A cost-sensitive deep learning based approach for network traffic classification

Akbar Telikani[1], Amir H. Gandomi[2], Kim-Kwang Raymond Choo[3], Jun Shen[1]

[1] School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, 2522, Australia
[2]Faculty of Engineering & Information Technology, University of Technology Sydney, Ultimo, Australia
[3] Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249, USA

E-mails: at952@uowmail.edu.au, gandomi@uts.edu.au, raymond.choo@fulbrightmail.org, jshen@uow.edu.au

Abstract

Network Traffic Classification (NTC) plays an important role in cyber security and network performance, for example in intrusion detection and facilitating higher quality of service. However, due to the unbalanced nature of traffic datasets, NTC can be extremely challenging and can lead to poor classification performance. While existing NTC methods seek to re-balance data distribution through resampling strategies, such approaches are known to suffer from information loss, overfitting, and increased model complexity. To address these challenges, we propose a new cost-sensitive deep learning approach to increase the robustness of deep learning classifiers against the imbalanced class problem in NTC. First, the dataset is divided into different partitions, and a cost matrix is created for each partition by considering the data distribution. Then, the costs are applied to the cost function layer to penalize classification errors. In our approach, costs are diverse in each type of misclassification because the cost matrix is specifically generated for each partition. To determine its utility, we implement the proposed cost-sensitive learning method in two deep learning classifiers, namely: stacked autoencoder and convolution neural networks. Our experiments on the ISCX VPN-nonVPN dataset show that the proposed approach can obtain higher classification performance on low-frequency classes, in comparison to three other NTC methods.

Keywords: Encrypted traffic classification, Class imbalance, Deep learning, Cost-sensitive learning

1. Introduction

Network Traffic Classification (NTC) is an essential task in computer network management, for example, to ensure or improve quality of service, and facilitate accounting and resource usage planning, as well as cyber security (e.g., malware and intrusion detection) [1],[2]. In recent times, several deep learning (DL)-based approaches have been proposed to facilitate NTC in the literature. However, DL models can be overfitted when dealing with unbalanced distribution datasets (i.e., some classes, majority classes, greatly outnumber other classes, minority classes), which is common in traffic data. In this situation, the classifier is biased towards high-frequency traffic, where minority instances are wrongly detected as majority classes. The accuracy of the classifier on the majority

classes is high, but the accuracy is low on the minority classes. This has implications on network resource management, system security, etc. Further complicating the issue, the resampling strategies (e.g., under/oversampling) designed to re-balance data distribution introduce other challenges, such as information loss when removing majority instances, high complexity and being prone to overfitting when generating minority samples [4].

Cost-sensitive learning strategy is a useful strategy to ensuring robustness in DL classifiers against unbalanced datasets, by considering misclassification cost in the training process and subsequently minimizing the cost of DL models [38]. This strategy assigns a higher cost to the minority instances as the majority objects. The class-specific costs are integrated with the loss function of DL classifiers. The first work that investigates cost-sensitive DL was presented by Chung et al. [18], who applied costs in the pretraining phase of deep neural network and Convolution Neural Networks (CNN) classifiers. Wang et al. [9] improved the Mean Squared Error (MSE) loss function for deep neural network by considering the average error in each class. Khan et al. [8] formulated an automated adjustment technique for class-dependent costs using data statistics, instead of handcrafted cost matrix. Three cost-sensitive loss functions (i.e., MSE, cross-entropy, and SVM Hinge) were also introduced. Telikani and Gandomi [4] enhanced the cost function layer of stacked autoencoder (SAE) for intrusion detection in IoT. Several other cost-sensitive DL approaches, such as evolutionary cost-sensitive approaches, have been proposed in the literature [6],[7],[10],[19]. Examples of such efforts include approaches designed to handle class imbalance in intrusion detection [4], rice borer infestation detection [5], hospital readmission prediction [10], and image recognition [11]. To the best of our knowledge, there is no existing work that employs cost-sensitive DL for NTC. Hence, in this paper we propose a cost-sensitive DL framework to improve the performance of encrypted NTC on unbalanced traffic data.

To deal with class imbalance problem in NTC, we develop a new cost-sensitive DL framework. In this framework, cross-entropy loss function layer is optimized by considering misclassification costs to make DL classifiers more effective in detecting low-frequency attacks. To learn robust feature representations, diverse cost matrices are generated at each epoch based on data statistics. This strategy enables DL models to be trained using different cost values, which leads to more robust DL models on unbalanced datasets. We apply our cost-sensitive learning strategy on CNN and SAE DL classifiers, which are respectively referred to as CostCNN and CostSAE. In these classifiers, modified cross-entropy is employed as the loss function. We then conduct comparative experiments to evaluate the performance of our proposed models with those of SMOTE [22], Deep Packet [3], and Deep-Full-Range (DFR) [12] on the "ISCX VPN-nonVPN" dataset.

The remainder of this paper is organized as follows. Section 2 reviews prior encrypted NTC approaches, before describing the ISCX VPN-nonVPN dataset in Section 3. Section 4 presents our new cost-sensitive DL-based approach, and Section 5 describes the evaluation setup and discusses the

findings. Finally, this paper is concluded in Section 6. A summary of notations used in this paper is presented in Table I.

TABLE I: Summary of notations in alphabetical order.

Notation	Explanation
CNN	Convolutional Neural Networks
CostCNN	Cost-sensitive CNN
CostSAE	Cost-sensitive SAE
CSCE	Cost-Sensitive Cross Entropy
DBN	Deep Belief Networks
DFR	Deep-Full-Range
DL	Deep Learning
FN	False Negative
FP	False Positive
GAN	Generative Adversarial Networks
IoT	Internet of Things
LSTM	Long Short-Term Memory
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NTC	Network Traffic Classification
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SAE	Stacked Auto Encoder
SMOTE	Synthetic Minority Oversampling Technique
TN	True Negative
TP	True Positive

2. Related works

NTC aims to classify traffic flows according to their generation applications. In this regard, supervised learning algorithms are mostly used to train detection models based on labeled training data [33]. The current NTC research trend appears to focus on the application of DL techniques. Table II discusses previous studies on DL-based NTC.

Table II: A summary of works on DL-based network traffic classification

Reference	Classifier	Dataset	Class imbalance strategy
Chen et al. [14]	CNN	Case study dataset	–
Lopez-Martin et al. [15]	CNN, LSTM	RedIRIS from Spain	–
Wang et al. [29]	CNN	ISCX VPN-nonVPN	–
Wang et al. [16]	MLP, CNN, SAE	ISCX VPN-nonVPN	Undersampling to remove instances of the majority classes
Shi et al. [23]	DBN	Cambridge and UNIBS datasets	–
Zou et al. [30]	CNN+LSTM	ISCX VPN-nonVPN	–
Zeng et al. [12]	MLP, CNN, SAE	ISCX 2012	–
Yao et al. [1]	Capsule Network	UTSC-2016	–
Hasibi et al. [25]	Convolutional Recurrent Neural Network	A case study dataset from Iran	A data augmentation technique based on LSTM to generate packets for low-frequency classes
Lotfollahi et al. [3]	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 Conditional GAN
Xu et al. [27]	CNN, LSTM, ResNet	ISCX VPN-nonVPN	Improved cross entropy loss function
Aceto et al. [28]	CNN+LSTM	Mobile datasets	–

Rezaei and Li [31]	CNN	QUIC, ISCX VPN-nonVPN	–
Ren et al. [32]	RNN	ISCX VPN-nonVPN	–
Sadeghzadeh et al. [34]	SAE	ISCX VPN-nonVPN	Undersampling
Aceto et al. [35]	SAE, LSTM, and CNN	Human users activity	–
Huang et al. [36]	CNN	CTU-13, ISCX	–
Shi et al. [37]	DBN	Cambridge, UNIBS	–
Our proposed approach	CNN, SAE	ISCX VPN-nonVPN	Cost-sensitive DL

DL has achieved good results in traditional generic NTC, as shown in the literature [13]. The Seq2Img approach [14], 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. [15] 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 [23].

Nevertheless, due to user privacy, security protocols (e.g., HTTPS, SSH, and SSL) are used in most applications to encrypt data traffic. Therefore, encrypted NTC has become an essential task these days. Datanet [16] is an application-aware framework 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. [3], 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) [12] 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 [28] is a multimodal DL framework for encrypted NTC. A 1D CNN was developed by Wang et al. [29] for both the flow-level and session-level classification. Zou et al. [30] stacked 2D CNN and LSTM models for encrypted NTC. Multi-task learning is a recently developed framework for NTC that there is no need for a large labeled traffic dataset [31]. Ren et al. [32] 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. [35] developed a systematic framework to classify mobile traffic classifications using SAE, LSTM, and CNN algorithms. They investigated their frameworks 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. [36] 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 [37] to provide optimal and robust features for NTC.

Class imbalance is a challenging problem in encrypted 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 similar 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 [25]. Xu et al. [27] designed an improved cross-entropy loss function based on the probability obtained from the Softmax layer. Sadeghzadeh et al. [34] 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.

Although there have been various DL-based NTC approaches, only some previous works addressed the class imbalance problem. The approaches applied either undersampling technique [16][3] or synthesized traffic generation for the minority classes [25][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 [4]. None of the related works ever employed a cost-sensitive learning strategy in DL models for addressing the class imbalance problem. In this paper, we proposed a cost-sensitive DL framework that improves cross-entropy loss function using cost-specific weights related to each class.

3. Dataset description

In this study, we used the ISCX VPN-nonVPN dataset [20], consisting of 24 different types of traffic categories. Traffic data was collected in the form of *pcap* files, which include the packets produced by end-user applications (e.g., Facebook and Gmail) and the activities engaged by the user during the capture session (e.g., chat, email, and file transfer). This dataset also includes packets generated in Virtual Private Network (VPN) sessions. A VPN is private communication between different parties across a public network (e.g., the Internet) through an encrypted layered tunneling protocol. Similar to non-VPN traffic, VPN traffic is applied to different applications and activities. For example, packets

produced by the Tor browser, which is used for Twitter, Google, and Facebook, are another type of traffic data included in the ISCX VPN-nonVPN dataset. This browser is used to protect users against Internet monitoring (i.e., traffic analysis) because it not only encrypts connections through relays in the network but also follows a sophisticated port opacity mechanism for privacy and anonymity. Therefore, detection of these packets is a challenging task in NTC.

Table III presents the data distribution of each class included in the dataset. All *pcap* files are labeled based on the application that produced the traffic data, resulting in 12 classes. Seven classes are minority (i.e., AIM chat, Email, Gmail, ICQ, Spotify, Torrent, Vimeo) and other five classes have high distribution and are considered the majority classes. A total of 18M traffic packets were collected from the *pcap* files.

Table III Data distribution of ISCX VPN-nonVPN dataset

Class name	Size	Ratio
AIM chat	5K	0.000275
Email	28K	0.001542
Facebook	2502K	0.13776
FTPS	7872K	0.433432
Gmail	12K	0.000661
Hangouts	3766K	0.207356
ICQ	7K	0.000385
Skype	2872K	0.158132
Spotify	40K	0.002202
Torrent	70K	0.003854
VoIP buster	842K	0.046361
Vimeo	146K	0.008039

4. Cost-sensitive encrypted traffic classification

In this section, we present a cost-sensitive DL approach for managing the class imbalance problem in encrypted NTC. Fig. 1 shows the framework of our proposed method, consisting of four main phases: preprocessing, cost matrix generation, DL model, and cost-sensitive loss function (see also Sections 4.1 to 4.4).

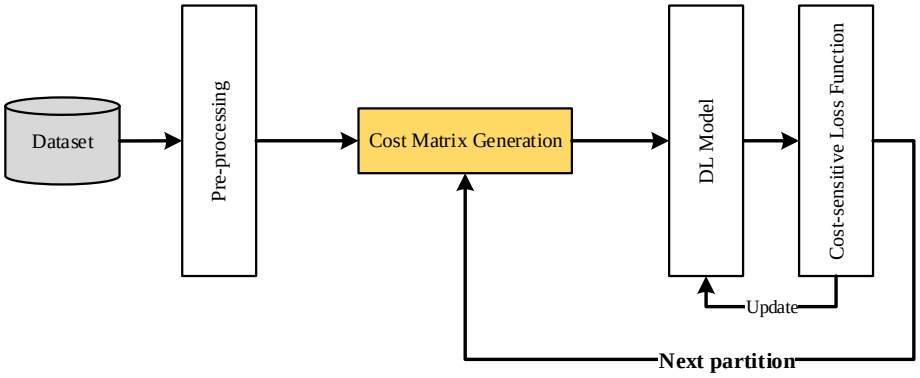


Fig. 1 Proposed cost-sensitive deep learning framework

4.1. Preprocessing

In this section, a preprocessing process consisting of six essential steps, as explained below and in Fig. 2 to provide appropriate input data for the DL classifiers. Each *pcap* file is preprocessed through these steps and then the processed traffic data files are combined to generate a single dataset at the integration stage.

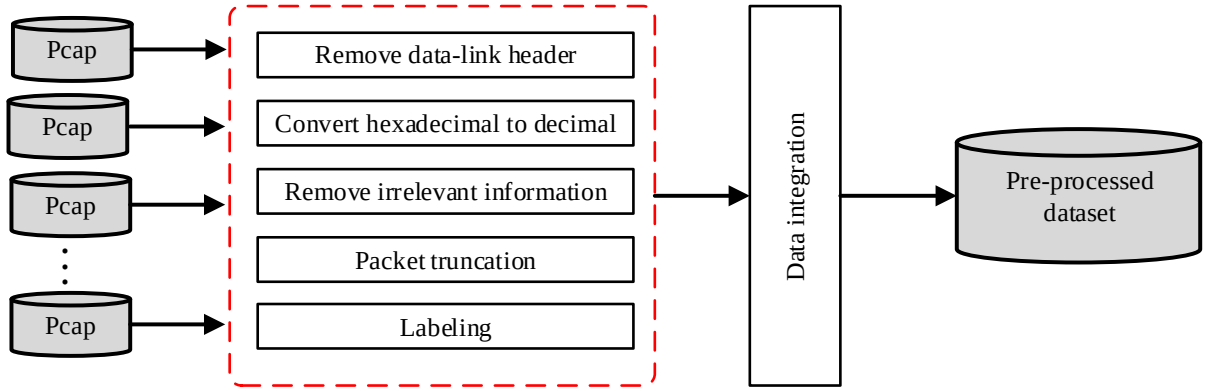


Fig. 2 Preprocessing of data packets

(1) Remove data-link header: Since data packets were produced at the data-link layer, each packet includes a data-link header that contains the source and destination addresses as well as IP address, but this is not informative for the classification task. Thus, we remove the Ethernet header that is 14 bytes.

(2) Convert hexadecimal to decimal: All values of the dataset are in hexadecimal form (i.e., a number between "00" and "*ff*"). To provide appropriate values for DL classifiers, all values should be converted to a value between "0" and "255".

(3) Remove irrelevant information: The dataset consists of TCP segments with SYN, ACK, or FIN flag sets. These segments are necessary for the handshaking process and when a connection is established or finished. We discard these segments because they do not carry any valuable information for classification.

(4) Packet uniformity: DL models need a fixed-length input, whereas the size of packets varies. Therefore, the length of all packets should be made uniform via cutting and padding techniques to be fixed to 1480 bytes.

(5) Labeling: Each *pcap* file has been labeled according to their applications. All packets are given a label according to the application type (e.g., Skype, Facebook, and Gmail). The labeling resulted in a 17-class classification.

(6) Data integration: All prepared files are first combined into one dataset, and then, a dataset with a total of 19M samples is generated. However, this huge number of samples requires big data processing technologies. Therefore, we selected only 10% of the samples for each class and obtained a dataset with 1.9M samples.

4.2. Cost matrix generation

To train the DL model with different costs, we introduce a cost matrix generation process that aims to generate diverse cost matrices. Fig. 3 depicts our developed cost matrix generation procedure, in which the dataset was divided into different partitions, and a cost matrix was generated for each partition.

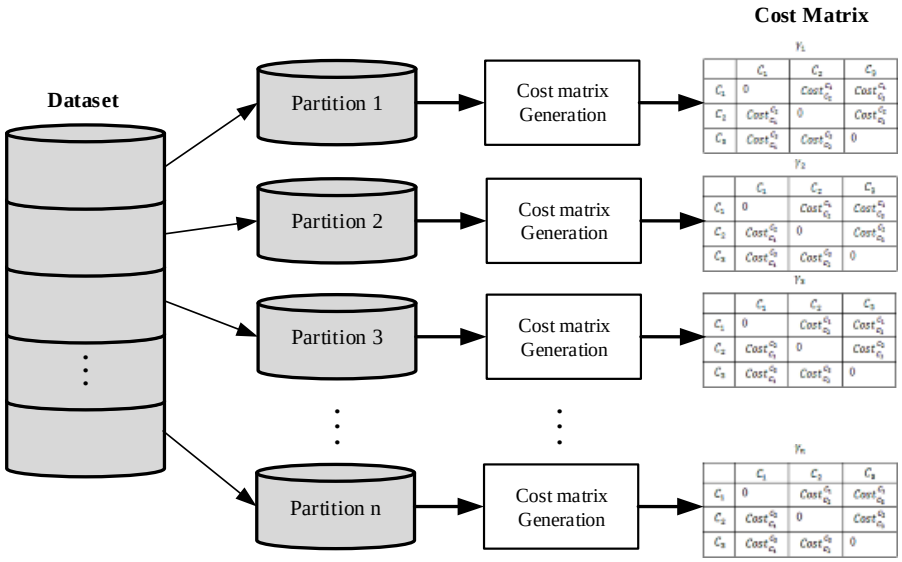


Fig. 3 Cost matrix generation using different partitions

To generate a cost matrix, a heuristic is formulated based on data distribution. A higher cost is assigned to the misclassification of minority classes compared to the cost of misclassifying the majority classes. In this procedure, pairwise comparisons are adopted between traffic classes. The cost value of each misclassification between two classes is determined based on their distribution than the total distributions of the two classes. The misclassification cost of class *i* in class *j* is computed using Eq. (1). Table IV shows an example of a cost matrix for the three-class classification.

TABLE IV: An example of cost matrix with 3 classes

	Predicted C_1	**Predicted C_2**	**Predicted C_3**
Actual C_1	0	$\gamma_{1,2}$	$\gamma_{1,3}$
Actual C_2	$\gamma_{2,1}$	0	$\gamma_{2,3}$
Actual C_3	$\gamma_{3,1}$	$\gamma_{3,2}$	0

$$\begin{cases} \gamma_{i,j} = \frac{\alpha_i}{\alpha_i + \alpha_j} \\ subject\ to\ i \neq j \end{cases} \quad i, j = 1,2, \dots, C \tag{1}$$

In the above equation, α_i and α_j are the number of instances of class *i* and class *j*, respectively. The practical steps of the cost matrix generation phase are summarized in Algorithm 1.

Algorithm 1: Cost matrix generation
Input: y_train, n_classes

Output: cost_matrix γ

1: **Begin**

2: $\gamma \leftarrow$ Initialize with zeros
3: $\alpha \leftarrow$ Compute frequency of classes
4: **For each** $i \in$ labels
5: **For each** $j \in$ labels
6: **if** i≠j
7: $\gamma_{i,j} = \frac{\alpha_i}{\alpha_i + \alpha_j}$

4.3. Two architectures of deep learning models

Stacked auto-encoder (SAE): Our SAE architecture consists of stacking two AEs made up of two encoding layers and two decoding layers (Fig. 4). The input layer has a size of 1480, and the output layer has a size of 17 for application identification.

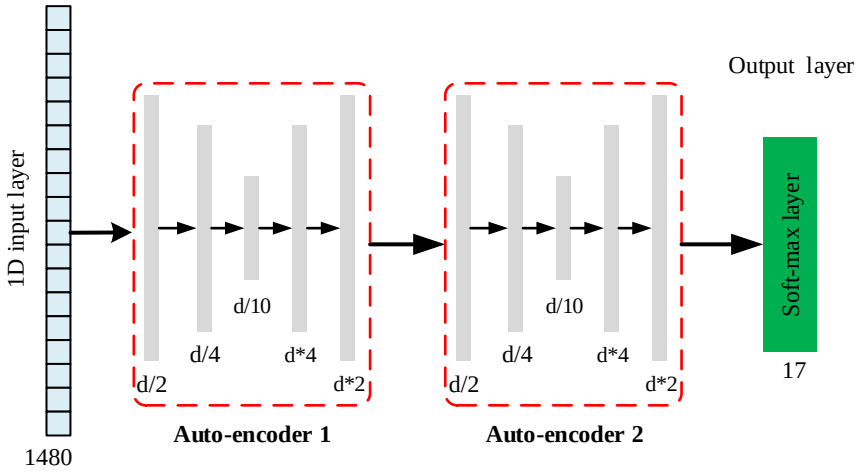


Fig. 4 Our CostSAE architecture for network traffic classification

Convolutional neural network (CNN): We designed a CNN architecture with a one-dimensional input layer (Fig. 5) and three convolution layers, each of which has a convolution followed by ReLU and max-pooling layers. The filter size of the convolution layer is 8*1 and stride=1, and each maxpooling layer processes a 4*1 input with stride=2. After each ReLu layer, we used batch normalization and a dropout with a ratio of 0.05. After the convolution layers, two dense layers were applied for NTC tasks.

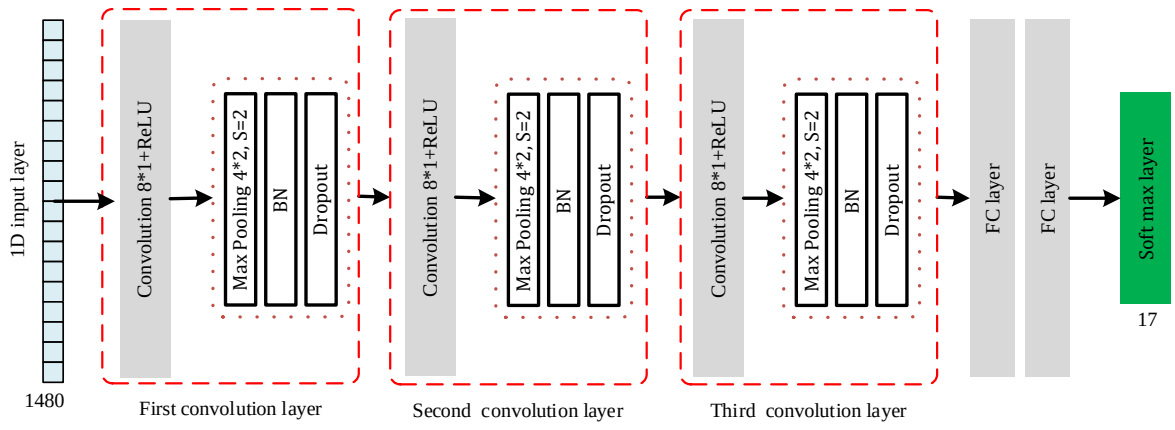


Fig. 5 Our CostCNN architecture for network traffic classification

4.4. Cost-sensitive loss function

To tackle the class imbalance problem during the process of feature learning, we developed a cost-sensitive strategy for DL models. As elaborated earlier, this approach aims to modify the cross-entropy loss function by considering the cost values related to each type of misclassification. This approach makes DL models more sensitive to the misclassification of the minority class. Indeed, the output of the Softmax layer that is the form of probabilities feeds to the loss function in order to compute cost-sensitive loss value. The reason behind the selection of the cross-entropy is that it can outperform other loss functions in most cases and the cross-entropy can prevent learning from slowing down the problem of the mean squared error loss.

Before further describing our cost-sensitive DL strategy, we discuss how a Softmax layer works. Assume the output layer is $\{X, Y\} = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_C)\}$, where $x_i \in \mathbb{R}^{d\times 1}$ and $y_i \in \mathbb{R}^{C\times 1}$ (d is the size of the last layer and C is the number of classes). The Softmax function computes the probability of the object i (x_i) belonging to each class, as expressed bellows:

$$f_\theta(x) = \frac{1}{\sum_{j=1}^{C} e^{\theta_j^T x_i}} \begin{bmatrix} e^{\theta_1^T x_i} \\ e^{\theta_2^T x_i} \\ \dots \\ e^{\theta_C^T x_i} \end{bmatrix} = \begin{bmatrix} p(y_i = 1|x_i; \theta_1) \\ p(y_i = 2|x_i; \theta_2) \\ \dots \\ p(y_i = C|x_i; \theta_C) \end{bmatrix} \quad (2)$$

where θ is the parameter mapping towards the j–the class (*s.t.* $b_j + W_j x$).

Our proposed approach aims to punish the misclassification errors in the cross-entropy cost function based on the costs determined in the cost matrix (γ) to maximize the closeness of the prediction to the desired output. Total lost value of each batch with N training samples is computed using Eq. 3:

$$\mathcal{L}(O, y) = -\frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(O_i, y_i) \quad (3)$$

In the above equation, O stands for the computed probability of outputs via Softmax layer, y represents the true class labels, O_i is the output probability for the sample i, and y_i is the actual label for the sample i. Cross-entropy value is the mean of loss values for all N training samples. Lost value of each prediction is calculated by Eq. 4:

$$\mathcal{L}(O_i, y_i) = -\sum_{i=1}^{C} \left(y_{o,c} \log p(y_i = 1|x_i; \theta_i)\right) \quad (4)$$

In the above equation, $y_{o,c}$ is a binary indicator (0 or 1) referring to the correct prediction for observation o. The value $y_{o,c}$ is 1 for the incorrectly predicted class and is 0 for the actual class. The probability of a misclassified class is changed by incorporating the corresponding class-dependent cost (Eq. 5):

$$p(y_i = 1|x_i) = \frac{\gamma_{i,j}.\exp(O_i)}{\sum_{i=1}^{C} \exp(O_i)} \quad (5)$$

Based on Eq. 5, multiplying the cost associated with the minority classes reduces the new probability value sharply and, thus, it leads to an increase in the loss value of the classification in Eq. 4. In this way, the minority classes influence more on the loss function than the majority ones.

Algorithm 2 presents the pseudo-code of the improved Cost-Sensitive Cross-Entropy loss function (CSCE) designed for our cost-sensitive DL approach.

Algorithm 2: Cost-sensitive cross-entropy (CSCE)

Input: cost matrix ($\boldsymbol{\gamma}$), Actual values ($\boldsymbol{y_A}$), Predicted values ($\boldsymbol{y_p}$)
Output: Loss value $\mathcal{L}$

Begin
  $\mathcal{L} \leftarrow 0$
  For each $i \in N$
    $\boldsymbol{loss_i} = \boldsymbol{y_{Ai}} + \mathbf{log}(\boldsymbol{y_{pi}} \times \boldsymbol{\gamma_{i,j}})$
    $\mathcal{L} \leftarrow \mathcal{L} + \boldsymbol{loss_i}$
  Return $\mathcal{L}$/N
End

5. Experimental results

In this section, the performances of our proposed cost-sensitive DL models (i.e., CostSAE and CostCNN) are compared with those of SMOTE [22], Deep Packet [3], and DFR [12]. Keras library and Tensorflow were used as the backend for implementing the DL models. All models were trained with 100 epochs. An early stopping strategy was employed to avoid the overfitting problem, in which the training process stops when the loss value on the validation data is not changed for several epochs. We applied Adam as the optimizer for the neural networks. In all experiments, 80% of data was used as the training set, 10% as the validation set, and 10% as testing sets. For coding, we usedp python and the evaluations were conducted on a platform with the following configuration: an Intel(R) Xeon(R) CPU with 2.2 GHz and 13 GB memory.

5.1. Evaluation measures

We used four classification metrics, including *accuracy* (Eq. 6), *recall* (Eq. 7), *precision* (Eq. 8), and *F1-score* (Eq. 9), to evaluate the encrypted NTC approaches.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{6}$$

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

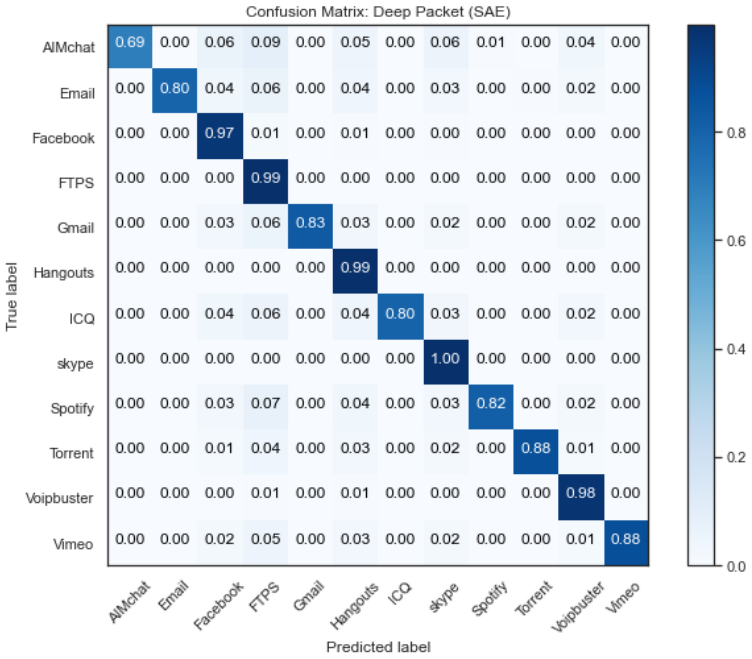
$$Precision = \frac{TP}{TP+FP} \tag{8}$$

$$F1\text{-}Score = \frac{2.\mathrm{Recall}.\mathrm{Precision}}{\mathrm{Recall}+\mathrm{Precision}} \tag{9}$$

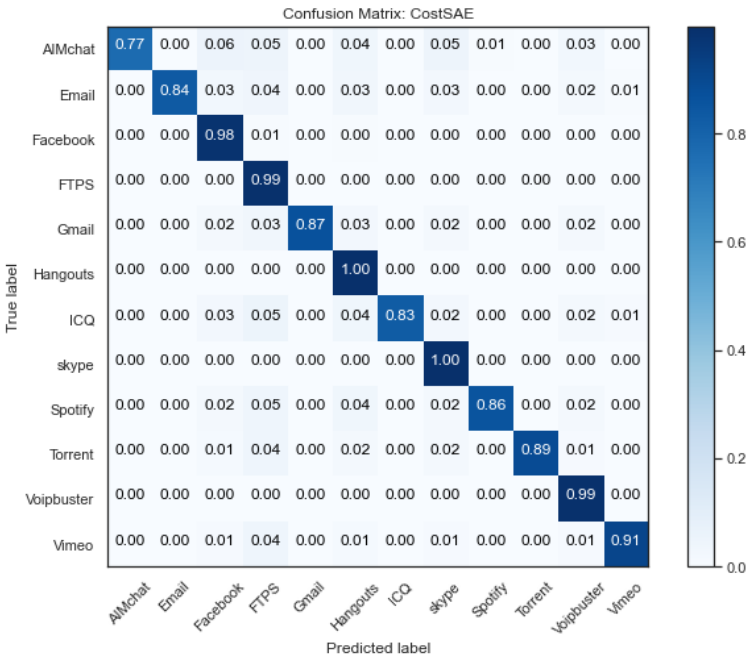
In these equations, *TP*, *FP*, *TN*, and *FN* indicate *True Positive*, *False Positive*, *True Negative, and False Negative*, respectively.

5.2. Results and discussion

Fig. 6 and 7 illustrate the confusion matrices of our cost-sensitive DL models in comparison with their cost-insensitive versions. The confusion matrices of Deep Packet (SAE) and CostSAE are provided in Fig. 6, whereas the matrices associated to Deep Packet (CNN) and CostCNN are compared in Fig. 7. The results show the classes of Facebook, FTPS, Hangouts, and Skype, which have high number of instances in the dataset, have noticeable negative effect on the prediction of the minority classes (i.e., AIM chat, Email, Gmail, ICQ, Spotify, Torrent, and Vimeo). Our approaches are able to lessen the number of false predictions, particularly for low-frequency traffic instances. Additionally, the number of true positives for all classes increased.

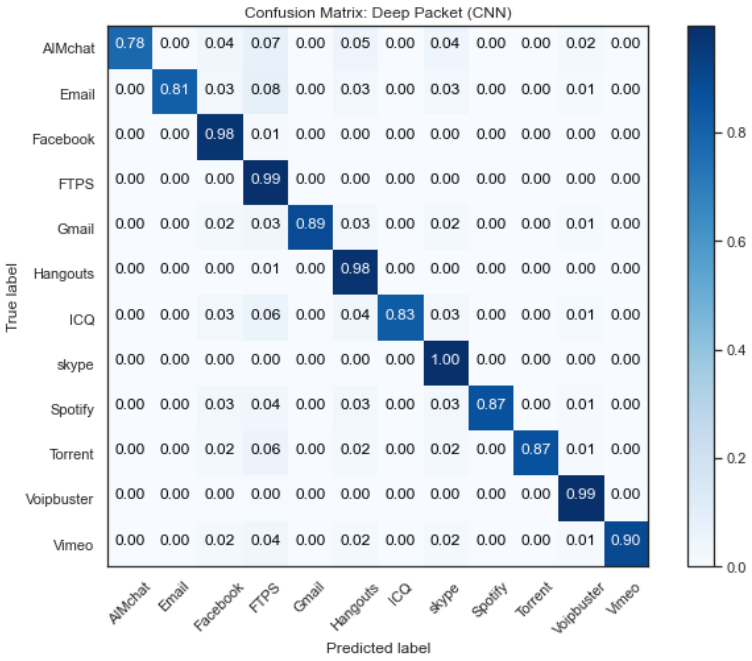


(a) Deep Packet (SAE)

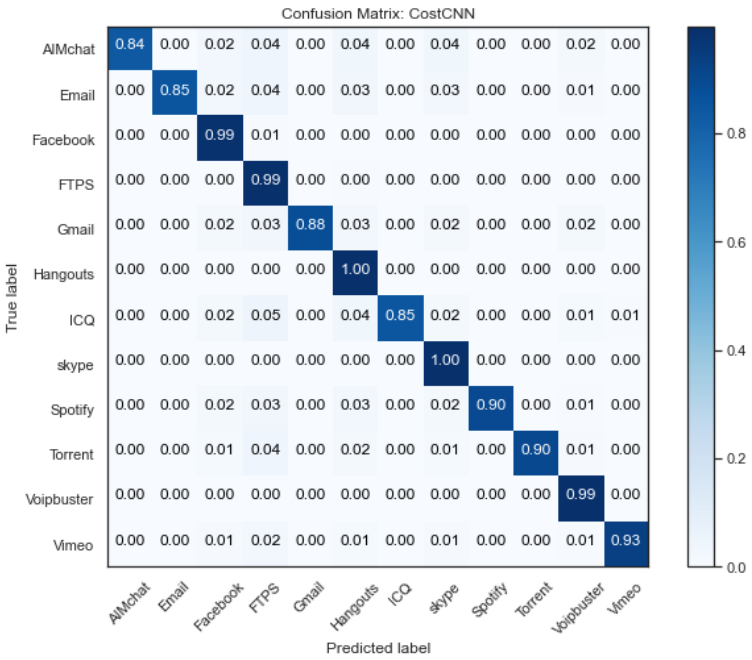


(b) CostSAE

Fig. 6 Confusion matrices of Deep Packet (SAE) and CostSAE



(a) Deep Packet (CNN)



(b) CostCNN

Fig. 7 Confusion matrices of Deep Packet (CNN) and CostCNN

Fig. 8 presents the average performance of DL-based NTC approaches. As can be seen, CostCNN model outperforms the others in terms of all measures, followed by the CostSAE. A significant trend in the results is that CNN-based encrypted NTC models can yield higher performance than other methods, especially for recall measure, that indicates the ability of a classification model in predicting the minority samples. In our approach, there was a noticeable improvement in terms of recall measure which was more than any that was caused by other methods. Indeed, our models are agile in detecting traffic data because they work on the original dataset and handle class imbalance problem during the training phase. Our proposed approach obtains the average results of 98.6%, 91.8%, 97.7%, and 94.7% for accuracy, recall, precision, and F1-Score, respectively. One can observe that our approach performs well in terms of recall measure, because it makes a priority on the minority classes; thus, reducing the number of misclassifications than that of insensitive techniques.

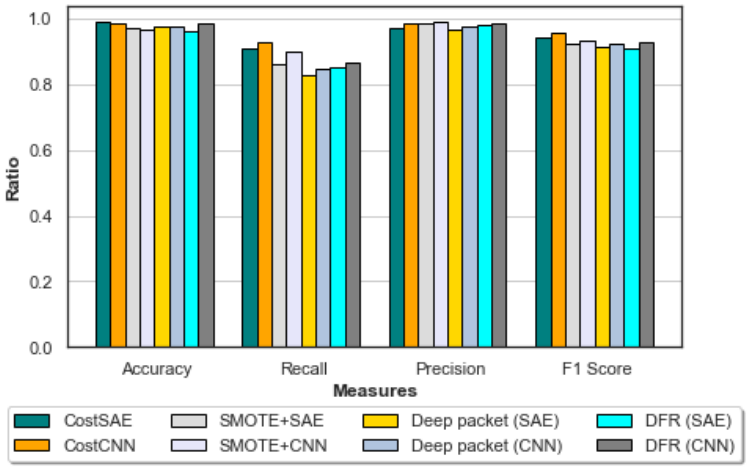


Fig. 8 Performance of traffic classification models

The results have shown that resampling-based models (i.e., SMOTE+SAE and SMOTE+CNN) could obtain the second performance in terms of recall, precision, and F1-Score. However, model training on the balanced datasets generated by SMOTE technique required a huge amount of times because the number of traffic data increased. Fig. 9 has proven issue, where SMOTE+SAE and SMOTE+CNN approaches consume remarkably more runtime than the other DL algorithms. Training SAE and CNN models on the datasets balanced by SMOTE technique requires almost 32,500 seconds and 34,000 seconds, respectively. While, running times of the other DL algorithms were under 28,000 seconds.

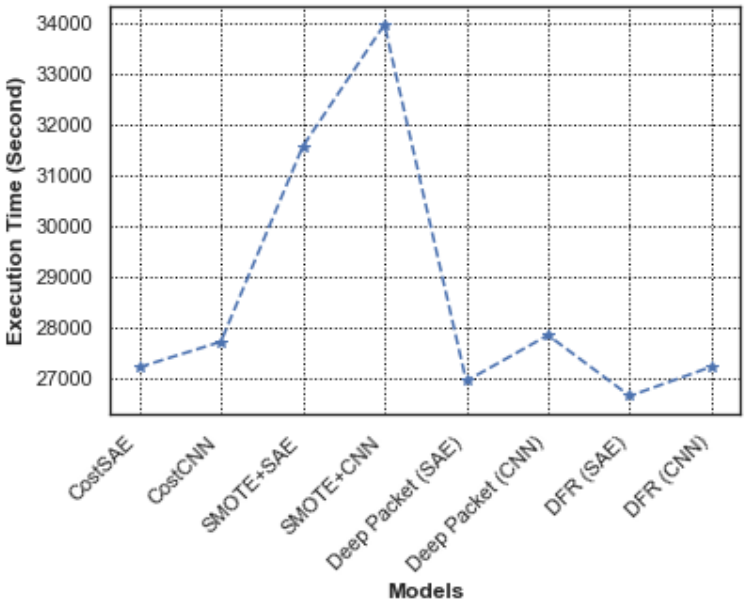


Fig. 9 Execution time of traffic classification models

Table III presents the comparison of recall ratios. Recall is the most important measure for assessing the performance of classification models since the number of the minority instances classified as the majority classes is high (i.e., the number of FN is great for the minority classes). Therefore, the recall ratio for the minority categories is lower than that of the majority ones. Due to the importance of

recall values of minority classes, we evaluated the performance of DL models on low-frequency classes (Fig. 10). The results show that the cost-sensitive DL approach can achieve the highest performance for the low-frequency classes (i.e., AIM chat, Email, Gmail, ICQ, Spotify, Torrent, and Vimeo). This manifests that our models are able to correctly detect the minority classes. It can be seen that CostCNN and CostSAE outperformed other classifiers. Overall, our proposed approach obtained a recall ratio of 86.5% for low-frequency classes on average, followed by DL models on the balanced dataset (82.9%) and Deep Packet (80.5%).

Table III Recall comparison of deep learning models for traffic classification

	CostSAE	CostCNN	SMOTE +SAE	SMOTE +CNN	Deep Packet (SAE)	Deep Packet (CNN)	DFR (SAE)	DFR (CNN)
AIM chat	0.768	0.837	0.694	0.78	0.72	0.8	0.54	0.62
Email	0.837	0.854	0.802	0.815	0.792	0.811	0.728	0.754
Facebook	0.984	0.986	0.97	0.98	0.97	0.976	0.971	0.972
FTPS	0.993	0.991	0.993	0.989	0.997	0.996	0.99	0.992
Gmail	0.871	0.882	0.834	0.852	0.773	0.818	0.842	0.884
Hangouts	0.996	0.996	0.99	0.984	0.996	0.995	0.99	0.991
ICQ	0.826	0.852	0.802	0.828	0.728	0.814	0.671	0.8
Skype	0.997	0.996	0.997	0.996	0.998	0.976	0.977	0.993
Spotify	0.859	0.896	0.816	0.87	0.752	0.793	0.774	0.827
Torrent	0.892	0.902	0.876	0.87	0.874	0.855	0.821	0.821
VoIP buster	0.992	0.992	0.98	0.99	0.99	0.979	0.976	0.989
Vimeo	0.914	0.933	0.878	0.897	0.887	0.864	0.795	0.817
Average	**0.911**	**0.926**	**0.886**	**0.904**	**0.873**	**0.89**	**0.838**	**0.871**

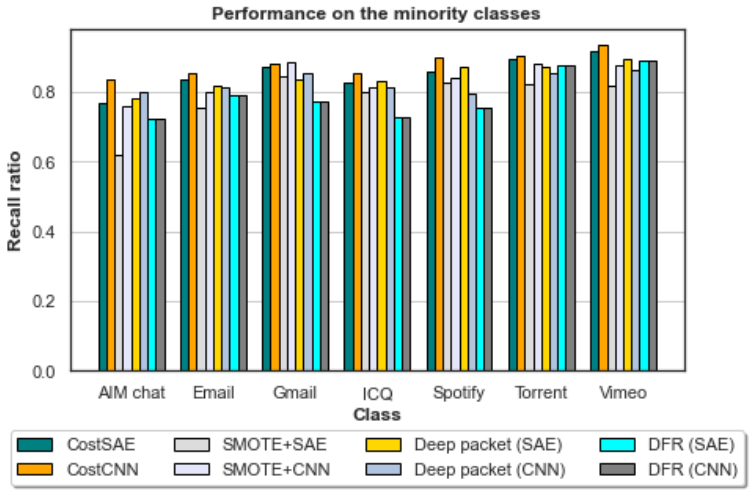


Fig. 10 Performance of traffic classification models for low-frequency classes

When evaluating classification models on unbalanced data using precision criterion, the performance of the minority classes is higher than that of the majority classes because the number of FP increases for the majority class. Thus, classification models are biased towards the majority classes, and instances of the minority classes are classified incorrectly as the majority classes. This issue is

confirmed by the results in Table IV, where the precision of the minority classes is higher than their recall values. According to the results, CostCNN outperformed the other DL models with a precision measure of 96.3%, followed by DFR (CNN) and CostSAE models with precision measures of 95.9% and 95.6%, respectively.

Table IV Precision comparison of deep learning models for traffic classification

	CostSAE	CostCNN	SMOTE +SAE	SMOTE +CNN	Deep Packet (SAE)	Deep Packet (CNN)	DFR (SAE)	DFR (CNN)
AIM chat	0.935	0.967	0.988	0.993	0.96	0.982	0.965	0.988
Email	0.986	0.986	0.99	0.992	0.985	0.993	0.982	0.982
Facebook	0.928	0.99	0.982	0.98	0.933	0.989	0.994	0.987
FTPS	0.998	0.995	0.967	0.976	0.998	0.994	0.988	0.983
Gmail	0.951	0.981	0.992	0.995	0.935	0.988	0.963	0.985
Hangouts	0.993	0.986	0.98	0.984	0.936	0.873	0.985	0.968
ICQ	0.965	0.97	0.99	0.99	0.965	0.986	0.964	0.98
Skype	0.993	0.99	0.986	0.988	0.994	0.986	0.991	0.993
Spotify	0.985	0.984	0.981	0.984	0.983	0.99	0.985	0.994
Torrent	0.99	0.99	0.99	0.996	0.99	0.993	0.987	0.99
VoIP buster	0.937	0.978	0.984	0.986	0.941	0.973	0.993	0.994
Vimeo	0.986	0.995	0.985	0.972	0.989	0.992	0.99	0.99
Average	**0.971**	**0.984**	**0.985**	**0.988**	**0.967**	**0.978**	**0.982**	**0.986**

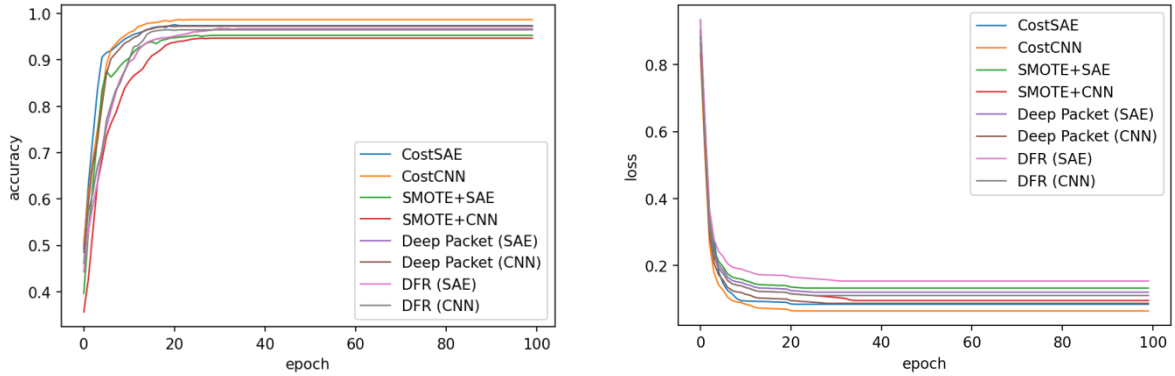
F1-Score is a trade-off between recall and precision measures, in which F-measure evaluates the harmonic mean of these two values. Table V provides a comparison between F1Score values of encrypted NTC methods, which are the average of recall and precision. CostCNN achieved the highest performance with an F1-Score of 0.988. Overall, our models could obtain an average F-measure of 0.986, indicating that cost-sensitive DL approach could optimally train neural networks classifiers considering unbalanced distribution between different classes. In this way, classifiers are able to learn discriminating features from the data to carefully distinguish each class.

Table V F1-Score comparison of deep learning models for traffic classification

	CostSAE	CostCNN	SMOTE +SAE	SMOTE +CNN	Deep Packet (SAE)	Deep Packet (CNN)	DFR (SAE)	DFR (CNN)
AIM chat	0.844	0.898	0.698	0.763	0.806	0.869	0.825	0.884
Email	0.906	0.915	0.839	0.857	0.884	0.895	0.877	0.888
Facebook	0.956	0.988	0.976	0.976	0.951	0.984	0.982	0.981
FTPS	0.996	0.993	0.978	0.984	0.996	0.992	0.992	0.989
Gmail	0.910	0.929	0.869	0.912	0.882	0.936	0.907	0.926
Hangouts	0.995	0.992	0.985	0.987	0.962	0.925	0.99	0.981
ICQ	0.891	0.908	0.8	0.885	0.876	0.9	0.83	0.889
Skype	0.996	0.993	0.981	0.990	0.995	0.991	0.994	0.984

Spotify	0.918	0.938	0.865	0.899	0.892	0.926	0.853	0.882
Torrent	0.939	0.944	0.898	0.900	0.929	0.927	0.927	0.918
VoIP buster	0.964	0.986	0.98	0.987	0.96	0.981	0.991	0.986
Vimeo	0.95	0.964	0.88	0.888	0.93	0.942	0.936	0.923
Average	**0.94**	**0.955**	**0.903**	**0.924**	**0.925**	**0.941**	**0.928**	**0.938**

Fig. 11 illustrates the influence of epoch number on the training accuracy of the encrypted NTC models. It can be observed that cost-sensitive models reached maximum accuracy at epoch 20, approximately 98%. In contrast, the training accuracy of other models has been maximized later with lower ratios, below 97%.



(a) Training accuracy (b) Training loss

Fig. 11 Training accuracy and loss of traffic classification models

6. Conclusion

This study proposed a cost-sensitive DL approach for traffic classification to tackle the class imbalance problem on the detection of low-frequency traffic data. This approach adapts misclassification costs during the training phase to minimize the cost of classifiers. The costs are dynamically assigned based on data distribution of the training set, instead of defining a handcrafted cost matrix by the domain expert judgment. To train DL networks with diverse misclassification costs, the dataset is divided into partitions and a cost matrix is created according to each partition instead of the entire training set. Learning with different costs enables the robustness of DL models against unseen imbalanced datasets. The costs are considered for updating parameters in the fine-tuning phase of classifiers. Our proposed approach was adapted in two widely-used DL classifiers: SAE (CostSAE) and CNN (CostCNN). To show the superiority of our approach over other approaches, the "ISCX VPN-nonVPN" dataset was used. The results have proved that our models can attain high levels of performance, achieving 98.6%, 91.8%, 97.7%, and 94.7% for accuracy, recall, precision, and F1-Score, respectively. The superiority was more obvious for the recall value of the low-frequency classes, which was 86.5% against 82.9%, 80%, and 76.3% for SMOTE, Deep Packet, and DFR, respectively.

In the future, we will explore other approaches, such as evolutionary cost-sensitive DL, for traffic classification to optimize misclassification costs on the training data. We improved cross-entropy loss function for class imbalance, however, other loss functions, such as SVM Hinge Loss and MSE, can be improved using a cost-sensitive strategy. One other research direction involves tuning trained models for other traffic classification tasks with different labeled samples through transfer learning and domain adaptation strategies. We will also focus on how to minimize the computational time for DL-based traffic classification using parallel processing. Our proposed cost-sensitive DL approach can be extended to other application areas associated with class imbalance, such as IoT intrusion detection.

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 of Things Journal, vol. 6, no. 5, pp. 7515–7525, 2019.

[2] S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," IEEE communications magazine, vol. 57, no. 5, pp. 76–81, 2019.

[3] 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 Computing, vol. 24, no. 3, pp. 1999–2012, 2020.

[4] A. Telikani and A. H. Gandomi, "Cost-sensitive stacked auto-encoders for intrusion detection in the internet of things," Internet of Things, p. 100122, 2019.

[5] 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," Knowledge-Based Systems, vol. 168, pp. 49–58, 2019.

[6] M. L. Wong, K. Seng, and P. K. Wong, "Cost-sensitive ensemble of stacked denoising autoencoders for class imbalance problems in business domain," Expert Systems with Applications, vol. 141, p. 112918, 2020.

[7] C. Zhang, K. C. Tan, and R. Ren, "Training cost-sensitive deep belief networks on imbalance data problems," in 2016 international joint conference on neural networks (IJCNN). IEEE, 2016, pp. 4362–4367.

[8] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data," IEEE transactions on neural networks and learning systems, vol. 29, no. 8, pp. 3573–3587, 2017.

[9] S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P. J. Kennedy, "Training deep neural networks on imbalanced data sets," in 2016 international joint conference on neural networks (IJCNN). IEEE, 2016, pp. 4368–4374.

[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 transactions on computational biology and bioinformatics, vol. 15, no. 6, pp. 1968–1978, 2018.

[11] M. Terzi, G. A. Susto, and P. Chaudhari, "Directional adversarial training for cost sensitive deep learning classification applications," Engineering Applications of Artificial Intelligence, vol. 91, p. 103550, 2020.

[12] 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. 45 182–45 190, 2019.

[13] Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2432– 2455, 2017.

[14] Z. Chen, K. He, J. Li, and Y. Geng, "Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks," in 2017 IEEE International Conference on Big Data (Big Data). IEEE, 2017, pp. 1271–1276.

[15] 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. 18 042–18 050, 2017.

[16] 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. 55 380–55 391, 2018.

[17] P. Tapkan, L. O¨ zbakır, S. Kulluk, and A. Baykasog˘lu, "A cost-sensitive classification algorithm: Bee-miner," Knowledge-Based Systems, vol. 95, pp. 99–113, 2016.

[18] Y.-A. Chung, H.-T. Lin, and S.-W. Yang, "Cost-aware pretraining for multiclass cost-sensitive deep learning," arXiv preprint arXiv:1511.09337, 2015.

[19] C. Zhang, K. C. Tan, H. Li, and G. S. Hong, "A cost-sensitive deep belief network for imbalanced classification," IEEE transactions on neural networks and learning systems, vol. 30, no. 1, pp. 109–122, 2018.

[20] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in Proceedings of the 2nd international conference on information systems security and privacy (ICISSP), 2016, pp. 407–414.

[22] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, Journal of Artificial Intelligence Research 16 (2002) 321–357.

[23] Shi, H., Li, H., Zhang, D., Cheng, C., & Cao, X. (2018). An efficient feature generation approach based on deep learning and feature selection techniques for traffic classification. *Computer Networks*, *132*, 81-98.

[25] Hasibi, R., Shokri, M., & Dehghan, M. (2019). Augmentation scheme for dealing with imbalanced network traffic classification using deep learning. *arXiv preprint arXiv:1901.00204*.

[26] Wang, P., Li, S., Ye, F., Wang, Z., & Zhang, M. (2020, June). PacketCGAN: Exploratory study of class imbalance for encrypted traffic classification using CGAN. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)* (pp. 1-7). IEEE.

[27] Xu, L., Zhou, X., Lin, X., Ren, Y., Qin, Y., & Liu, J. (2020, June). A new loss function for traffic classification task on dramatic imbalanced datasets. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)* (pp. 1-7). IEEE.

[28] Aceto, G., Ciuonzo, D., Montieri, A., & Pescapè, A. (2019). MIMETIC: Mobile encrypted traffic classification using multimodal deep learning. *Computer Networks*, *165*, 106944.

[29] Wang, W., Zhu, M., Wang, J., Zeng, X., & Yang, Z. (2017, July). End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)* (pp. 43-48). IEEE.

[30] Zou, Z., Ge, J., Zheng, H., Wu, Y., Han, C., & Yao, Z. (2018, June). Encrypted traffic classification with a convolutional long short-term memory neural network. In *IEEE 20th International Conference on High Performance Computing and Communications* (pp. 329-334). IEEE.

[31] Rezaei, S., & Liu, X. (2020, August). Multitask learning for network traffic classification. In *2020 29th International Conference on Computer Communications and Networks (ICCCN)* (pp. 1-9). IEEE.

[32] Ren, X., Gu, H., & Wei, W. (2021). Tree-RNN: Tree structural recurrent neural network for network traffic classification. *Expert Systems with Applications*, *167*, 114363.

[33] Zhang, J., Chen, C., Xiang, Y., Zhou, W. and Vasilakos, A.V., 2013. An effective network traffic classification method with unknown flow detection. *IEEE Transactions on Network and Service Management*, *10*(2), pp.133-147.

[34] Sadeghzadeh, A.M., Shiravi, S. and Jalili, R., 2021. Adversarial Network Traffic: Towards Evaluating the Robustness of Deep-Learning-Based Network Traffic Classification. *IEEE Transactions on Network and Service Management*, *18*(2), pp.1962-1976.

[35] Aceto, G., Ciuonzo, D., Montieri, A. and Pescapé, A., 2019. Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges. *IEEE Transactions on Network and Service Management*, *16*(2), pp.445-458.

[36] Huang, H., Deng, H., Chen, J., Han, L. and Wang, W., 2018. Automatic Multi-task Learning System for Abnormal Network Traffic Detection. *International Journal of Emerging Technologies in Learning*, *13*(4).

[37] Shi, H., Li, H., Zhang, D., Cheng, C. and Cao, X., 2018. An efficient feature generation approach based on deep learning and feature selection techniques for traffic classification. *Computer Networks*, *132*, pp.81-98.

[38] Telikani, A., Tahmassebi, A., Wolfgang, B., and Gandomi, A., 2021. Evolutionary Machine Learning: A Survey. *ACM Computing Surveys*, 54(8), pp.11-50.