$$

where $F_i(x)$ represents the prediction made by i^{th} base classifier for input x, W_i be the weight assigned to i^{th} classifier and $y_{ensemble}$ represents the final ensemble prediction for input x. The weight to each base classifier's prediction is assigned based on the number of misclassified samples it incurred during its testing phase. The rationale behind this weight assignment is that the higher the number of misclassification incurred by a base classifier, the lower the importance that it should receive in the ensemble approach. In essence, the weights are inversely proportioned to the number of the misclassified samples, aimed to capture and prioritize the contributions of base classifiers based on their individual performance in minimizing misclassifications. This relationship can be denoted by the following formula:

$$w_{\text{i}} = \frac{1}{m_{\text{i}} + \varepsilon} \tag{3.6}$$

The weights for each base classifier's prediction are calculated using the (5) first where m_i represents the total number of misclassification made by i^{th} classifier and to handle rare cases when m_i is equal to 0, we have incorporated a very small constant

value of 0.001 for ε. Before assigning the weight to each base classifier's prediction, each weight is normalized within the range of 0 to 1. This normalization process is implemented to ensure that the contribution of each base classifier is aligned within the same scale. The goal of this normalization step is to mitigate the sensitivity to the magnitude of weights, fostering a more balanced and effective weighted ensemble stage. This normalization process is ensured by dividing the weight of i^{th} classifier by the total sums of weights which can be represented by:

$$W_{\mathrm{i}} = \frac{w_{\mathrm{i}}}{\sum_{i=1}^{n} w_{\mathrm{i}}} \tag{3.7}$$

Algorithm 1 outlines the step-by-step procedure of our weighted ensemble learning approach. The process initiates by undertaking the training of three discrete VGG16 base models on the designated training dataset. Subsequent to this training phase, predictions are systematically generated on the testing dataset by each base classifier, enabling a comprehensive assessment of each classifier's predictive capability. Next, by evaluating the performance and considering the number of misclassified samples incurred by each base classifier, weights for each base classifier's prediction are calculated using 3.6. Ultimately, we leverage the optimized weights within our weighted ensemble learning approach to make the final prediction.

The weighting mechanism in our proposed ensemble approach is inspired by the algorithm of adjusting weights proposed in [13]. However, in our scheme, the use of mean square error has been departed from, and instead, misclassification samples have been employed as the key criterion for weight determination for each base classifier. In the ensemble learning stage, the assignment of weight to the base classifier is based on the total number of misclassified samples. This approach offers various advantages over other static or extensively fine-tuned weight assignment processes. Not only does this dynamic adaptation enhance the ensemble's flexibility across diverse datasets, but it also simplifies the conceptual understanding of weight assignment. Importantly, this approach maintains robustness without necessitating extensive tuning efforts, thereby circumventing the time and resource consumption typically associated with such tuning processes.

Algorithm 1 Weighted Ensemble Learning with Optimized Weight

Require: Training Data $D_{\text{Train}} = (X_{\text{Train}}, Y_{\text{Train}})$, Testing Data $D_{\text{Test}} = (X_{\text{Test}}, Y_{\text{Test}})$, Base Models, M = M_1 (VGG16-SA), M_2 (VGG16-TPE), M_3 (VGG16-RS)

for $i = 1$ to 3 **do**
    Train $M[i]$ on D_{Train}
end for
for $i = 1$ to 3 **do**
    Predictions[i] = $M[i]$.predict(X_{Test})
end for
TotalMisclassified = 0
for $j = 1$ to length(D_{Test}) **do**
    Misclassified = 0
    for $i = 1$ to 3 **do**
        if Predictions[i][j] $\neq$ TrueClass[j] **then**
            Misclassified = Misclassified + 1
        end if
    end for
    TotalMisclassified = TotalMisclassified + Misclassified
end for
for $i = 1$ to 3 **do**
    $W[i] = 1/(MisclassifiedSamples[i] + \varepsilon)$
end for
Normalize W such that $\sum W = 1$
for $i = 1$ to 3 **do**
    $W[i] = W[i]/\sum W$
end for
for $i = 1$ to length(D_{Test}) **do**
    WeightedSum = 0
    for $j = 1$ to 3 **do**
        WeightedSum += ($W[j]$ * Predictions[j][i])
    end for
    FinalPrediction[i] = WeightedSum
end for

Chapter 4

Performance Evaluation

In this chapter, we begin by delving into the experimental setup that underpinned the implementation of our framework, providing details about the hardware and software configuration. We then discuss about the evaluation metrics used to gauge the performance of different IDS schemes. Subsequently, we conduct a comprehensive performance analysis of traditional ML-based IDS applied to the selected dataset. Finally, we present the outcomes of ME-IDS and undertake a comparative performance analysis against the base classifiers utilized in our framework and extend the comparison to three other benchmark ensemble methods, namely Stacked Ensemble [78], Confidence Averaging [89], and Concatenation Ensemble [89]. This thorough comparative assessment allows us to gauge the efficacy and superiority of our proposed framework in relation to existing methodologies.

4.1 Experimental Setup

All the IDS schemes under investigation were implemented using Python 3.8, and further enhanced through the integration of widely used libraries for machine learning and deep learning framework, including Scikit-Learn and TensorFlow 2.8. To leverage significantly faster training for this undertaking, a high-performance server, equipped with an Nvidia Quadro RTX 8000 GPU and 32 GB of memory, is utilized specifically tailored to boast significant processing capabilities and efficiently handle large-scale datasets. To put it to the test, one of the renowned and widely accepted real-world network intrusion detection datasets is utilized, namely UNSW-NB15, as extensively described in Chapter 3 of our work.

4.2 Evaluation Metrics

The performance of our proposed ME-IDS framework is evaluated through four universally recognized performance metrics: accuracy, precision, recall and f1-score. These metrics collectively can offer a thorough evaluation different IDS's ability to recognize and categorize network intrusion. Notably, most real-world IDS datasets reveal a considerable imbalance, with varied frequencies between normal and attack flows. Acknowledging this inherent challenge, these four evaluation metrics are selected to obtain a more comprehensive view of the performance of different IDS schemes, which emphasizes its ability to maintain a balanced trade-off between true positives (TP), true negatives (TN), false positives (FP) and false negative (FN). For the binary classification scenario, each of these metrics is computed using the average value of both classes.

- **Accuracy:** Accuracy is used to determine the percentage of the right predictions the model makes out of all the predictions. It is computed as the ratio of the total number of correct predictions to the total number of predictions. The formula of accuracy can be denoted as follows, that transforms a model's accuracy into a percentile value which serves as a mean to assess the model's overall performance.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \tag{4.1}$$

- **Precision:** Precision is another evaluation metric that measures the accuracy of positive predictions made by a model. It quantifies how good the model is at correctly identifying true positive instances out of all the positive predictions it generates. In other words, it is used to determine a model's ability to avoid false positive predictions, which can be mathematically represented using the following formula.

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

- **Recall:** Recall, also known as sensitivity, indicates the model's capability to accurately identify the positive instances out of all the actual positive instances.

It is considered as one of the effective evaluation metrics in case of data imbalance. It assesses a model's effectiveness in capturing all the real positives within a dataset. Recall is computed by taking the ratio of true positive predictions to the total number of positive instances in the dataset, which can be denoted by the following formula.

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

- **F1-Score:** The F1-Score is a single evaluation metric that combines both the precision and recall into a single value. It is the harmonic mean of precision and recall, and is calculated using the following mathematical formula:

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.4}$$

F1-Score is also a useful evaluation metric when there is an uneven class distribution among target classes or when it's crucial to minimize both the false positives and false negatives.

4.3 Performance of Traditional ML-based IDS

During the progression of the thesis, the initial architecture of the IDS is formulated using ML and explored using a range of ML-based IDS approaches. This section will provide an overview of our experiments and results we achieved during the testing phase of the traditional ML-based IDS. The practical application of ML involves making predictions about the characteristics of data based on prior analysis conducted during a training phase. In context of the IDS, the primary goal is to determine the nature of the network packets, determining whether it falls into a normal traffic flow or a malicious one. The primary method for building such a system is supervised learning, in which the ML models are trained using a set of training samples, including normal and malicious flow. Based on the learning of signatures and patterns of network flows from the training set, the ML-based IDS is used to classify unseen network flows as normal or attack. In this perspective, the IDS will consistently perceive attack packets as deviation from the norm, and from a statistical perspective,

the characteristics of such flow's data will always appear as outliers when contrasted with the standard baseline.

In this thesis, we have implemented six different ML-based IDS with their default parameters to build the initial architecture for IDS, namely Decision Tree (DT), Random Forest (RF), Gaussian Naïve Bayes (GNB), Logistic Regression (LR), k-Near Neighbors (kNN), and Stochastic Gradient Descent (SGD). These particular classifiers are selected due to their robust effectiveness and versatility in a wide range of classification tasks, making them ideal candidates for the tasks at hand. Moreover, these classifiers encompass a wide array of algorithmic techniques [68], covering tree-based methods (DT and RF), probabilistic models (GNB), linear models (LR), instance-based learning (kNN), and optimization based approaches (SGD) which help in exploring a varied range of modeling strategies and preventing from limiting to the constraints of a single classifier. As stated earlier, the UNSW-NB15 dataset comprises a total of 42 features, out of which 39 features were identified significant through chi-squared p-value in Chapter 3. As a matter of fact, all the six ML-based IDS are trained and evaluated using both the selected 39 features and the complete set of 42 features from the UNSW-NB15 dataset. This approach enables to showcase the importance of the feature selection methods chosen in the proposed framework within the realm of IDS.

RF and DT, both the tree-based IDS, worked well in identifying the network flows compared to other ML-based IDS when trained and tested with all the features from UNSW-NB15 dataset, with RF exhibiting the highest accuracy of 87.03% and DT following closely at 85.96%. RF also yielded strong precision and recall score of 89.64% and 85.74%, respectively, resulting in a commendable F1-Score of 86.41%. On the other hand, DT maintained a balanced trade-off between precision and recall score, yielding an F1-score of 85.41%. However, GNB lagged behind in accuracy at 75.20%, with precision and recall scores that are not as competitive, leading to a relatively lower F1-score of 73.83%. LR, kNN, and SGD fell in between in terms of performance, with RF demonstrating the best overall performance with all the features from UNSW-NB15 dataset. The performance evaluation of the above mentioned ML-based IDS with all the 42 features from UNSW-NB15 dataset is summarized in Table 4.1.

Table 4.1: Performance of Traditional ML-based IDS with All Features from UNSW-NB15 Dataset

Classifier	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
DT	85.96	87.57	84.87	85.41
RF	87.03	89.64	85.74	86.41
GNB	75.20	76.79	73.64	73.83
LR	80.40	84.91	78.47	78.86
kNN	84.78	87.12	83.46	84.04
SGD	80.93	87.05	78.80	79.17

The classification results, outlined in Table 4.3, offer the significance of the selected features by providing an insight into the performance of various ML-based IDS. These results are particularly noteworthy as they are based on the utilization of 39 out of the 42 available features. The deliberate selection of these features has evidently contributed to an enhancement in performance metrics such as accuracy, precision, recall, and F1-score. The analysis of these outcomes provides a nuanced understanding of the impact of specific features on the overall detection capabilities, thus aiding in the refinement and optimization of intrusion detection methodologies. Notably, RF stands out as the top performer is case of 39 features as well, achieving an increased accuracy of 87.18%. RF also exhibits the highest precision and recall score of 89.79% and 85.89% in the 39-feature scenario, respectively, which results in a commendable F1-score of 86.56%. DT also showed a performance improvement, with slight increase of accuracy score from 85.96% to 86.06% when trained the classifier with 39 features instead of all the features. DT also showed a balanced trade-off between precision (87.65%) and recall (84.91%) score. On the other hand, GNB lags behind the tree-based classifiers in terms of 39 features as well, with an accuracy of 75.99%, suggesting that it may not be the best ideal choice for the particular dataset. Logistic Regression (LR) delivered moderate performance with an accuracy of 80.40%, accompanied by balanced precision (84.92%) and recall (78.47%), resulting in an F1-score of 78.86%. k-Nearest Neighbors (kNN) exhibited slightly better accuracy at

84.87%, with precision (87.20%) and recall (83.56%) figures, resulting in an F1-score of 84.14%, showcasing a trade-off between precision and recall. Stochastic Gradient Descent (SGD) posted similar results to kNN, with an accuracy of 80.93% and an F1-score of 79.17%, implying a fair balance between precision and recall.

Table 4.2: Performance of Traditional ML-based IDS with Selected Features from UNSW-NB15 Dataset

Classifier	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
DT	86.01	87.67	84.91	85.46
RF	87.18	89.79	85.89	86.56
GNB	75.99	78.81	74.15	74.29
LR	80.40	84.92	78.47	78.86
kNN	84.87	87.20	83.56	84.14
SGD	80.93	87.05	78.80	79.17

It is noticeable from Table 4.1 and 4.2 that both the tree-based IDS, RF and DT, displayed consistent high performance in both scenarios. The accuracy of RF increase from 87.03% with all features to 87.18% with selected feature set, showing a slight improvement of 0.15%. In case of DT, the accuracy also improved from 85.96% with all features to 86.01% with reduced feature set, indicating a 0.05% improvement. While both LR and SGD based IDS demonstrated consistent performance across all performance metrics when evaluated with both the reduced feature set and the complete set of features, it is noteworthy that the GNB-based IDS exhibited a notable and substantial improvement specifically in precision. GNB's precision increased from 76.79% when using all features to 78.81% with the reduced feature set, marking a notable 2.02% improvement. The improvement in precision underscores the effectiveness of feature selection in enhancing the GNB's capacity to accurately classify instances within the positive class. This refinement is particularly noteworthy in scenarios where minimizing false positives is crucial, emphasizing the significance of thoughtful feature selection strategies in optimizing model performance for specific classification objectives. Additionally, the performance of kNN remained consistent

in both the scenario across all the evaluation metrics, showing minimal improvement when using fewer features. The performance improvement of all the metrics for the six traditional ML-based IDS when utilizing 39 features instead of 42 features has been visually represented in Figure 4.1. This illustration provides a clear overview of how the selection of the important features positively impacted classification by making slight increase in all the evaluation metrics.

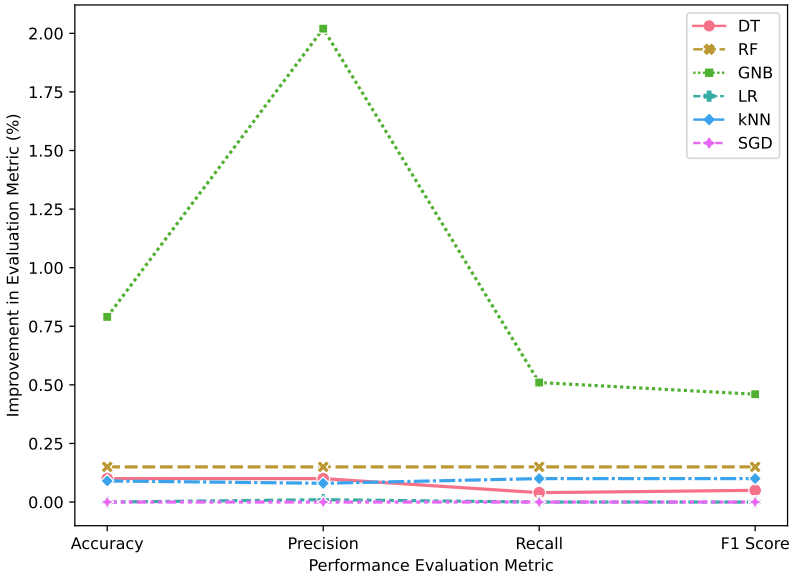

Figure 4.1: Performance Improvement of Traditional ML-based IDS with Selected Features from UNSW-NB15 Dataset

4.4 Performance of ME-IDS and Other Benchmark IDS Frameworks

In our proposed ME-IDS framework, three different hyper-tuned variants of VGG16, namely VGG-SA, VGG-TPE, and VGG16-RS, are utilized in our approach using SA, TPE, and RS hyper-parameter optimization techniques, respectively. The prediction of these three variants of VGG16 classifier are combined together with an optimized weight assigned to each classifier based on the performance of each variant, as mentioned in Chapter 3. The effectiveness of our proposed ME-IDS framework is assessed using four above-mentioned four performance evaluation metrics as well: accuracy, precision, recall and f1-score.

In order to assess the performance of ME-IDS, we conducted a comparative analysis with three other benchmark ensemble-based methods, namely Stacked Ensemble

[78], Concatenation Ensemble [89], and Confidence Averaging [89]. Stacked Ensemble learning is constructed as a two-layer classifier, with DT, LR, and GNB serving as the level 0 or base classifiers, while SGD is employed as the level 1 or meta classifier, as mention in [78]. On the other hand, five different pre-trained transfer learning models, namely VGG16, VGG19, Xception, ResNet50, and InceptionV3, are primarily utilized to train and test the images generated from the image data extracted from the UNSW-NB15 dataset. Among the five pre-trained models employed, the top three performing models on our dataset are selected to build the Confidence Averaging and Concatenation Ensemble. The evaluation of Stacked Ensemble is assessed using the tabular representation of the UNSW-NB15 dataset, while the remaining two methods, Concatenation Ensemble and Confidence Averaging, are evaluated based on the images generated from the same UNSW-NB15 dataset. To ensure a fair and meaningful comparison between our approach and other benchmark methods, we have conducted evaluations under consistent conditions including using the same simulation settings as solely comparing performance metrics mentioned in the respective original publications without its implementation can be deceptive when determining the superiority of one method over another.

Initially, we conducted an evaluation of the proposed ME-IDS framework, comparing its performance to base classifiers and benchmark ensemble learning approaches using images generated from all 42 features of the UNSW NB15 Dataset. Starting with the stacked ensemble, implemented using the tabular data, demonstrated an overall accuracy score of 85.89%, showing strong precision but slightly lower recall and F1-Score. Moving on to the concatenation ensemble and confidence averaging, both implemented using the generated RGB images, showed significant improvements with accuracies of 95.71% and 97.39%, respectively, demonstrating a better balance between precision and recall. The significant leap in the evaluation metrics unquestionably underscores the superiority of the image-based approach over the tabular data. The subsequent three hyper-tuned variants of VGG16 utilized in ME-IDS framework, namely VGG16-SA, VGG16-TPE and VGG16-RS, exhibited much higher accuracy, with VGG16-TPE being the top performer, achieving 98.31% accuracy, 98.34% precision, 98.24% recall, and a F1-score of 98.29%. The proposed

framework, ME-IDS, combining the predictions of all three VGG16 hypertuned variants with optimized weights, outperforms all other methods, boasting an impressive accuracy score of 98.77%, along with high precision, recall, and F1-score, making it the most robust among those evaluated. Table 4.3 summarizes the evaluation metric scores of our proposed ensemble framework, as well as the scores of three different hyper-tuned variants of VGG16, namely VGG-SA, VGG-TPE, and VGG16-RS, utilized in our approach respectively, on the RGB images generated from all the features of the UNSW-NB15 dataset, alongside three other mentioned benchmark ensemble methods.

Table 4.3: Performance and Comparison of ME-IDS with Three Other Benchmark Ensemble Learning Approaches for IDS and Base Classifiers on Image Data Generated from All Features of UNSW-NB15 Dataset

Method	**Accuracy** (%)	**Precision** (%)	**Recall** (%)	**F1-Score** (%)
Stacked Ensemble [78]	85.89	87.45	84.81	85.34
Concatenation Ensemble [89]	95.71	96.38	95.22	95.61
Confidence Averaging [89]	97.39	97.73	97.09	97.34
VGG16-SA	98.16	98.33	97.98	98.13
VGG16-TPE	98.31	98.34	98.24	98.29
VGG16-RS	97.69	97.99	97.44	97.66
ME-IDS	**98.77**	**98.87**	**98.67**	**98.76**

The performance analysis, summarized in Table 4.3, reveals its substantial performance over base classifiers and other benchmark ensemble learning approaches, when evaluated on the RGB images generated from all features. Compared to stacked ensemble, the proposed approach exhibited an increase of 12.88%, 11.42%, 13.86% and 13.42% in accuracy, precision, recall, and F1-score. Similarly, when contrasted with Concatenation ensemble and confidence averaging, the proposed approach showcased a significant gain of 3.06% and 1.38% in accuracy, respectively as well. The proposed

framework also outperformed the base classifiers, with improvements observed across all four evaluation metrics. Figure 4.2 illustrates the performance improvement across different evaluation metrics of ME-IDS, evaluated on the RGB images generated using all features, in compared to the three individual classifiers employed within the framework, and three other mentioned benchmark ensemble methods.

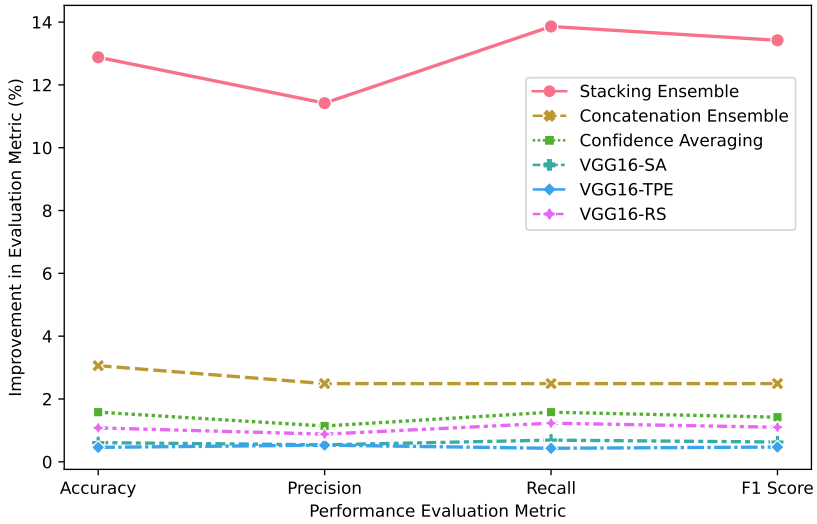

Figure 4.2: Performance improvement of ME-IDS compared to base classifiers used in ensemble approach and three other benchmark ensemble methods, evaluated on RGB images generated using All features.

The evaluation scores presented in Table 4.4 for the ME-IDS, as well as the base classifiers and other benchmark ensemble learning methods applied to the images generated from 39 features out of 42 features offer a significant insights into the impact of the feature selection on performance metrics. Except for the stacked ensemble and concatenation ensemble, the other methods demonstrated substantial improvements across all evaluation metrics with reduced number of features. Namely, Confidence Averaging achieved an accuracy score of 97.72%, showing a slight increase from 97.39% in the previous metrics. In case of base classifiers, VGG16-RS experienced a substantial improvement performance when trained and evaluated with the RGB images generated using selected features. It achieved an impressive accuracy score of 99.43%, a notable increase from its previous accuracy of 97.69%. VGG16-RS demonstrated notable improvements across other evaluation metrics as well, showcasing enhanced precision, recall, and F1-score. This signifies that the positive impact

of the feature selection, contributing to a more nuanced and refined performance in terms of the model's performance. VGG16-TPE also showed a significant improvement with reduced number of features, achieving an accuracy score of 99.15% from 98.31%. Additionally, this improvement extended to other evaluation metrics, with VGG16-TPE's precision, recall, and F1-score also improving to 99.17%, 99.11%, and 99.14%, respectively. When the predictions of the base classifiers were combined with optimized weights in the proposed framework ME-IDS, a notable improvement was observed. ME-IDS, with the reduced number of features, saw its performance elevate from an initial accuracy of 98.77%, 98.87% precision, 98.67% recall, and a 98.76% F1-score to an impressive 99.72% accuracy, 99.74% precision, 99.68% recall, and a 99.71% F1-score. This outcome underscores the significant enhancement achieved through the feature selection strategy, making ME-IDS a highly effective intrusion detection system.

Table 4.4: Performance and Comparison of ME-IDS with Three Other Benchmark Ensemble Learning Approaches for IDS and Base Classifiers on Image Dataset Generated using Selected Features of UNSW-NB15 Dataset

Method	**Accuracy** (%)	**Precision** (%)	**Recall** (%)	**F1-Score** (%)
Stacked Ensemble [78]	85.85	87.47	84.75	85.29
Concatenation Ensemble [89]	95.31	96.07	94.78	95.20
Confidence Averaging [89]	97.72	97.96	97.50	97.69
VGG16-SA	98.72	98.79	98.63	98.70
VGG16-TPE	99.15	99.17	99.11	99.14
VGG16-RS	99.43	99.43	99.43	99.43
ME-IDS	**99.72**	**99.74**	**99.68**	**99.71**

The ME-IDS framework stands out prominently with reduced set of features as well, exhibiting substantial improvements across all evaluation metrics when compared to base classifiers and benchmark ensemble methods. When compared to

stacked ensemble, ME-IDS outperformed it significantly, with an accuracy increase of 13.87%, precision improvement of 12.27%, recall enhancement of 14.93%, and an F1-socre boost of 14.42%. Concatenation ensemble lags behind ME-IDS with an accuracy different of 4.41%, while confidence averaging also falls short, showing an accuracy difference of 2.00%. Notably, it showcased substantial improvement in recall score as well, by surpassing concatenation ensemble by 4.9% in recall score, indicating its effectiveness in capturing a higher proportion of actual positive instances. Moreover, ME-IDS also outperformed the base classifiers in terms of accuracy by margins of 1.00%, 0.57%, and 0.29% compared to VGG16-SA, VGG16-TPE, and VGG16-RS, respectively. Additionally, ME-IDS demonstrated significant performance improvement across the other evaluation metrics as well under investigation, as illustrated in Figure 4.3.

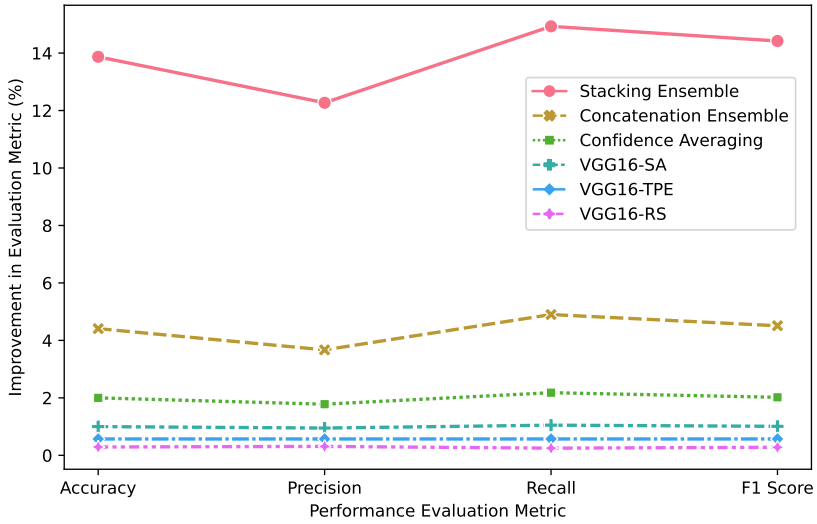

Figure 4.3: Performance improvement of ME-IDS compared to base classifiers used in ensemble approach and three other benchmark ensemble methods, evaluated on RGB images generated using All features.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

The substantial volume of data transmitted over the internet presents unique challenges for the traditional IDS. Recently, new IDS solutions have harnessed the advantages of DL, which has helped address many traditional problems. However, DL based IDS still encounters various constraints, including a scarcity of training data and the substantial computation resources required for training models from scratch. To address these challenges, this paper proposed a novel framework that combines the principles of transfer learning and ensemble learning, where transfer learning mitigates the issue of computational complexity by eliminating the need to train a model from scratch, and ensemble learning merges the results of multiple classifiers with optimized weights to create a more robust IDS. Furthermore, to address the challenges of complex model training and curse of feature dimensionality, one of the filter-based feature selection methods has been employed with specific threshold within the proposed framework. The experimental results demonstrate that the proposed ensemble transfer learning-based IDS framework effectively distinguishes between normal and attack flows, achieving superior performance with accuracy, precision, recall, and F1-score scores of 99.72%, 99.74%, 99.68%, and 99.71% with reduced set of features, respectively, when compared to other state-of-the-art ensemble methods on a real-world IDS dataset.

5.2 Future Work

5.2.1 Other Tabular Data to Image Conversion Methods

CNNs have been widely applied in various field, especially in the speech [67] and image processing [79], where the arrangement of features contains crucial information for modelling. Nonetheless, CNN are generally not suitable for modelling tabular

data since such data typically lack any spatial relationships among their features [92]. To meet this challenge, in our proposed work, we utilized a chunk based method to transform tabular data to RGB images to make it suitable for feeding into pre-trained CNN model. There are some other methods proposed in recent years to transfer non-image tabular data to images. Zhu et al. [92] proposed an algorithm called image generated for tabular data (IGTD), where each feature of the tabular data is mapped to a specific pixel location in a way that places similar features near one another in the transformed image. The algorithm optimizes this mapping by minimizing the disparity between the order of the feature distances and the order of distances between their corresponding pixels within the image. Inspired by their own work of Super Characters [73], Sun et al. [74] proposed superTML method which combines both the idea of Super Characters and two-dimensional embedding to transform tabular data into 2D images. The SuperTML method suggested deals with categorical data and missing values in tabular data seamlessly, eliminating the necessity to convert them into numerical values beforehand. The generation of images with two-dimensional embedding utilizing SuperTML method is illustrated in Figure 5.1.

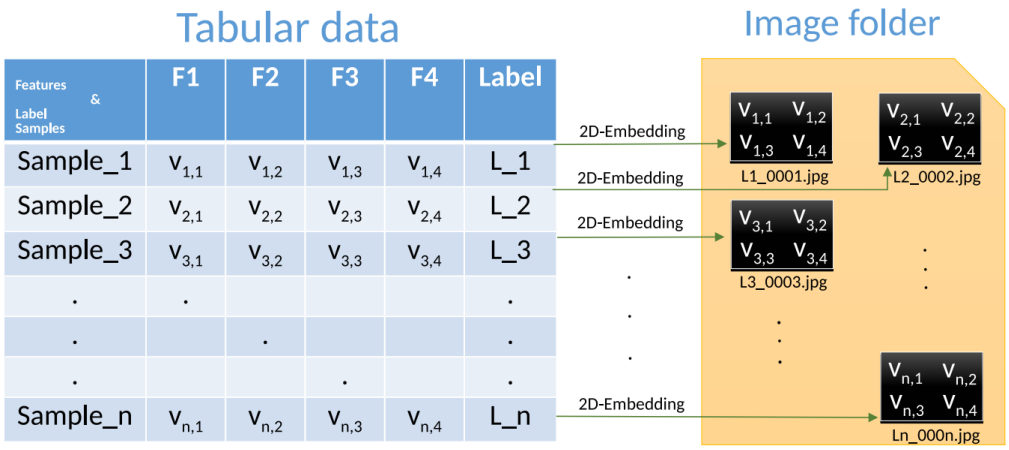

Figure 5.1: Process of Transforming Tabular Data to Images using SuperTML.

In the context of a classification task, where some features are more important than others, SuperTML_VF is employed to allocate more space and increase the font size for the significant features. Conversely, SuperTML_EF maintains the same font size for all features. An example of generated images from the Iris and Wine Datasets using SuperTML_EF and SuperTML_VF is shown in Figure 5.2. In future research,

(a) (b)

Figure 5.2: Generated Images using SuperTML: (a) Image Generated using SuperTML_EF on Iris data with equal importance given to each feature; (b) Image Generated using SuperTML_VF on Wine data with varied font size based on feature importance.

both of these image transformation methods will be used with the UNSW-NB15 dataset, and a comparison will be made between the chunk-based approach applied in the ME-IDS framework and these two alternative methods.

5.2.2 GAN-based Synthetic Data Generation for IDS

Within the domain of IDS, the imbalanced dataset issue is a pervasive challenge, where the number of normal traffic samples significantly surpasses the instances of attack traffic samples. This substantial class imbalance can impede the efficacy of both ML and DL based IDS in accurately identifying and categorizing the samples that belong to the minority class. The primary concern is that this imbalance may lead the classifier to exhibit a bias, favoring the majority class and consequently compromising its ability to detect and correctly classify instances from the minority class [49].

To tackle the challenge of imbalanced datasets, the Synthetic Minority Oversampling Technique (SMOTE) has become a widely embraced approach across different domains, including the research domain of IDS. SMOTE is valued for its simple design and established effectiveness in improving the performance of classifiers dealing with imbalanced data [22]. Nevertheless, despite its widespread use, SMOTE comes

with its own set of limitations. One notable limitation of SMOTE lies in the potential for over-generalization due to oversampling noisy and uninformative data [71]. When the minority class is over-sampled, including instances that may not be genuinely informative, the classifier might learn patterns that do not accurately represent the underlying distribution of the data. Moreover, SMOTE can increase overlap between classes near their boundaries [71], introducing complexities in classifier decision boundaries and possibly causing misclassifications or reduced precision in distinguishing between majority and minority classes.

In recent years, different variants of Generative Adversarial Networks (GAN) are extensively employed for generating synthetic data. The architecture of GANs, proposed in 2014 [25], consists of two neural network designed to imitate a data distribution in a unsupervised way, which function as a two-player zero-sum game: a Generator (G) creates new data samples resembling the original distribution without copying it, while a Discriminator (D) aims to differentiate between real and generated data in a zero-sum game scenario. From a formal perspective, D is used to estimate the $p(y|x)$, which is, the probability of a label y based on a given sample x, while G creates a samples from hidden space z, expressed as $G(z)$. These networks engage in a competitive process where G aims to produce increasingly realistic output as the learning progresses, while D strives to enhance its ability to distinguish between real and synthesized samples. Both networks are in a continual cycle of improvement, where if G generates better outputs, D faces a tougher challenge in telling them apart. Conversely, if D becomes more accurate, G encounters increased difficulty in fooling D. The process of GAN is based on a min-max game [32] in which D aims to maximize its accuracy while G aims to minimize it, which can be represented as:

$$\min_G \max_D L(D, G) = \mathbb{E}_{x \sim p_r}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(x))] \tag{5.1}$$

where $x \sim p_r$ represents the distribution of the real data, while $z \sim p_z$ signifies the probability distribution in the latent space of the generator (G), typically implemented as Gaussian or uniform noise. G utilizes the noise to generate new data samples denoted as *G(z)*. The role of the discriminator (D) is to distinguish between real data distribution *D(x)* and the synthesized distribution *D(G(x))*. As the training of GAN progresses, the discriminator model gains resilience to data transformations

as it continually generates synthetic data. Figure 5.3 illustrates an abstract overview of the sturcture of a GAN model.

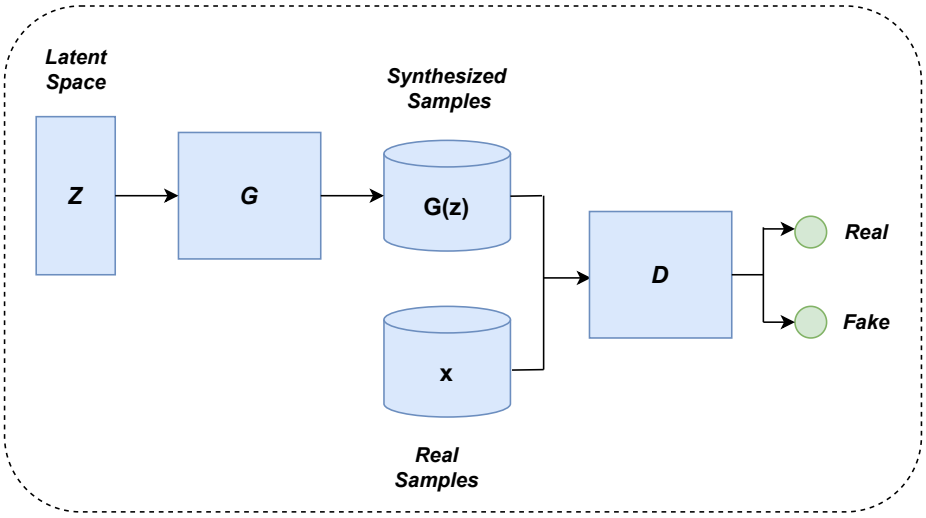

Figure 5.3: Model Structure of Generative Adversarial Networks (GAN)

While GANs showed its significance in data synthesizing, only a few of these methods specifically concentrate on synthesizing data to improve the training of IDS. Lee et al. [42] employed a standard GAN to produce synthetic data resembling the exiting dataset in order to address the data imbalance problem of CICIDS2017 dataset. The resulting oversampled dataset was then fed as an input to Random Forest model, which indicate notable enhancement in the classification of minority attack classes. Although the UNSW-NB15 dataset is acknowledged for having a relatively more balanced class distribution compared to some other IDS datasets, the issue of class imbalance persists in nearly all other IDS datasets. As a matter of fact, the data synthesis through GAN is a promising research area that should be investigated as the newly synthesized data has little to no substantial effect on the statistical distribution of the input data.

5.2.3 Integration of eXplainable AI in IDS

The pursuit of adaptable and efficient anomaly-based IDS has led researchers to assess numerous ML/DL/TL algorithms with the goal of achieving high classification accuracy and minimizing flase alarms. Additionally, there remains a growing need for explaining the predictions made by these classfiers, as various user groups, specially

the cyber-security specialists can benefit from understanding the reasons behind these intrusions and working mechanism of detecting these intrusion [34]. This has given rise to the field of explainable artificial intelligence (XAI), which aims to shed light on the inner workings and predictions of the IDS models. While XAI is currently been applied to different domain due to its superiority, its application in IDS still requires further exploration to determine its effectiveness in identifying attack surfaces and vectors.

There are two promising model-agnostic visualization-based XAI methods, namely Shapley Additive Explanations (SHAP) [47] and Partial Dependence Plot (PDP) [23] which can be used in explaining the functioning of a utilized classifier as a whole. SHAP was initially introduced in 2017 as a comprehensive method for explaining the predictions of black box model. SHAP employs a game theoretic strategy to provide explanations for the outcomes of any classification model. This approach establishes a link between optimal credit assignment and local explanations by leveraging Shapley values, a concept from game theory, and their associated extensions. SHAP operates on the principles of assessing the significance of each feature by quantifying their contributions to the classifier's decision in classifying a class. The SHAP value is computable by considering individual data instances. An explanation for a given instance x can be approximated using this formula:

$$g(z') = \phi_0 + \sum_{i=1}^{m} \phi_i z'_i, \quad z' \in \{0,1\}^m \tag{5.2}$$

One notable benefit of SHAP over Local Interpretable Model-agnostic Explanations (LIME) is that SHAP distributes effects more uniformly, while LIME assumes the model's behavior to be locally linear. In particular, LIME evaluates instances by their proximity to the original instance, whereas in SHAP, instances are weighted to form a coalition in the Shapley value estimation. Another visualization-based XAI method, Partial Dependence Plot (PDP), is widely utilized to illustrate the connection feature values and the predictions of a black-box model through visual representation. PDP serves as a universal and model-agnostic method that takes into account all data points to depict the extensive connection between a particular feature and the model's predictions. Through visual representation, PDP showcases how alterations in the selected feature impact the overall predictions of the model. This relationship

can be expressed using the following formula.

$$PD(x_s) = E(x_c)[f(x_s, x_c)] = \int f(x_s, x_c) dP(x_c) \tag{5.3}$$

where f represents the classifiers, $x_s \in S$ corresponds to the set of input features for which PDP is generated, $x_c \in C$ determines the set of remaining input features, and $dP(x_c)$ indicates the marginal distribution of x_c. the PDP function integrates the model's outcomes across the distribution of set C to illustrate the connection between the features within set S and the model's output. The incorporation of these XAI methods in IDS can effectively instill trust in the mind of cyber-security analysts, as it enables them to get a better view and understanding of the system's decisions and in turn, gain confidence in its effectiveness. As a result, our forthcoming stages of research and development will prioritize the integration and refinement of XAI methods within the IDS framework. This strategic focus aims to empower cybersecurity analysts with a more comprehensive and interpretable understanding of the IDS's behavior, fostering trust and confidence in its ability to accurately identify and respond to security threats.

Bibliography

[1] Ali Agga, Ahmed Abbou, Moussa Labbadi, Yassine El Houm, and Imane Hammou Ou Ali. Cnn-lstm: An efficient hybrid deep learning architecture for predicting short-term photovoltaic power production. *Electric Power Systems Research*, 208:107908, 2022.

[2] Farhan Ahmad, Zeeshan Ahmad, Chaker Abdelaziz Kerrache, Fatih Kurugollu, Asma Adnane, and Ezedin Barka. Blockchain in internet-of-things: Architecture, applications and research directions. In *2019 International conference on computer and information sciences (ICCIS)*, pages 1–6. IEEE, 2019.

[3] Zeeshan Ahmad, Adnan Shahid Khan, Kartinah Zen, and Farhan Ahmad. Msads: Multistage spectrogram image-based anomaly detection system for iot security. *Transactions on Emerging Telecommunications Technologies*, page e4810, 2023.

[4] Zeeshan Ahmad, Adnan Shahid Khan, Kashif Nisar, Iram Haider, Rosilah Hassan, Muhammad Reazul Haque, Seleviawati Tarmizi, and Joel JPC Rodrigues. Anomaly detection using deep neural network for iot architecture. *Applied Sciences*, 11(15):7050, 2021.

[5] Muataz Salam Al-Daweri, Khairul Akram Zainol Ariffin, Salwani Abdullah, and Mohamad Firham Efendy Md. Senan. An analysis of the kdd99 and unsw-nb15 datasets for the intrusion detection system. *Symmetry*, 12(10):1666, 2020.

[6] Mohammad Al-Omari, Majdi Rawashdeh, Fadi Qutaishat, Mohammad Alshira'H, and Nedal Ababneh. An intelligent tree-based intrusion detection model for cyber security. *Journal of Network and Systems Management*, 29:1–18, 2021.

[7] Isra Al-Turaiki and Najwa Altwaijry. A convolutional neural network for improved anomaly-based network intrusion detection. *Big Data*, 9(3):233–252, 2021.

[8] Ammar Alazab, Michael Hobbs, Jemal Abawajy, and Moutaz Alazab. Using feature selection for intrusion detection system. In *2012 international symposium on communications and information technologies (ISCIT)*, pages 296–301. IEEE, 2012.

[9] Razvan Andonie and Adrian-Catalin Florea. Weighted random search for cnn hyperparameter optimization. *arXiv preprint arXiv:2003.13300*, 2020.

[10] Jamil Asim, Adnan Shahid Khan, Rashad Mahmood Saqib, Johari Abdullah, Zeeshan Ahmad, Shehla Honey, Shehroz Afzal, Malak S Alqahtani, and Mohamed Abbas. Blockchain-based multifactor authentication for future 6g cellular networks: A systematic review. *Applied Sciences*, 12(7):3551, 2022.

[11] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[12] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

[13] Dariusz Brzeziński and Jerzy Stefanowski. Accuracy updated ensemble for data streams with concept drift. In *International conference on hybrid artificial intelligence systems*, pages 155–163. Springer, 2011.

[14] Ismail Butun, Salvatore D Morgera, and Ravi Sankar. A survey of intrusion detection systems in wireless sensor networks. *IEEE communications surveys & tutorials*, 16(1):266–282, 2013.

[15] Nadia Chaabouni, Mohamed Mosbah, Akka Zemmari, Cyrille Sauvignac, and Parvez Faruki. Network intrusion detection for iot security based on learning techniques. *IEEE Communications Surveys & Tutorials*, 21(3):2671–2701, 2019.

[16] Zhewei Chen, Linyue Zhou, and Wenwen Yu. Adasyn- random forest based intrusion detection model. In *2021 4th International Conference on Signal Processing and Machine Learning*, pages 152–159, 2021.

[17] Fangda Cui. A wasserstein gan based framework for adversarial attacks against intrusion detection systems. 2022.

[18] Mwamba Kasongo Dahouda and Inwhee Joe. A deep-learned embedding technique for categorical features encoding. *IEEE Access*, 9:114381–114391, 2021.

[19] Harsh Dhillon and Anwar Haque. Towards network traffic monitoring using deep transfer learning. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 1089–1096. IEEE, 2020.

[20] Abebe Abeshu Diro and Naveen Chilamkurti. Distributed attack detection scheme using deep learning approach for internet of things. *Future Generation Computer Systems*, 82:761–768, 2018.

[21] S Eugene. James p. anderson: An information security pioneer. *IEEE Security & Privacy*, 6(1):9, 2008.

[22] Alberto Fernández, Salvador Garcia, Francisco Herrera, and Nitesh V Chawla. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, 61:863–905, 2018.

[23] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[24] Mengmeng Ge, Naeem Firdous Syed, Xiping Fu, Zubair Baig, and Antonio Robles-Kelly. Towards a deep learning-driven intrusion detection approach for internet of things. *Computer Networks*, 186:107784, 2021.

[25] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[26] Azidine Guezzaz, Said Benkirane, Mourade Azrour, and Shahzada Khurram. A reliable network intrusion detection approach using decision tree with enhanced data quality. *Security and Communication Networks*, 2021:1–8, 2021.

[27] Achref Haddaji, Samiha Ayed, and Lamia Chaari Fourati. A transfer learning based intrusion detection system for internet of vehicles. In *2023 15th international conference on developments in esystems engineering (dese)*, pages 533–539. IEEE, 2023.

[28] Asmaa Halbouni, Teddy Surya Gunawan, Mohamed Hadi Habaebi, Murad Halbouni, Mira Kartiwi, and Robiah Ahmad. Cnn-lstm: hybrid deep neural network for network intrusion detection system. *IEEE Access*, 10:99837–99849, 2022.

[29] Vanlalruata Hnamte and Jamal Hussain. Dcnnbilstm: An efficient hybrid deep learning-based intrusion detection system. *Telematics and Informatics Reports*, 10:100053, 2023.

[30] Faisal Hussain, Syed Ghazanfar Abbas, Muhammad Husnain, Ubaid U Fayyaz, Farrukh Shahzad, and Ghalib A Shah. Iot dos and ddos attack detection using resnet. In *2020 IEEE 23rd International Multitopic Conference (INMIC)*, pages 1–6. IEEE, 2020.

[31] Muzaffer Can Iban. An explainable model for the mass appraisal of residences: The application of tree-based machine learning algorithms and interpretation of value determinants. *Habitat International*, 128:102660, 2022.

[32] Guillermo Iglesias, Edgar Talavera, and Alberto Díaz-Álvarez. A survey on gans for computer vision: Recent research, analysis and taxonomy. *Computer Science Review*, 48:100553, 2023.

[33] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. A deep learning approach for network intrusion detection system. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pages 21–26, 2016.

[34] Marwa Keshk, Nickolaos Koroniotis, Nam Pham, Nour Moustafa, Benjamin Turnbull, and Albert Y Zomaya. An explainable deep learning-enabled intrusion detection framework in iot networks. *Information Sciences*, 639:119000, 2023.

[35] Adnan Shahid Khan, Zeeshan Ahmad, Johari Abdullah, and Farhan Ahmad. A spectrogram image-based network anomaly detection system using deep convolutional neural network. *IEEE Access*, 9:87079–87093, 2021.

[36] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, 2(1):1–22, 2019.

[37] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, Joarder Kamruzzaman, and Ammar Alazab. Hybrid intrusion detection system based on the stacking ensemble of c5 decision tree classifier and one class support vector machine. *Electronics*, 9(1):173, 2020.

[38] Taehoon Kim and Wooguil Pak. Deep learning-based network intrusion detection using multiple image transformers. *Applied Sciences*, 13(5):2754, 2023.

[39] Nishoak Kosaraju, Sainath Reddy Sankepally, and K Mallikharjuna Rao. Categorical data: Need, encoding, selection of encoding method and its emergence in machine learning models—a practical review study on heart disease prediction dataset using pearson correlation. In *Proceedings of International Conference on Data Science and Applications: ICDSA 2022, Volume 1*, pages 369–382. Springer, 2023.

[40] Ketan Kotecha, Raghav Verma, Prahalad V Rao, Priyanshu Prasad, Vipul Kumar Mishra, Tapas Badal, Divyansh Jain, Deepak Garg, and Shakti Sharma. Enhanced network intrusion detection system. *Sensors*, 21(23):7835, 2021.

[41] Satish Kumar, Sunanda Gupta, and Sakshi Arora. Research trends in network-based intrusion detection systems: A review. *IEEE Access*, 9:157761–157779, 2021.

[42] JooHwa Lee and KeeHyun Park. Gan-based imbalanced data intrusion detection system. *Personal and Ubiquitous Computing*, 25:121–128, 2021.

[43] Matheus Macedo Leonardo, Tiago J Carvalho, Edmar Rezende, Roberto Zucchi, and Fabio Augusto Faria. Deep feature-based classifiers for fruit fly identification (diptera: Tephritidae). In *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pages 41–47. IEEE, 2018.

[44] Jingmei Liu, Yuanbo Gao, and Fengjie Hu. A fast network intrusion detection system using adaptive synthetic oversampling and lightgbm. *Computers & Security*, 106:102289, 2021.

[45] Siti-Farhana Lokman, Abu Talib Othman, Muhamad Husaini Abu Bakar, and Shahrulniza Musa. The impact of different feature scaling methods on intrusion detection for in-vehicle controller area network (can). In *Advances in Cyber Security: First International Conference, ACeS 2019, Penang, Malaysia, July 30–August 1, 2019, Revised Selected Papers 1*, pages 195–205. Springer, 2020.

[46] Maya Hilda Lestari Louk and Bayu Adhi Tama. Dual-ids: A bagging-based gradient boosting decision tree model for network anomaly intrusion detection system. *Expert Systems with Applications*, 213:119030, 2023.

[47] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[48] Mohammad Masum and Hossain Shahriar. Tl-nid: Deep neural network with transfer learning for network intrusion detection. In *2020 15th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 1–7. IEEE, 2020.

[49] Mariama Mbow, Hiroshi Koide, and Kouichi Sakurai. An intrusion detection system for imbalanced dataset based on deep learning. In *2021 Ninth International Symposium on Computing and Networking (CANDAR)*, pages 38–47. IEEE, 2021.

[50] John McHugh, Alan Christie, and Julia Allen. Defending yourself: The role of intrusion detection systems. *IEEE software*, 17(5):42–51, 2000.

[51] Nour Moustafa and Jill Slay. The significant features of the unsw-nb15 and the kdd99 data sets for network intrusion detection systems. In *2015 4th international workshop on building analysis datasets and gathering experience returns for security (BADGERS)*, pages 25–31. IEEE, 2015.

[52] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 military communications and information systems conference (MilCIS)*, pages 1–6. IEEE, 2015.

[53] Nour Moustafa and Jill Slay. The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. *Information Security Journal: A Global Perspective*, 25(1-3):18–31, 2016.

[54] Mohammad Reza Norouzian and Sobhan Merati. Classifying attacks in a network intrusion detection system based on artificial neural networks. In *13th International Conference on Advanced Communication Technology (ICACT2011)*, pages 868–873. IEEE, 2011.

[55] Ogobuchi Daniel Okey, Dick Carrillo Melgarejo, Muhammad Saadi, Renata Lopes Rosa, João Henrique Kleinschmidt, and Demóstenes Zegarra Rodríguez. Transfer learning approach to ids on cloud iot devices using optimized cnn. *IEEE Access*, 11:1023–1038, 2023.

[56] Suad Mohammed Othman, Fadl Mutaher Ba-Alwi, Nabeel T Alsohybe, and Amal Y Al-Hashida. Intrusion detection model using machine learning algorithm on big data environment. *Journal of big data*, 5(1):1–12, 2018.

[57] Yazan Otoum, Dandan Liu, and Amiya Nayak. Dl-ids: a deep learning–based intrusion detection framework for securing iot. *Transactions on Emerging Telecommunications Technologies*, 33(3):e3803, 2022.

[58] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[59] Florian Pargent, Florian Pfisterer, Janek Thomas, and Bernd Bischl. Regularized target encoding outperforms traditional methods in supervised machine learning with high cardinality features. *Computational Statistics*, 37(5):2671–2692, 2022.

[60] Robi Polikar. Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, 6(3):21–45, 2006.

[61] Nicholas Pudjihartono, Tayaza Fadason, Andreas W Kempa-Liehr, and Justin M O'Sullivan. A review of feature selection methods for machine learning-based disease risk prediction. *Frontiers in Bioinformatics*, 2:927312, 2022.

[62] Medha Pujari, Bhanu Prakash Cherukuri, Ahmad Y Javaid, and Weiqing Sun. An approach to improve the robustness of machine learning based intrusion detection system models against the carlini-wagner attack. In *2022 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 62–67. IEEE, 2022.

[63] Vinayakumar Ravi, Rajasekhar Chaganti, and Mamoun Alazab. Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Computers and Electrical Engineering*, 102:108156, 2022.

[64] B Pavan Venkat Reddy, Likith Preetham Alla, Hemprasad Yashwant Patil, et al. Parkinson's disease classification using quantile transformation and rfe. In *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pages 01–05. IEEE, 2021.

[65] LM Rasdi Rere, Mohamad Ivan Fanany, and Aniati Murni Arymurthy. Simulated annealing algorithm for deep learning. *Procedia Computer Science*, 72:137–144, 2015.

[66] Eva Rodríguez, Pol Valls, Beatriz Otero, Juan José Costa, Javier Verdú, Manuel Alejandro Pajuelo, and Ramon Canal. Transfer-learning-based intrusion detection framework in iot networks. *Sensors*, 22(15):5621, 2022.

[67] Tara N Sainath, Brian Kingsbury, Abdel-rahman Mohamed, George E Dahl, George Saon, Hagen Soltau, Tomas Beran, Aleksandr Y Aravkin, and Bhuvana Ramabhadran. Improvements to deep convolutional neural networks for lvcsr. In *2013 IEEE workshop on automatic speech recognition and understanding*, pages 315–320. IEEE, 2013.

[68] IH Sarker. Machine learning: algorithms, real-world applications and research directions. sn comput sci 2: 160, 2021.

[69] Shyla Shyla, Vishal Bhatnagar, Vikram Bali, and Shivani Bali. Optimization of intrusion detection systems determined by ameliorated hnadam-sgd algorithm. *Electronics*, 11(4):507, 2022.

[70] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[71] Paria Soltanzadeh and Mahdi Hashemzadeh. Rcsmote: Range-controlled synthetic minority over-sampling technique for handling the class imbalance problem. *Information Sciences*, 542:92–111, 2021.

[72] Hyun Min Song, Jiyoung Woo, and Huy Kang Kim. In-vehicle network intrusion detection using deep convolutional neural network. *Vehicular Communications*, 21:100198, 2020.

[73] Baohua Sun, Lin Yang, Patrick Dong, Wenhan Zhang, Jason Dong, and Charles Young. Super characters: A conversion from sentiment classification to image classification. *arXiv preprint arXiv:1810.07653*, 2018.

[74] Baohua Sun, Lin Yang, Wenhan Zhang, Michael Lin, Patrick Dong, Charles Young, and Jason Dong. Supertml: Two-dimensional word embedding for the precognition on structured tabular data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019.

[75] Bayu Adhi Tama, Marco Comuzzi, and Kyung-Hyune Rhee. Tse-ids: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. *IEEE access*, 7:94497–94507, 2019.

[76] Bayu Adhi Tama and Sunghoon Lim. Ensemble learning for intrusion detection systems: A systematic mapping study and cross-benchmark evaluation. *Computer Science Review*, 39:100357, 2021.

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

[78] Ngamba Thockchom, Moirangthem Marjit Singh, and Utpal Nandi. A novel ensemble learning-based model for network intrusion detection. *Complex & Intelligent Systems*, pages 1–22, 2023.

[79] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 648–656, 2015.

[80] Mubarak Albarka Umar and Chen Zhanfang. Effects of feature selection and normalization on network intrusion detection. *Authorea Preprints*, 2023.

[81] Patrick Vanin, Thomas Newe, Lubna Luxmi Dhirani, Eoin O'Connell, Donna O'Shea, Brian Lee, and Muzaffar Rao. A study of network intrusion detection systems using artificial intelligence/machine learning. *Applied Sciences*, 12(22):11752, 2022.

[82] Ravi Vinayakumar, Mamoun Alazab, KP Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. Deep learning approach for intelligent intrusion detection system. *Ieee Access*, 7:41525–41550, 2019.

[83] Monika Vishwakarma and Nishtha Kesswani. A new two-phase intrusion detection system with naïve bayes machine learning for data classification and elliptic envelop method for anomaly detection. *Decision Analytics Journal*, 7:100233, 2023.

[84] Ning Wang, Yimin Chen, Yang Hu, Wenjing Lou, and Y Thomas Hou. Feco: Boosting intrusion detection capability in iot networks via contrastive learning. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 1409–1418. IEEE, 2022.

[85] Raniyah Wazirali. An improved intrusion detection system based on knn hyperparameter tuning and cross-validation. *Arabian Journal for Science and Engineering*, 45(12):10859–10873, 2020.

[86] Carsten Witt. Worst-case and average-case approximations by simple randomized search heuristics. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 44–56. Springer, 2005.

[87] Tao Wu, Honghui Fan, Hongjin Zhu, Congzhe You, Hongyan Zhou, and Xianzhen Huang. Intrusion detection system combined enhanced random forest with smote algorithm. *EURASIP Journal on Advances in Signal Processing*, 2022(1):1–20, 2022.

[88] Li Yang and Abdallah Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, 2020.

[89] Li Yang and Abdallah Shami. A transfer learning and optimized cnn based intrusion detection system for internet of vehicles. In *ICC 2022-IEEE International Conference on Communications*, pages 2774–2779. IEEE, 2022.

[90] Ruizhe Yao, Ning Wang, Zhihui Liu, Peng Chen, and Xianjun Sheng. Intrusion detection system in the advanced metering infrastructure: a cross-layer feature-fusion cnn-lstm-based approach. *Sensors*, 21(2):626, 2021.

[91] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[92] Yitan Zhu, Thomas Brettin, Fangfang Xia, Alexander Partin, Maulik Shukla, Hyunseung Yoo, Yvonne A Evrard, James H Doroshow, and Rick L Stevens. Converting tabular data into images for deep learning with convolutional neural networks. *Scientific reports*, 11(1):11325, 2021.

[93] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.