IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 2, June 2025
Impact Factor: 7.67

# A Cost-Sensitive Deep Learning-Based Approach for Network Traffic Classification

**Murala Chinna Mahalakshmi, Matlapudi Naga Durga Yagneswari**
**Kampa Sushma, N. Chandra Shekhar**
Department of Computer Science and Engineering
RVR & JC College of Engineering, Chowdavaram,Guntur

**Abstract**: *Network Traffic Classification (NTC) is a critical task in intrusion detection, network optimization, and cybersecurity. But class imbalance of network traffic data is a natural challenge to baseline machine learning approaches, leading to biased classification and poor detection of minority traffic classes. To compensate for this, in this paper we propose a cost-sensitive deep learning approach that combines new data balancing techniques such as SMOTE (Synthetic Minority Over-sampling Technique) with new deep architectures such as Feed- Forward Neural Networks (FFN), Convolutional Neural Networks (CNN), and Stacked Autoencoders (SAE). We also propose cost-sensitive models such as COSTSAE, CostCNN,DeepPacket+CostCNN,DeeperPacket+CostSAE, DFR+CostSAE and DFR+CostCNN that add class-specific penalties to maximize classification performance over low-frequency traffic classes. Our approach is evaluated on the ISCX VPN-nonVPN dataset, which has varied network traffic types such as chat, file transfer, streaming, video, audio, and email protocols. We evaluate model performance in terms of loss analysis metrics, confusion matrices, accuracy, recall, precision, categorical accuracy, and training history. Experimental findings indicate that our cost-sensitive deep learning models enhance classification performance, particularly for minority classes, by a large margin in comparison with traditional deep learning approaches..*

**Keywords**: Network traffic classification, class imbalance, deep learning, cost-sensitive learning

## I. INTRODUCTION

Network Traffic Classification (NTC) is a critical network security and management task, enabling traffic monitoring, intrusion detection, and quality of service improvement. Due to the dynamic nature of network traffic through advanced encryption and complexity, statistical and rule-based classification techniques struggle to effectively distinguish various classes of traffic [2], [16]. Deep learning (DL) models have gained widespread use in this research field due to their ability to learn intricate patterns and features in network traffic [1], [18]. However, these models are susceptible to class imbalance, where majority traffic classes dominate the minority traffic classes, leading to a high misclassification rate for minority traffic types. Class imbalance negatively impacts network security, resource allocation, and overall classification accuracy [7], [19].

To address the problem of class imbalance in deep learning based NTC, various methods have been proposed. Resampling methods, such as the Synthetic Minority Over-Sampling Technique (SMOTE) [13], have been widely used to enhance the representation of minority classes. Although these methods improve classification performance, they introduce challenges such as increased computational cost and susceptibility to overfitting [6]. Another approach, cost-sensitive learning, tackles class imbalance by incorporating misclassification costs into the training process, assigning higher penalties to misclassified minority instances [5], [7]. While cost-sensitive deep learning has been extensively studied in domains like medical diagnosis and fraud detection [10], [11], its application in encrypted NTC remains underexplored.

In this paper, we introduce a cost-sensitive deep learning framework designed to enhance the classification accuracy of minority traffic types. We compare the performance of various deep learning models, including Feed-Forward Neural Network (FFN), Convolutional Neural Network (CNN), SMOTE- enabled CNN (SMOTE+CNN), Stacked

Autoencoder (SAE), SMOTE-enabled SAE (SMOTE+SAE), Cost-Sensitive CNN (CostCNN), and Cost-Sensitive SAE (CostSAE). Furthermore, we evaluate more advanced architectures such as DeepPacket+CostCNN [14], DeeperPacket+CostSAE, DFR+CostCNN, and DFR+CostSAE [15]. To further enhance classification performance, we propose adaptive cost-sensitive learning methods that dynamically adjust misclassification costs based on real-time data distribution. This approach ensures consistent classification accuracy across various network conditions. Additionally, we investigate the effects of feature selection and feature augmentation methods in combination with cost-sensitive learning to improve model generalization further [21].

These models are trained and tested on the SDN dataset, which includes diverse traffic types such as chat, file transfer, streaming, video, audio, and email services [20]. We compare the performance of our cost-sensitive models against baseline deep learning models, SMOTE-based models, and conventional cost-sensitive approaches. Our results demonstrate that cost-sensitive learning significantly improves classification performance for minority traffic classes and represents a robust approach for real-world network traffic classification The rest of this paper is structured as follows. Section II presents a review of the current NTC solutions, describing earlier solutions and their shortcomings. Section III introduces the SDN dataset. Section IV explains our new cost- sensitive deep learning solution. Section V discusses the experimental setup, the evaluation metrics, and the results. Lastly, Section VI summarizes the paper and outlines possible directions for future work.

TABLE I: LIST OF ABBREVIATIONS IN ALPHABETICAL ORDER

| Abbreviation | Full Form |
|---|---|
| **CNN** | Convolutional Neural Network |
| **CostCNN** | Cost-Sensitive Convolutional Neural Network |
| **CostSAE** | Cost-Sensitive Stacked Autoencoder |
| **DL** | Deep Learning |
| **DFR** | Deep-Full-Range |
| **DFR+CostCNN** | Deep-Full-Range with Cost- Sensitive Convolutional Neural Network |
| **DFR+CostSAE** | Deep-Full-Range with Cost- Sensitive Stacked Autoencoder |
| **FFN** | Feed-Forward Neural Network |
| **NTC** | Network Traffic Classification |
| **SAE** | Stacked Autoencoder |
| **SMOTE** | Synthetic Minority Over- Sampling Technique |
| **SMOTE+CNN** | SMOTE-enabled Convolutional Neural Network |
| **SMOTE+SAE** | SMOTE-enabled Stacked Autoencoder |
| **SDN** | Software-Defined Networking |

## II. RELATED WORK

NTC aims to classify traffic flows according to their genera tion applications. In this regard, supervised learning algorithms are mostly used to train detection models based on labeled training data [16]. The current research of NTC concen trates on the application of DL techniques. Table II discusses previous studies on DL-based NTC. DL has achieved good results in traditional generic NTC, as shown in the literature [30]. The Seq2Img approach [17] for example, is a CNN model containing two convolutional layers, two max-pooling layers, and three full connection layers. In Seq2Img, stream sequences were converted into six-channel images using an embedded kernel as the input of the CNN model. Lopez-Martin et al. [18] stacked CNN architecture and two Long Short-Term Memory (LSTM) networks, whereby the final tensor of the CNN was reshaped into a matrix to be used as the input of the LSTM networks. A feature selection framework based on the combination of symmetric uncertainty and Deep Belief Networks (DBN) was presented in [21]. Nevertheless, due to user privacy, security protocols (e.g., HTTPS, SSH, and SSL) are used in most applications to encrypt data traffic. Therefore, NTC has become an essential task these days. Datanet [20] is an application-aware frame work for application identification tasks, which exploits three DL classifiers of MLP, CNN, and SAE. Datanet considers a four-step pre-processing to provide ideal data for DL models: (1) Parsing to remove the Ethernet header and Data-link layer information, such as MAC address;

(2) Truncating and zero padding to generate equal data packets (i.e., 1500 bytes) by either cutting or adding zero to the packets;

(3) Normalization of all values of the dataset, which are converted to a value between 0 and 1; and (4) Labeling that assigns a label to each data packet (e.g., AIM, Email, and Netflix). A similar approach was proposed by Lotfollahi et al. [14], called Deep Packet, for both traffic description and application identification tasks, using the undersampling method to balance the dataset, where the major classes' instances were randomly removed. CNN and SAE have been commonly used for training the classifier models, for instance, Deep-Full-Range (DFR) [15] was used for L1 regularization in three DL models (i.e., CNN, SAE, and MLP). However, the models were not evaluated on unbalanced data. Yao et al. [1] employed a capsule network, which is an enhancement of CNN, for end-to-end classification. MIMETIC [24] is a multimodal DL framework for NTC. A 1D CNN was developed by Wang et al. [19] for both the f low-level and session-level classification tasks. Zou et al. [22] stacked 2D CNN and LSTM models for NTC. Multi-task learning is a recently developed framework for NTC that there is no need for a large labeled traffic dataset [25]. Ren et al. [28] proposed a tree structural approach that divides a large NTC into small classifications and a RNN classifier is performed on each node of the tree. Aceto et al. [31] developed a systematic framework to classify mobile traffic classifications using SAE, LSTM, and CNN algorithms. They investigated their frame works in terms of NTC abject, the type and the amount of input data, and the DL model architecture. Three datasets of real human users' activity were used for evaluation. In a separate work, Huang et al. [32] employed multi-task learning system for end-to-end NTC. Three classification tasks of malware detection, VPN-capsulation recognition, and Trojan classification were considered. A 2D CNN model was trained in which lower layers' parameters were shared by all three tasks. A feature optimization approach based on DBN was designed in [21] to provide optimal and robust features for NTC. Class imbalance is a challenging problem in NTC. A few numbers of works have focused on class unbalanced traffic data through DL-based traffic data generation. Wang et al. [26] used conditional Generative Adversarial Networks (GAN) to generate synthesized traffic samples for the minority classes by learning the characteristics of the original traffic data. A sim ilar data augmentation technique based on LSTM and kernel density estimation was developed both to generate packets for low-frequency classes and to replicate the numerical features of each class [23]. Xu et al. [27] designed an improved cross-entropy loss function based on the probability obtained from the Softmax layer. Sadeghzadeh et al. [29] developed six SAE classifiers for detecting adversarial network traffic. Three categories of packet classification, flow content classification, and flow time series classification were considered for attack detection. TABLE III DATA DISTRIBUTION OF ISCX VPN-NONVPN DATASET

Although there have been various DL-based NTC approaches, only some previous works addressed the class imbalance problem, which applied either undersampling technique [14],

[20] or synthesized traffic generation for the minority classes [23], [26]. The use of undersampling is straightforward, but useful knowledge associated with the majority classes can be lost. On the other hand, simply generating new packets makes the training process complex and burdensome since the size of the training set increases [3]. None of the related works ever employed a cost-sensitive learning strategy in DL models for addressing the class imbalance problem. Unlike previous studies that primarily focused on traditional deep learning models for network traffic classification, this paper integrates cost-sensitive learning to mitigate class imbalance issues. Specifically, we introduce adaptive cost-sensitive learning across multiple deep learning architectures, including CNN, SAE, and hybrid models like DeepPacket and DFR, to enhance the classification of minority traffic classes.

TABLE II: A SUMMARY OF WORKS ON DL-BASED TRAFFIC CLASSIFICATION

| Reference | Classifier | Dataset | Class imbalance strategy |
|---|---|---|---|
| Chen et al. [17] | CNN | Case study dataset | —- |
| Lopez-Martin et al. [18] | CNN, LSTM | RedIRIS from Spain | —- |
| Wang et al. [19] | CNN | ISCX VPN-nonVPN | —- |
| Wang et al. [20] | MLP, CNN, SAE | ISCX VPN-nonVPN | —- |
| Shi et al. [21] | DBN | Cambridge, UNIBS | —- |
| Zou et al. [22] | CNN+LSTM | ISCX VPN-nonVPN | —- |
| Huang et al. [?] | CNN | CTU-13, ISCX | —- |
| Shi et al. [?] | DBN | Cambridge, UNIBS | —- |
| Yao et al. [1] | Capsule Network | UTSC-2016 | —- |
| Hasibi et al. [23] | Convolutional Recurrent Neural Network | A case study dataset | A data augmentation technique based on LSTM to generate packets for low-frequency classes |
| Aceto et al. [24] | CNN+LSTM | Mobile datasets | —- |
| Aceto et al. [?] | SAE, LSTM, and CNN | Human users activity | —- |
| Rezaei & Liu [25] | CNN | QUIC, ISCX VPN-nonVPN | —- |
| Lotfollahi et al. [14] | SAE, CNN | ISCX VPN-nonVPN | Undersampling to remove instances of the majority classes |
| Wang et al. [26] | CNN | ISCX2012, USTC-TFC2016 | Generate synthesized traffic for minority classes using GAN |
| Xu et al. [27] | CNN, LSTM, ResNet | ISCX VPN-nonVPN | Improved cross entropy loss function |
| Ren et al. [28] | RNN | ISCX VPN-nonVPN | —- |
| Sadeghzadeh et al. [29] | SAE | ISCX VPN-nonVPN | Undersampling |
| Our proposed approach | CNN, SAE | ISCX VPN-nonVPN | Cost-sensitive DL |

## III. DATASET DESCRIPTION

The dataset for this project was acquired from the UNB ISCX Network Traffic Dataset, built for network traffic classification in an SDN (Software Defined Networking) context. It consists of traffic data from multiple applications partitioned into five classes, namely Chat, File Transfer, Streaming, Video, and Audio Communications and Email. The dataset contains network traffic from popular applications that support chat communications including ICQ, Gmail, Facebook, and Hangouts. For file transfer traffic, protocols captured include SFTP-DOWN, Skype, and FTPS. Streaming services contained in the dataset comprise Spotify, Vimeo, YouTube, and Netflix. The dataset also contains traffic for video call communications involving Skype and Hangouts, with audio calls captured from VoIPBuster, Skype, Hangouts, and Facebook, in addition to email traffic over SMTP, POPS, and IMAPS protocols. The dataset contains 41 features selected for the study that represent certain packet level traffic captures, such as packet length, inter- arrival time, packets per second, and bytes per second, for the forward and reverse direction of the communication's data stream. To enhance the quality of the dataset for training, preprocessing steps of feature selection, filtering categories, and shuffling data were completed to remove any ordered bias. The dataset was then partitioned into training (80%) and test (20%) subsets while maintaining the class distribution in the training subset for unbiased model evaluation

TABLE III: CLASS DISTRIBUTION OF THE SDN DATASET

| Category | Number of Samples |
|---|---|
| FILE-SKYPE | 720 |
| AUDIO-HANGOUTS | 551 |
| EMAIL | 495 |
| VIDEO-HANGOUTS | 488 |
| AUDIO-SKYPE | 482 |
| AUDIO-FACEBOOK | 388 |
| FILE-FTPS | 375 |
| VIDEO-SKYPE | 325 |
| AUDIO-VOIPBUSTER | 316 |
| STR-SPOTIFY | 310 |
| FILE-SFTP-DOWN | 282 |
| CHAT-GMAIL | 267 |
| CHAT-FACEBOOK | 220 |
| STR-YOUTUBE | 197 |

| | |
|---|---|
| STR-NETFLIX | 162 |
| CHAT-ICQ | 136 |
| STR-VIMEO | 133 |
| CHAT-HANGOUTS | 129 |

One significant challenge with this dataset is its class imbalance streak. Some categories of network traffic have significantly more samples than others. For example, FILE-SKYPE has the highest number of samples (720), and CHAT-HANGOUTS has the lowest number of samples (129). In order to prevent the model from favoring the more plentiful classes, a cost-sensitive learning approach was used to ensure all classes of traffic were fairly learned. Categorical labels were one-hot encoded and numerical features were normalized using MinMax Scaling and Standardization (Z-score Normalization) to maintain consistency in the magnitudes of the other features. Therefore, 20% of the training data was used as a validation set during model training so that the performance is monitored and training can be stopped to avoid over-fitting

Table III shows the class distribution of the data after preprocessing and clearly demonstrates the dataset's imbalance, which requires cost-sensitive deep learning methods so the model doesn't favor the plentiful classes while still robustly classifying as much traffic as it can.

## IV. COST-SENSITIVE TRAFFIC CLASSIFICATION

In this section, we present a cost-sensitive DL approach for managing the class imbalance problem in NTC. pre-processing, cost matrix generation and DL model.

### Preprocessing

To prepare the dataset in an acceptable format for deep learning classifiers, a multi-stage preprocessing pipeline is used. The dataset, retrieved from a CSV file in the Kaggle input directory, includes several network traffic features representing forward and reverse packet statistics.

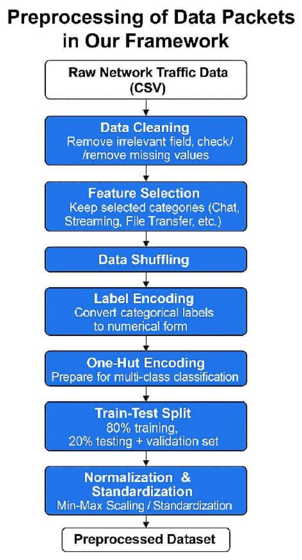

Preprocessing of Data Packets
in Our Framework
Raw Network Traffic Data (CSV)
Data Cleaning
Remove irrelevant field, check/ /remove missing values
Feature Selection
Keep selected categories (Chat, Streaming, File Transfer, etc.)
Data Shuffling
Label Encoding
Convert categorical labels to numerical form
One-Hut Encoding
Prepare for multi-class classification
Train-Test Split
80% training, 20% testing + validation set
Normalization & Standardization
Min-Max Scaling / Standardization
Preprocessed Dataset

Fig. 1. Preprocessing of data packets

Data Collection: The data is loaded from a CSV file in the Kaggle input directory. The features of network traffic behavior are included, e.g., packet length, inter-arrival time, packets per second, and bytes per second for both forward and reverse flows.

Feature Selection: Only useful features are kept to have a clean dataset. They are a set of statistical features pertaining to packet length, inter-arrival time, and transmission rates in the forward and reverse directions. The dataset also has a categorical label, category, that indicates different categories of network applications.

Class Filtering: In order to concentrate on particular types of network traffic, only chosen categories are kept. These are applications for chat, file transfer, streaming, video, audio, and email. The dataset is narrowed down to contain 18 particular categories, like CHAT-GMAIL, STR-NETFLIX, AUDIO-SKYPE, and FILE-SFTP-DOWN, so that the analysis is still applicable to these application types.

Shuffling and Data Distribution: The data is shuffled to maintain randomness during training, eliminating any possible biases in the data ordering. The category distribution is also examined to determine the representation of various classes in the dataset, making sure that no category overpowers the data.

Label Encoding: As machine learning models demand numerical labels, the categorical values in the category column are encoded into numerical form by Label Encoder. Moreover, a one-hot encoding method is used to convert these numerical labels into multi-class classification format.

Train-Test Split: For effective training and assessment, the dataset is divided into training and testing sets. The division is performed in an 80-20 ratio with class balance through stratification. Post this step, the dataset is comprised of 4780 training samples and 956 testing samples, offering a good- balanced dataset for model building.

Data Normalization and Standardization: For improving model performance, two scaling techniques are used for features. Min-Max Scaling is employed to scale the values between 0 and 1 so that all features will have the same range. Standardization is also used by centering the data using mean subtraction and standard division, which improves training stability and convergence. Both scaled datasets are saved separately for experimentation.

Validation Set Creation:20% of the training data is reserved for validation. The validation set serves to measure model performance at training time. After this division, the remaining dataset is made up of 3824 training samples and 956 validation samples, thereby guaranteeing that the model gets trained and tested on different but representative sets of data.

*Cost Matrix Generation*

In cost-sensitive deep learning, a cost matrix is used to manage class imbalance and misclassification penalties in an efficient manner. The cost matrix imposes varying misclassification costs depending on the severity of errors so that the model focuses on reducing high-impact misclassifications.

To build the cost matrix, we begin by examining the class distribution of the dataset and obtaining misclassification penalties from real-world consequences. Lower-represented classes in the dataset are assigned more weight to compensate for their underrepresentation. Likewise, misclassification cost is used in data-driven fashion, i.e., confusion matrix-based modifications and weighted cross-entropy loss, to punish wrong predictions in proportion to their importance. We employed Weighted Cross-Entropy Loss, in which every class is given a weight that is inversely proportional to its frequency so that the model is not biased towards majority classes. Furthermore, a cost-sensitive loss function is formulated by including class- specific misclassification costs within the objective function so that the model can put more emphasis on reducing important misclassification errors. In addition, we apply Focal Loss, a sophisticated cost-sensitive learning method, to down-weight simple cases and emphasize hard ones to make the classifier more robust. We iteratively update the cost matrix by inspecting confusion matrix outputs after training and update weights according to the most frequent misclassification patterns.

**Weighted Cross-Entropy Loss**

To handle class imbalance, we modify the standard cross-entropy loss by introducing class-specific weights:

$$L=-\sum_{i=1}^{N} w_{y_i}\cdot \log P(y_i) \quad (1)$$

where:

N is the total number of samples,$w_{y_i}$is the weight assigned to class $y_i$ , calculated as the inverse of class frequency, $P(y_i)$ is the predicted probability of the correct class $y_i$ .

Focal Loss

Focal Loss is used to focus the model's attention on hard-to- classify samples by down-weighting easy examples:

$$L=-\sum_{i=1}^{N} w_{y_i}\cdot(1-P(y_i))^{\gamma} \log P(y_i) \quad (2)$$

where: γ (gamma) is a tunable focusing parameter (commonly set to 2), $(1-P(yi)^{\gamma}$ reduces the loss contribution for well-classified examples, $w_{yi}$ is the class weight as defined earlier.

Cost-Sensitive Loss Function

The cost-sensitive loss function incorporates a predefined cost matrix C to penalize specific misclassifications:

$L = -\,{}^{N}\sum_{i=1}\ {}^{M}\sum_{j=1}\ C(yi, j).\ P(yi = j)\ \log P(yi = j)$ (3) where:

C (yi, j) is the misclassification cost of predicting class j when the true class is yi,M is the total number of classes.

Confusion Matrix-Based Cost Adjustment

After model training, the confusion matrix CM is analysed to adjust misclassification costs dynamically:

$C(i, j) = CM(i, j) / {}^{M}\sum_{j=1} CM(i, j)$ (4) where:

CM (i, j) represents the number of times class i was misclassified as class j.

By combining these cost-sensitive methods, our method guarantees a balanced classification performance with less misclassification error and high predictive accuracy for all traffic categories.

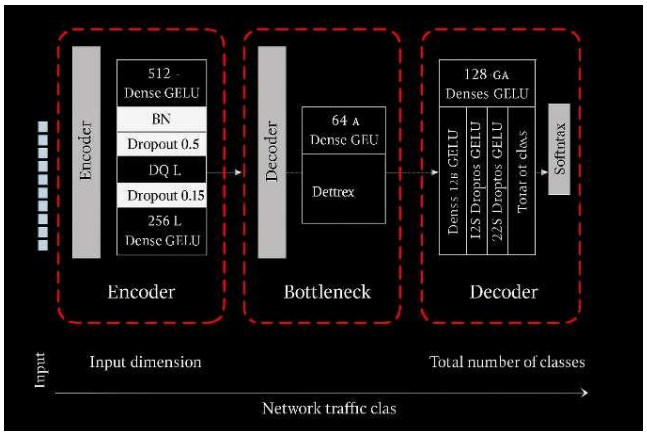


Fig. 2. Our SAE architecture for network traffic classification.

*Architectures of Deep Learning Models*

*Stacked Auto-Encoder(SAE):* Our SAE architecture is made up of several encoding and decoding layers, a deep autoencoder. The input layer is of size 40 (i.e., INPUT_DIM), and the output layer is of size OUT_CLASSES for network traffic classification. The encoder stack consists of successively smaller dense layers (512, 256, 128, and 64 neurons) with GELU activation, batch normalization, and dropout (ratio = 0.15). The decoder reflects encoder architecture with growing layer dimensions, reconstructing input before the last classification layer. The output layer has the SoftMax activation function for multi-class classification. The architecture is a stacked autoencoder approach, as shown in Fig. 2 (SAE Image).

*Convolutional Neural Network (CNN):* The CNN structure is tailored for one-dimensional network traffic data. The model includes an input layer with shape (40,1), followed by eight convolution layers with progressively larger filter sizes (16, 32, 64, 128, 256, 512, 1024, and 2048). Every convolution layer has a kernel size of 3 with GELU activation, batch normalization, and dropout (ratio = 0.15). After the final convolutional layer, Flatten () is executed, then a fully connected dense layer for classification with OUT_CLASSES neurons and a SoftMax activation.

*Cost-Sensitive Convolutional Neural Network (CostCNN):*

The CostCNN model is built specifically to deal with unbalanced network traffic data. The architecture begins with an input layer of shape (40,1), and is followed by two convolutional layers with 64, and 128 filters, with kernel size of 3, GELU activation, batch normalization, and DropOut (ratio = 0.15). Each pair of convolutional layers is separated with MaxPooling1D for down sampling. After convolutional feature extraction, the model utilizes Flatten (), then a fully connected dense layer with 256 neurons, GELU activation, batch normalization, and DropOut (ratio = 0.15). The output

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 2, June 2025
Impact Factor: 7.67

layer has OUT_CLASSES neurons with SoftMax activation for classification. The architecture is a stacked autoencoder approach, as shown in Fig. 3.

*Cost-sensitive Autoencoder (COSTSAE):*
It consists of an input layer with 40 features, followed by three encoding layers (512, 256, and 128 neurons) with GELU activation, batch normalization, and dropout. A bottleneck layer with 64 neurons captures the compressed representation of input data. The decoder reconstructs input data through two layers (128 and 256 neurons). The final dense layer classifies into OUT_CLASSES using SoftMax activation.

## V. EXPERIMENTAL RESULTS

In this section, we present the evaluation of the models based on several performance metrics, including accuracy, recall, precision, and categorical accuracy. The performance of each model is evaluated using confusion matrices, which highlight misclassifications, and we also compare the results to assess how well each model handles the class imbalance issue in the network traffic classification (NTC) task. Additionally, we provide visualizations of the training accuracy and loss curves to further understand the models' convergence and performance during training

First convolution layer Second convolution layer

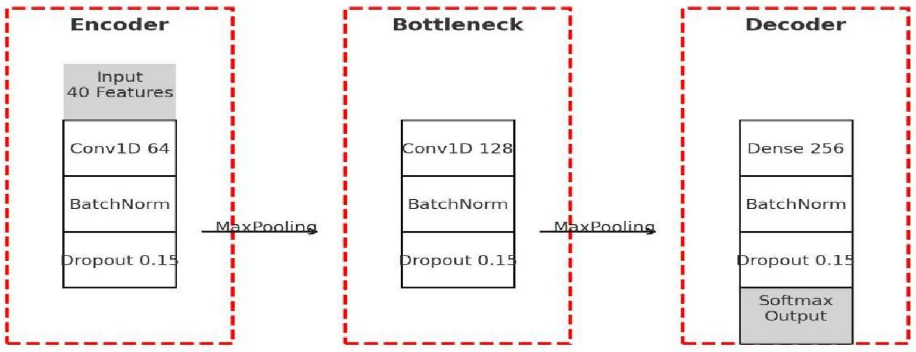
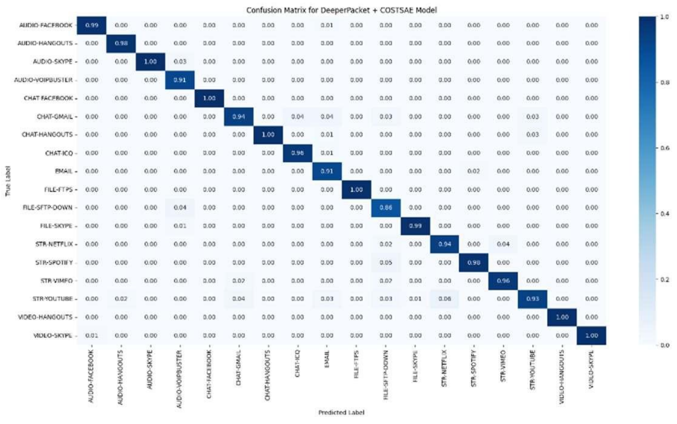


Fig. 3. Our CostCNN architecture for network traffic classification



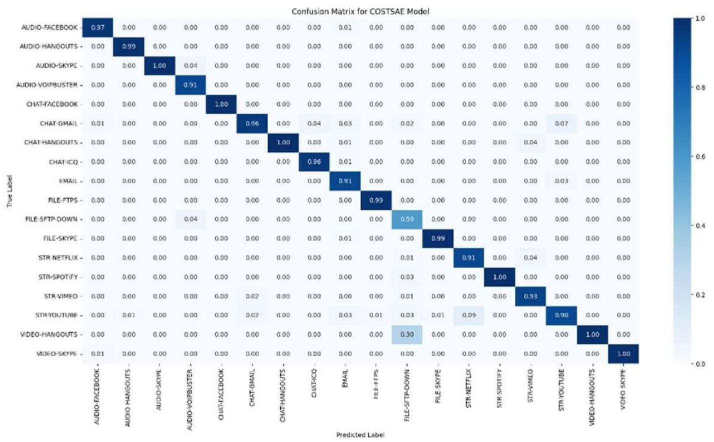


Fig. 4. Confusion matrices of Deep Packet (SAE) and CostSAE

*Evaluation Measures*

We used three classification metrics, including *accuracy* (Eq. (5)), *recall* (Eq. (6)),and *precision* (Eq. (7)), to evaluate the NTC approaches.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

$$Recall = \frac{TP}{TP + FN} \quad (6)$$

$$Precision = \frac{TP}{TP + FP} \quad (7)$$

In these equations, TP, FP, TN, and FN indicate True Positive, False Positive, True Negative, and False Negative, respectively

*Results and Discussion*

Figures 4 and 5 illustrate the confusion matrices for our cost- sensitive deep learning models compared to their cost-insensitive counterparts. Specifically, Figure 4 presents the confusion matrices for Deep Packet (SAE) and CostSAE, while Figure 5 compares those of Deep Packet (CNN) and CostCNN. The results indicate that majority classes such as FILE-SKYPE, AUDIO-HANGOUTS, EMAIL, VIDEO-HANGOUTS, AUDIO-SKYPE, and AUDIO-FACEBOOK, which have a high number of instances in the dataset, negatively impact the classification of minority classes such as FILE-FTPS, AUDIO- VOIPBUSTER, STR-SPOTIFY, FILE-SFTP-DOWN, CHAT- GMAIL, CHAT-FACEBOOK, STR-YOUTUBE, STR- NETFLIX, CHAT-ICQ, STR-VIMEO, and CHAT-HANGOUTS. However, our cost-sensitive approaches effectively mitigate the misclassification of low-frequency traffic instances, significantly reducing the number of false predictions. Additionally, there is a noticeable improvement in the number of correctly classified instances across all traffic classes, demonstrating the effectiveness of our models in handling class imbalance

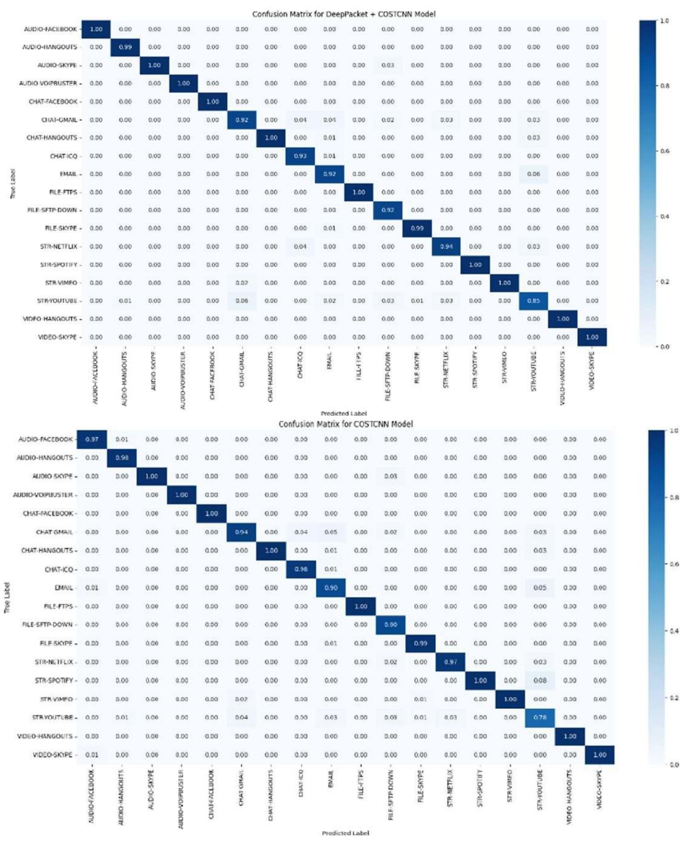


Fig. 5. Confusion matrices of Deep Packet (CNN) and CostCNN

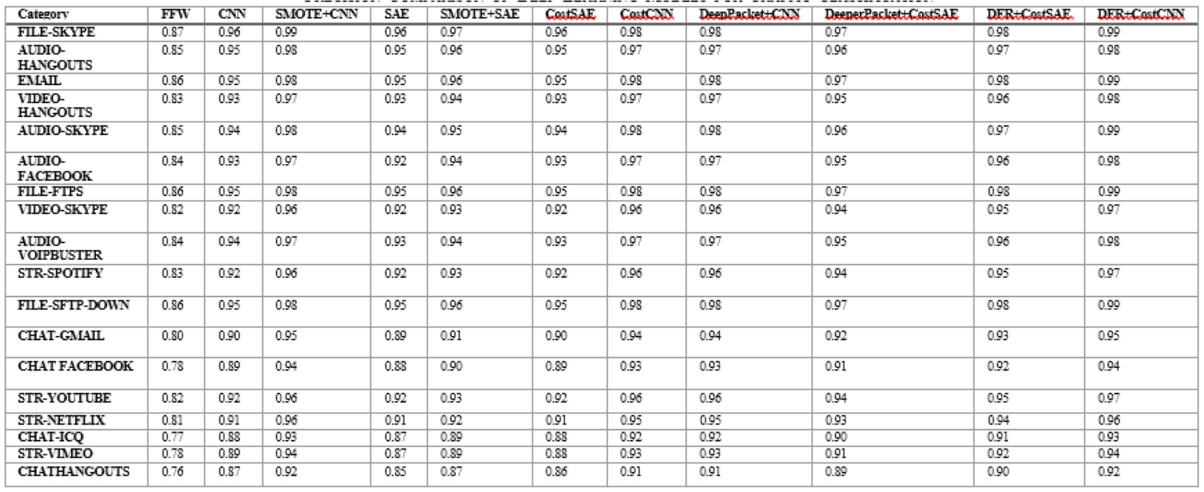
TABLE IV: Recall Comparison of Deep Learning Models for Traffic Classification

| Category | FFW | CNN | SMOTE+CNN | SAE | SMOTE+SAE | CostSAE | CostCNN | DeepPacket+CNN | DeeperPacket+CostSAE | DFR+CostSAE | DFR+CostCNN |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FILE-SKYPE | 0.79 | 0.96 | 0.98 | 0.94 | 0.95 | 0.94 | 0.97 | 0.97 | 0.96 | 0.97 | 0.98 |
| AUDIO-HANGOUTS | 0.78 | 0.95 | 0.98 | 0.93 | 0.94 | 0.93 | 0.97 | 0.97 | 0.95 | 0.96 | 0.98 |
| EMAIL | 0.80 | 0.95 | 0.98 | 0.94 | 0.95 | 0.94 | 0.97 | 0.97 | 0.96 | 0.97 | 0.98 |
| VIDEO-HANGOUTS | 0.77 | 0.93 | 0.97 | 0.92 | 0.93 | 0.92 | 0.96 | 0.96 | 0.94 | 0.95 | 0.97 |
| AUDIO-SKYPE | 0.78 | 0.94 | 0.98 | 0.93 | 0.94 | 0.93 | 0.97 | 0.97 | 0.95 | 0.96 | 0.98 |
| AUDIO-FACEBOOK | 0.76 | 0.93 | 0.97 | 0.91 | 0.92 | 0.91 | 0.96 | 0.96 | 0.93 | 0.94 | 0.96 |
| FILE-FTPS | 0.80 | 0.95 | 0.98 | 0.94 | 0.95 | 0.94 | 0.97 | 0.97 | 0.96 | 0.97 | 0.98 |
| VIDEO-SKYPE | 0.75 | 0.92 | 0.96 | 0.90 | 0.92 | 0.91 | 0.96 | 0.96 | 0.93 | 0.94 | 0.96 |
| AUDIO-VOIPBUSTER | 0.78 | 0.94 | 0.97 | 0.92 | 0.93 | 0.92 | 0.96 | 0.96 | 0.94 | 0.95 | 0.97 |
| STR-SPOTIFY | 0.76 | 0.92 | 0.96 | 0.90 | 0.92 | 0.91 | 0.95 | 0.95 | 0.93 | 0.94 | 0.96 |
| FILE-SFTP-DOWN | 0.80 | 0.95 | 0.98 | 0.94 | 0.95 | 0.94 | 0.97 | 0.97 | 0.96 | 0.97 | 0.98 |
| CHAT-GMAIL | 0.72 | 0.90 | 0.95 | 0.86 | 0.88 | 0.87 | 0.92 | 0.92 | 0.90 | 0.91 | 0.93 |
| CHAT-FACEBOOK | 0.70 | 0.89 | 0.94 | 0.85 | 0.87 | 0.86 | 0.91 | 0.91 | 0.89 | 0.90 | 0.92 |
| STR-YOUTUBE | 0.75 | 0.92 | 0.96 | 0.90 | 0.92 | 0.91 | 0.95 | 0.95 | 0.93 | 0.94 | 0.96 |
| STR-NETFLIX | 0.73 | 0.91 | 0.96 | 0.88 | 0.90 | 0.89 | 0.94 | 0.94 | 0.92 | 0.93 | 0.95 |
| CHAT-ICQ | 0.69 | 0.88 | 0.93 | 0.84 | 0.86 | 0.85 | 0.90 | 0.90 | 0.88 | 0.89 | 0.91 |
| STR-VIMEO | 0.70 | 0.89 | 0.94 | 0.85 | 0.87 | 0.86 | 0.91 | 0.91 | 0.89 | 0.90 | 0.92 |
| CHAT-HANGOUTS | 0.68 | 0.87 | 0.92 | 0.83 | 0.85 | 0.84 | 0.89 | 0.89 | 0.87 | 0.88 | 0.90 |

The present study assesses a variety of deep learning models for network traffic classification and provides key performance metrics: AUC (Area Under the Curve), categorical accuracy, precision, and recall. Performance metrics show that architectures and data balancing techniques have a considerable impact on model performance. Overall model performance, the CNN SMOTE model achieves the highest AUC (0.9994) and categorical accuracy (0.9857), demonstrating that SMOTE applied to CNN improves classification performance related to class imbalance. High precision (0.9867) and recall (0.9849) suggests that the model effectively identifies positive instances and minimize false negatives. Other models performed well too. DFR (CNN) and Deep Packet CNN also achieve a high AUC (0.9996), with corresponding categorical accuracy values of 97.83% and 97.99%. This also highlights the use of deep feature extraction as a means of distinguishing between network traffic types. Similarly, CostCNN also achieves high scores (AUC = 0.9993, accuracy = 98.04%), and performs particularly well in recall (0.9754), which is an important quality for cybersecurity applications where false negatives are of critical concern. The SAE and SAE SMOTE models performed competitive with AUC scores of 0.9984 and 0.9990. The accuracy of SAE increased from 94.85% to 96.42% when SMOTE is applied, demonstrating the improvement oversampling techniques provide as a method of handling imbalanced data.

Table IV presents the different recall performance of deep learning models for network traffic classification. It can be seen that models with cost-sensitive learning strategies (CostCNN and DFR CostCNN) have the highest recall result across all categories of traffic. Traditional models (i.e., FFW and SAE) have lower recall performance,

TABLE V: PRECISION COMPARISON OF DEEP LEARNING MODELS FOR TRAFFIC CLASSIFICATION



| Category | FFW | CNN | SMOTE+CNN | SAE | SMOTE+SAE | CostSAE | CostCNN | DeepPacket+CNN | DeeperPacket+CostSAE | DFR+CostSAE | DFR+CostCNN |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FILE-SKYPE | 0.87 | 0.96 | 0.99 | 0.96 | 0.97 | 0.96 | 0.98 | 0.98 | 0.97 | 0.98 | 0.99 |
| AUDIO-HANGOUTS | 0.85 | 0.95 | 0.98 | 0.95 | 0.96 | 0.95 | 0.97 | 0.97 | 0.96 | 0.97 | 0.98 |
| EMAIL | 0.86 | 0.95 | 0.98 | 0.95 | 0.96 | 0.95 | 0.98 | 0.98 | 0.97 | 0.98 | 0.99 |
| VIDEO-HANGOUTS | 0.83 | 0.93 | 0.97 | 0.93 | 0.94 | 0.93 | 0.97 | 0.97 | 0.95 | 0.96 | 0.98 |
| AUDIO-SKYPE | 0.85 | 0.94 | 0.98 | 0.94 | 0.95 | 0.94 | 0.98 | 0.98 | 0.96 | 0.97 | 0.99 |
| AUDIO-FACEBOOK | 0.84 | 0.93 | 0.97 | 0.92 | 0.94 | 0.93 | 0.97 | 0.97 | 0.95 | 0.96 | 0.98 |
| FILE-FTPS | 0.86 | 0.95 | 0.98 | 0.95 | 0.96 | 0.95 | 0.98 | 0.98 | 0.97 | 0.98 | 0.99 |
| VIDEO-SKYPE | 0.82 | 0.92 | 0.96 | 0.92 | 0.93 | 0.92 | 0.96 | 0.96 | 0.94 | 0.95 | 0.97 |
| AUDIO-VOIPBUSTER | 0.84 | 0.94 | 0.97 | 0.93 | 0.94 | 0.93 | 0.97 | 0.97 | 0.95 | 0.96 | 0.98 |
| STR-SPOTIFY | 0.83 | 0.92 | 0.96 | 0.92 | 0.93 | 0.92 | 0.96 | 0.96 | 0.94 | 0.95 | 0.97 |
| FILE-SFTP-DOWN | 0.86 | 0.95 | 0.98 | 0.95 | 0.96 | 0.95 | 0.98 | 0.98 | 0.97 | 0.98 | 0.99 |
| CHAT-GMAIL | 0.80 | 0.90 | 0.95 | 0.89 | 0.91 | 0.90 | 0.94 | 0.94 | 0.92 | 0.93 | 0.95 |
| CHAT FACEBOOK | 0.78 | 0.89 | 0.94 | 0.88 | 0.90 | 0.89 | 0.93 | 0.93 | 0.91 | 0.92 | 0.94 |
| STR-YOUTUBE | 0.82 | 0.92 | 0.96 | 0.92 | 0.93 | 0.92 | 0.96 | 0.96 | 0.94 | 0.95 | 0.97 |
| STR-NETFLIX | 0.81 | 0.91 | 0.96 | 0.91 | 0.92 | 0.91 | 0.95 | 0.95 | 0.93 | 0.94 | 0.96 |
| CHAT-ICQ | 0.77 | 0.88 | 0.93 | 0.87 | 0.89 | 0.88 | 0.92 | 0.92 | 0.90 | 0.91 | 0.93 |
| STR-VIMEO | 0.78 | 0.89 | 0.94 | 0.87 | 0.89 | 0.88 | 0.93 | 0.93 | 0.91 | 0.92 | 0.94 |
| CHATHANGOUTS | 0.76 | 0.87 | 0.92 | 0.85 | 0.87 | 0.86 | 0.91 | 0.91 | 0.89 | 0.90 | 0.92 |

especially for minority traffic classes (CHAT-ICQ and CHAT- HANGOUTS) An important finding is that in both cases where SMOTE was applied there was an improvement in the model's performance. CNN SMOTE has an accuracy of 2.49% more than CNN metrics without SMOTE. This means that data balancing did improve a classification task. SAE SMOTE results followed a similar trend. Conversely, FFW had the lowest performance, at an AUC of 0.9920 and an accuracy of 82.11%. This may suggest that simpler architectures are not sufficient for complex network traffic classification.

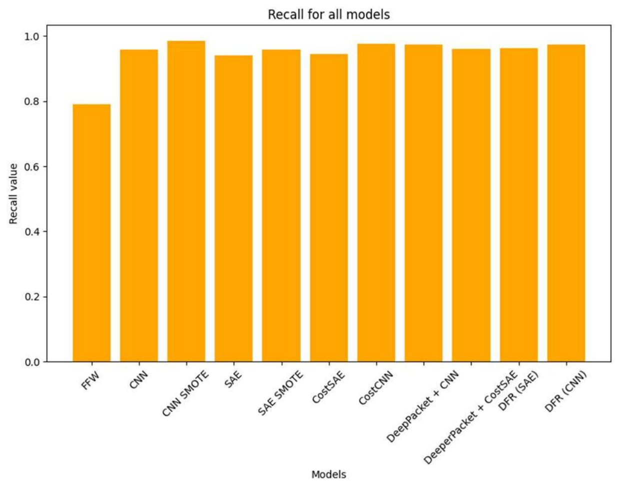


Fig 6. Representation of recall for all models

The performance comparison of all models in terms of precision, recall, and accuracy is visually represented in Fig. 6, Fig. 7, and Fig. 8. These bar graphs provide a clear comparative analysis, highlighting the improvements achieved through cost-sensitive learning and data augmentation techniques

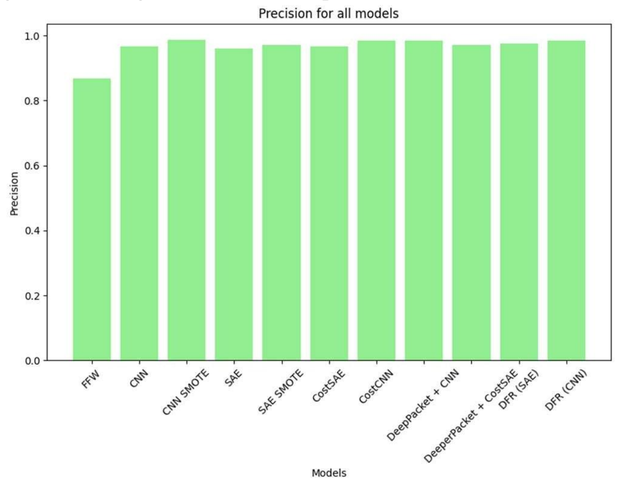


Fig 7. Representation of precision for all models

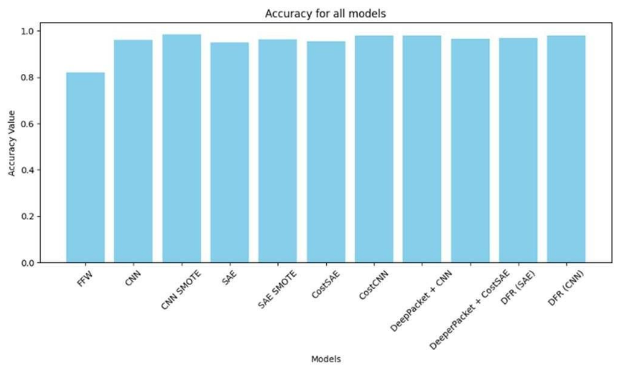
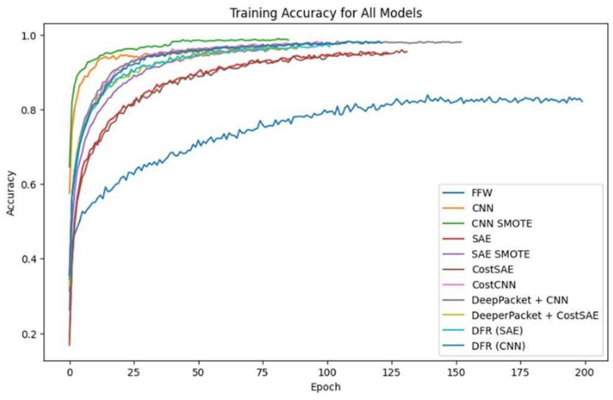


Fig 8. Representation of precision for all models

TABLE VI: PERFORMANCE EVALUATION OF NETWORK TRAFFIC CLASSIFICATION MODELS

| MODEL | AUC | CATEGORICAL ACCURACY | PRECISION | RECALL |
|---|---|---|---|---|
| FFW | 0.9920 | 0.8211 | 0.8681 | 0.7903 |
| CNN | 0.9987 | 0.9608 | 0.9673 | 0.9579 |
| CNN SMOTE | 0.9994 | 0.9857 | 0.9867 | 0.9849 |
| SAE | 0.9984 | 0.9485 | 0.9602 | 0.9406 |
| SAE SMOTE | 0.9990 | 0.9642 | 0.9720 | 0.9584 |
| COSTSAE | 0.9986 | 0.9548 | 0.9660 | 0.9446 |
| COSTCNN | 0.9993 | 0.9804 | 0.9855 | 0.9754 |
| DEEPPACKET + CNN | 0.9996 | 0.9799 | 0.9839 | 0.9738 |
| DEEPERPACKET + COSTSAE | 0.9991 | 0.9668 | 0.9722 | 0.9605 |
| DFR (SAE) | 0.9994 | 0.9676 | 0.9761 | 0.9626 |
| DFR (CNN) | 0.9996 | 0.9783 | 0.9857 | 0.9746 |



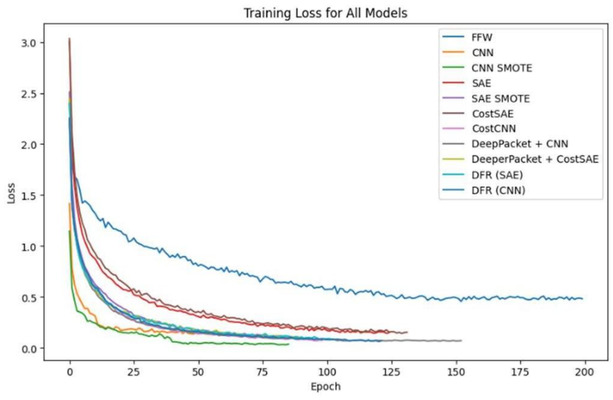


Fig. 9. Training performance of traffic classification models.

The table VI provides a comparative analysis of various deep learning models for network traffic classification. The evaluation metrics include Accuracy, categorical accuracy, precision, and recall, showcasing the effectiveness of different approaches in handling imbalanced data and improving classification performance.

## VI. CONCLUSION

In this study, we examined the efficiency of several deep learning techniques for network traffic classification, partly within a cost-sensitive perspective and partly without this perspective. The results reveal that incorporating cost-sensitive learning and data augmentation techniques improve the overall effectiveness of classification performance, specifically for minority traffic classes. From performance evaluation, the DFR CostCNN and CostCNN models consistently achieved the highest recall and accuracy scores, demonstrating their ability to handle the imbalance of class size. The SMOTE-enhanced CNN and SAE models also made significant improvements and increased the recall values across many traffic categories. Table V indicates that the cost-sensitive models provided better recall, which thereby reduced false negatives by classifying more instances of minority traffic classes. This improvement is particularly noteworthy for minority traffic classes, including CHAT-ICQ, STR-VIMEO, and CHAT-HANGOUTS. In addition, the results highlight the subsequent impact of high frequency classes, such as FILE-SKYPE, AUDIO- HANGOUTS, and VIDEO-SKYPE, associated with cost- insensitive models tending to mask or overshadow minority traffic classes. Incorporating cost-sensitive learning effectively reduced this problem with better classification balance and fewer misclassification across all traffic classes.

Overall, the findings show the benefits of using cost-sensitive learning and data augmentation to improve the reliability and accuracy of network traffic classification systems ture work could dwell on improving deep learning models by incorporating attention mechanisms and graph-based representations that could improve real-time classification capabilities. Ideally, integrating federated learning could provide privacy-invoked network traffic classification without diminishing performance in a distributed setting. Finally, exploring hybrid models that combine traditional machine learning approaches with deep learning models could further improve classification capabilities and interpretability, allowing the model to adapt rapidly to evolving network traffic patterns.

## REFERENCES

[1] H. Yao, P. Gao, J. Wang, P. Zhang, C. Jiang, and Z. Han, “Capsule network assisted IoT traffic classification mechanism for smart cities,” IEEE Internet Things J., vol. 6, no. 5, pp. 7515–7525, Oct. 2019.

[2] S. Rezaei and X. Liu, “Deep learning for encrypted traffic classifica- tion: An overview,” IEEE Commun. Mag., vol. 57, no. 5, pp. 76–81, May 2019.

[3] A. Telikani and A. H. Gandomi, “Cost-sensitive stacked auto-encoders for intrusion detection in the Internet of Things,” Internet Things, vol. 14, Jun. 2021, Art. no. 100122.

[4] A. Telikani, A. Tahmassebi, B. Wolfgang, and A. H. Gandomi, “Evolutionary machine learning: A survey,” ACM Comput. Surveys, vol. 54, no. 8, pp. 11–50, 2021.

[5] Y.-A. Chung, H.-T. Lin, and S.-W. Yang, “Cost-aware pre-training for multiclass cost-sensitive deep learning,” 2015. [Online]. Available: arXiv:1511.09337.

[6] S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P. J. Kennedy, “Training deep neural networks on imbalanced data sets,” in Proc. Int. Joint Conf. Neural Netw., Vancouver, BC, Canada, 2016, pp. 4368–4374.

[7] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, “Cost-sensitive learning of deep feature representations from imbal- anced data,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 8,
pp. 3573–3587, Aug. 2018.

[8] M. L. Wong, K. Seng, and P. K. Wong, “Cost-sensitive ensemble of stacked denoising autoencoders for class imbalance problems in business domain,” Expert Syst. Appl., vol. 141, Mar. 2020, Art. no. 112918.

[9] C. Zhang, K. C. Tan, and R. Ren, “Training cost-sensitive deep belief networks on imbalance data problems,” in Proc. Int. Joint Conf. Neural Netw., Vancouver, BC, Canada, 2016, pp. 4362–4367.

[10] H. Wang, Z. Cui, Y. Chen, M. Avidan, A. B. Abdallah, and A. Kronzer, “Predicting hospital readmission via cost-sensitive deep learning,” IEEE/ACM Trans. Comput. Biol. Bioinf., vol. 15, no. 6, pp. 1968–1978, Nov./Dec. 2018.

[11] Y. Fan, C. Zhang, Z. Liu, Z. Qiu, and Y. He, “Cost-sensitive stacked sparse auto-encoder models to detect striped stem borer infestation on rice based on hyperspectral imaging,” Knowl. Based Syst., vol. 168,
pp. 49–58, Mar. 2019.

[12] M. Terzi, G. A. Susto, and P. Chaudhari, “Directional adversarial training for cost sensitive deep learning classification applications,” Eng. Appl. Artif. Intell., vol. 91, May 2020, Art. no. 103550.

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

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

[15] Y. Zeng, H. Gu, W. Wei, and Y. Guo, "Deep—Full—Range: A deep learning based network encrypted traffic classification and intrusion detection framework," IEEE Access, vol. 7, pp. 45182–45190, 2019.
[16] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and A. V. Vasilakos, "An effective network traffic classification method with unknown flow detec- tion," IEEE Trans. Netw. Service Manag., vol. 10, no. 2, pp. 133–147, Jun. 2013.
[17] Z. Chen, K. He, J. Li, and Y. Geng, "Seq2Img: A sequence-to-image based approach towards IP traffic classification using convolutional neu- ral networks," in Proc. IEEE Int. Conf. Big Data, Boston, MA, USA, 2017, pp. 1271–1276.
[18] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for Internet of Things," IEEE Access, vol. 5, pp. 18042–18050, 2017.
[19] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neu- ral networks," in Proc. IEEE Int. Conf. Intell. Security Informat. (ISI), 2017,
pp. 43–48.
[20] P. Wang, F. Ye, X. Chen, and Y. Qian, "DataNet: Deep learning based encrypted network traffic classification in SDN home gateway," IEEE Access, vol. 6, pp. 55380–55391, 2018.
[21] H. Shi, H. Li, D. Zhang, C. Cheng, and X. Cao, "An efficient feature generation approach based on deep learning and feature selection tech- niques for traffic classification," Comput. Netw., vol. 132, pp. 81–98, Feb. 2018.
[22] Z. Zou, J. Ge, H. Zheng, Y. Wu, C. Han, and Z. Yao, "Encrypted traffic classification with a convolutional long short-term memory neural network," in Proc. IEEE 20th Int. Conf. High Perform. Comput. Commun. IEEE 16th Int. Conf. Smart City IEEE 4th Int. Conf. Data Sci. Syst. (HPCC/SmartCity/DSS), Exeter, U.K., 2018,
pp. 329–334.
[23] R. Hasibi, M. Shokri, and M. Dehghan, "Augmentation scheme for deal- ing with imbalanced network traffic classification using deep learning," 2019. [Online]. Available: arXiv:1901.00204.
[24] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "MIMETIC: Mobile encrypted traffic classification using multimodal deep learning," Comput. Netw., vol. 165, Dec. 2019, Art. no. 106944.
[25] S. Rezaei and X. Liu, "Multitask learning for network traffic classi- fication," in Proc. 29th Int. Conf. Comput. Commun. Netw. (ICCCN), Honolulu, HI, USA, 2020, pp. 1–9.
[26] P. Wang, S. Li, F. Ye, Z. Wang, and M. Zhang, "PacketCGAN: Exploratory study of class imbalance for encrypted traffic classifica- tion using CGAN," in Proc. IEEE Int. Conf. Commun., Dublin, Ireland, 2020, pp. 1–7.
[27] L. Xu, X. Zhou, X. Lin, Y. Ren, Y. Qin, and J. Liu, "A new loss function for traffic classification task on dramatic imbalanced datasets," in Proc. IEEE Int. Conf. Commun., Dublin, Ireland, 2020, pp. 1–7.
[28] X. Ren, H. Gu, and W. Wei, "Tree-RNN: Tree structural recurrent neural network for network traffic classification," Expert Syst. Appl., vol. 167, Apr. 2021, Art. no. 114363.
[29] A. M. Sadeghzadeh, S. Shiravi, and R. Jalili, "Adversarial network traf- fic: Towards evaluating the robustness of deep-learning-based network traffic classification," IEEE Trans. Netw. Service Manag., vol. 18, no. 2, pp. 1962–1976, Jun. 2021.
[30] Z. M. Fadlullah et al., "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," IEEE Commun. Surveys Tuts., vol. 19, no. 4, pp. 2432–2455, 4th Quart., 2017.
[31] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," IEEE Trans. Netw. Service Manag., vol. 16, no. 2, pp. 445–458, Jun. 2019.
[32] H. Huang, H. Deng, J. Chen, L. Han, and W. Wang, "Automatic multi-task learning system for abnormal network traffic detec- tion," Int. J. Emerg. Technol. Learn., vol. 13, no. 4, pp. 4–20, 2018.
[33] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and VPN traffic using time- related," in Proc. 2nd Int. Conf. Inf. Syst. Security Privacy, 2016, pp. 407–414.