An Effective Cost-Sensitive Convolutional Neural Network for Network Traffic Classification

Mhd Saeed Sharif
Department of Computer Science and Digital Technologies,
ACE, UEL, University Way, London
u1931703@uel.ac.uk

Mina Moein
Department of Computer Science and Digital Technologies,
ACE, UEL, University Way, London
u1931703@uel.ac.uk

Abstract—**The volume, and density of computer network traffic are increasing dramatically with the technology advancements, which has led to the emergence of various new protocols. Analyzing the huge data in large business networks has become important for the owners of those networks. As the majority of the developed applications need to guarantee the network services, while some traditional applications may work well enough without a specific service level. Therefore, the performance requirements of future internet traffic will increase to a higher level. Increasing pressure on the performance of computer networks requires addressing several issues, such as maintaining the scalability of new service architectures, establishing control protocols for routing, and distributing information to identified traffic streams. The main concern is flow detection and traffic detection mechanisms to help establish traffic control policies. A cost-sensitive deep learning approach for encrypted traffic classification has been proposed in this research, to confront the effect of the class imbalance problem on the low-frequency traffic data detection. The developed model can attain a high level of performance, particularly for low-frequency traffic data. It outperformed the other traffic classification methods.**

Keywords— Deep learning, Encrypted traffic classification, Class imbalance, Cost-sensitive learning, Convolutional neural networks.

I. INTRODUCTION

The data flow through a network at any given time is referred to as network traffic, also known as traffic or data traffic. Packets, the smallest and most basic units of data passed over a network, make up network data. For transmission, network traffic data is split into these packets and reassembled at the destination. In other words, the exchange of information between two computers connected to the Internet is referred to as Internet traffic. Given the fast growth in demand for super-power traffic, it is important to know the different categories using the resources of network to properly manage network resources. As a result, accurate classification has become a prerequisite for tasks such as supplying the Quality of Services (QoS), abnormality detection, and value. Network traffic analysis plays a significant role in different problems, such as planning for resource use, evaluating network application performance, controlling the quality of services, and creating a traffic model for research.

Due to advances in Machine Learning (ML), network traffic classification is one of the most widely used areas of ML. ML algorithms can create a data model automatically from a dataset. Generally, there are two types for these algorithms: supervised approach and unsupervised one. In the first approach, there are real results for each instructional instance, while the other algorithm design is used without any prior knowledge of the records that naturally fit in a group. The training approaches are identified into two types of traditional algorithms, for example; support vector machine, and Convolutional Neural Networks (CNN).

Despite their success for traffic classification, ML-based models are ineffective when the data is unbalanced, where some classes significantly outnumber other classes. In this case, ML models are tend to be thwart to classes with majority and cannot correctly detect low-frequency classes. In this paper, a cost-sensitive convolutional neural network is developed to enhance the performance of traffic classification systems on unbalanced datasets. In this strategy, minority classes receive a higher cost than the majority ones. Applying the costs in loss function of CNN model can make the CNN model strong to class imbalance problem in traffic classification. Experiments carried out on ISCX VPN-nonVPN dataset [6] demonstrate superiority of our proposed strategy over other traffic classification models.

This research paper consists of the following parts: part II briefly discusses related literature on traffic classification. In part III, our proposed methodology for cost-sensitive CNN is presented. Results and experimental evaluation are provided in this part IV. Finally, part V shows the conclusions and suggestions for future works.

II. RELATED WORK

Conventional methods for classifying the network traffic are divided in 5 main categories, which include (1) port-based type, (2) load capacity type, (3) pattern identification, (4), statistical, and (5) ML-based. Port-based approaches employ the details in packet headers of TCP/UDP to extract the port number associated with a particular program [7]. Despite their simplicity, these methods cannot provide good accuracy because of the availability of dynamic and private ports. Load capacity techniques are employing the analysis of the application layer and deploy pre-identified patterns, for example regular expressions which are used for each protocol signatures [8]. But, the employment of this format introduces important limitations such as expression limitations and the failure to cope with complex services. Pattern matching methods require reading the contents of packets for comparison between two strings [9]. However, retrieving the encrypted data is hard, these methods are facing different difficulties. Statistical classification methods avoid this problem through employing load capacity characteristics, for example, length, arrival time, and flow length. Deep learning, a sunset of ML is a new trend in traffic classification. Deep Packet method [1] used CNN and SAE algorithms for both traffic description and application identification tasks. In traffic description, network traffic is classified into different activities (such as

FTP, P2P, and chat). In application identification, end-user applications are identified, for example BitTorrent and Skype. This method has the ability to determine encrypted traffic in addition to differentiate between VPN traffic and a non-VPN one. The work in [10] suggested a model for identifying various Google services based on QUIC, including Google Hangout (chat, voice call), file transfer, YouTube, and Google Play music. Seq2Img [11] is an IP traffic classification technique based on converting the stream sequences to images. The images are then classified using CNN. In [12], a combination of Long Short-Term Memory (LSTM) and CNN was developed. The work in [13] combined recurrent neural network (RNN) with CNN to perform traffic classification for Internet of Things (IoT). Datanet [14] used three models, SAE, and CNN to optimize the Software-Defined Network (SDN) for end-to-end network management.

III. Proposed Methodology

A cost-sensitive deep learning model is presented to enhance the performance of detection systems in traffic detection. When data contains traffic that has a very low distribution, conventional systems have trouble detecting it and identify the traffic as normal or high-distributed traffic. This misdiagnosis may cause system malfunctions and poor resource management; Therefore, the focus of the proposed method is on traffic using a cost-sensitive learning strategy. In this strategy, traffic receives a higher cost than repetitive classes. Applying costs while learning Deep learning algorithms increases the power of learning models over traffic detection.

A. Dataset

For this article, the traffic is collected by the Information Security Centre of Excellence, known as the ISCX VPN-nonVPN traffic dataset [6] for public use by researchers. For a given data in this study (ISCX VPN-nonVPN. there are about 20,000 samples for application detection, with the FTPS class having the highest distribution rate (7082 out of 20,000, 35%) and the three classes Hangouts, Spotify, and Facebook having 3766 (19%), 2872, respectively. (14.3%) and 2502 (12.5%) are in the next ranks. Other classes have a rate close to zero, which is very challenging in terms of traffic classification and reduces the quality of services on the Internet. There are 18758 traffic description samples distributed to 14 classes. Distribution rate classes of samples for VOIP, VPN-Browsing, VPN-File Transfer, and VPN-VOIP is higher than the other samples.

B. Proposed methodology

A cost-based learning-based deep learning model is introduced, which includes four steps of data confusion, cost matrix generation, CNN model, and cost-sensitive loss function. Fig. 1 illustrates the steps of the developed approach and each step is discussed in detail below. To train the model at different costs, in each iteration, the data set is divided into several sections and a cost matrix is created corresponding to the relevant section. Because the data is constant at different iterations, the cost matrix for each segment is fixed and therefore the costs are not sufficiently varied. To meet this challenge, the position of records in the data set changes randomly in each iteration before the cost matrix has been developed. In this way, the classes distribution in different parts of the data set changes, leading to a variety of cost matrices

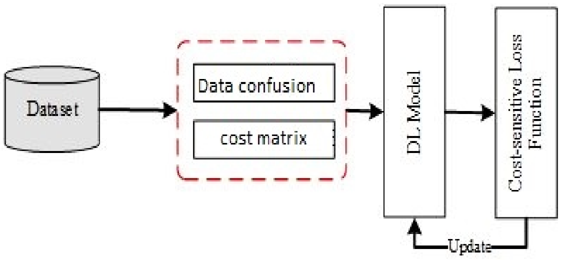

Fig. 1. Proposed method.

C. Cost matrix generation

Creating a cost matrix is essential for training in-depth learning models using costs related to different categories. This matrix is used in the loss function to calculate the amount of classification error. Unlike many previous methods for generating cost matrices, which are manually determined by the specialist for each category, the proposed method uses a revelation to determine costs automatically without user intervention. The costs have been identified by considering the classes distribution. Fig. 2 shows the process of generating the cost matrix γ.

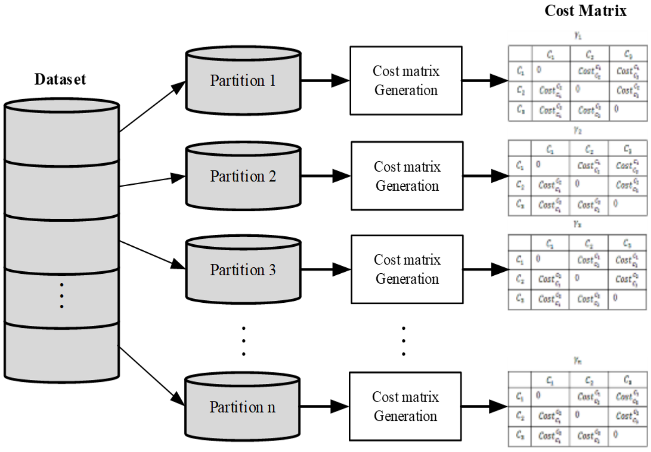

Fig. 2. Cost matrix production process.

In the first step, the distribution of each class in the dataset is calculated to be used to generate a cost matrix. To generate a cost matrix, an equation based on data distribution is performed. Higher classification costs are considered for minority classes, while lower classification costs are set for majority classes. The cost of incorrect classification of class *i* in class *j* is calculated using equation 1.

$$\begin{cases}\gamma_{i,j}=\frac{\alpha_i}{\alpha_i+\alpha_j} & i,j=1,2,\dots,C\\ subject\ to\ i\neq j & \end{cases} \qquad (1)$$

The terms α_i and α_j are the numbers of instances of classes i and j, respectively. The diagonal row of the cost matrix is defined as the utility vector. The utility vector indicates the classifications correction and is set to zero. Also, all costs are non-negative, i.e $\gamma_{i,j} > 0$. The cost matrix shown in Table I are for a three-tier classification. A 3 × 3 matrices are created in such a way that all the cells in the matrix are bigger than zero except those in the diagonal row that is always zero. This means that there is no cost when the sample algorithm classifies correctly. Otherwise, the CSCNN algorithm assigns a cost for incorrect classification depending on the cost allocated in the matrix.

TABLE I. THREE CLASSES OF A COST MATRIX

	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

The pseudo-code corresponding to the cost matrix generation step is shown in Algorithm 1.

```
Algorithm1: Cost matrix generation
Input: y_train, n_classes
Output: cost_matrix $\gamma$

Begin
    $\gamma$ ← Initialize with zeros
    $\alpha$ ← Compute frequency of classes
    For each $i$ ∈ labels
        For each $j$ ∈ labels
            if i≠j
                $\gamma_{i,j} = \frac{\alpha_i}{\alpha_i+\alpha_j}$
End
```

D. *The architecture of CNN model*

This section describes the architectures for CNN model (Fig. 3). The CNN model has a one dimensional input layer and three convolution layers, each of which has a convolution followed by layers of the ReLU activation function and maximum integration. The filter size for the convolution layer is 1 × 8 and stride = 1, and each integration layer processes a maximum of 1 × 4 input with stride = 2. After each ReLu layer, batch normalization and Dropout with a ratio of 0.05 are used. After the convolution layers, two complete connection layers were used to classify the traffic.

E. *Cost-sensitive loss function*

This section proposes a cost-sensitive loss function which has more sensitivity for identifying the incorrect classification of classes (the minority one). During training, the proposed learning method jointly optimizes related classes costs and neural network parameters. Compared to data surface approaches (resampling), the proposed method does not alter the distribution of original data, resulting in lower computational costs during the training process. In addition, unlike cost-sensitive methods, which are determined from an expert cost matrix based on an expert opinion, in the proposed method, the costs associated with each class are amended through the distribution of data throughout the learning process. The article method objective is to penalize all kinds of classification errors based on certain costs. The recompence is higher for the minority class being classified as a majority one than when the majority class being classified as minority class. As mentioned in the previous section, the majority and minority classes are defined to find the associated cost from the matrix. The advantage of the CSCNN approach; there is no need to define the type of classes (if they are minority or majority). In fact, the costs are based solely on the distribution of classes. This feature helps to use the algorithm in any data set. The article method objective is to penalize all types of classification errors based on a specific cost. The recompence is higher when the minority sample is classified as a majority class than when the majority sample is classified as a minority class. As mentioned in the previous section, minority and majority classes are defined and only need to find the corresponding cost from the cost matrix. The advantage of the CSCNN algorithm is that it is not necessary to determine the type of classes in terms of minority or majority. In fact, the costs are based solely on the distribution of classes. This feature helps to use the algorithm in any data set. Before describing the strategy of the cost-sensitive loss function, the SoftMax layer is explained. Suppose the output layer is $\{X, Y\} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_C)\}$, where $x_i \in R^{d\times1}$ and $y_i \in R^{C\times1}$. The term d is the output layer size and C is the classes number. The function calculates the probability of instance i (x_i) associated to a class.

$$f_\theta(x) = \frac{1}{\sum_{j=1}^{C} e^{y_i}} \begin{bmatrix} e^{y_1} \\ e^{y_2} \\ \ldots \\ e^{y_C} \end{bmatrix} = \begin{bmatrix} p(y_i = 1|x_i) \\ p(y_i = 2|x_i) \\ \ldots \\ p(y_i = C|x_i) \end{bmatrix} \qquad (2)$$

Variable θ is the mapping parameter for class j $(b_j + W_j x)$. The proposed approach in this study considers punishing incorrect classification in the cross-entropy loss function based on the costs specified in the cost matrix (γ) to maximize the projected proximity to the actual class. The total cost of each category with N samples is calculated using Eq. 3.

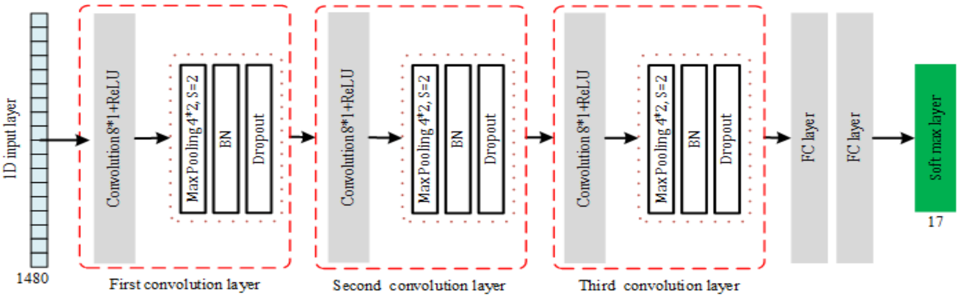

Fig. 3. Cost-Sensitive CNN Architecture

$$\mathcal{L}(O, y) = -\frac{1}{N}\sum_{j=1}^{N} \mathcal{L}(O_j, y_j) \qquad (3)$$

Where the cross-entropy is the average of the loss values for total N classification. The amount of loss for each forecast is calculated by Equation 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 this regard, $y_{o,c}$ is a (0 or 1) index that indicates the right observation prediction for the sample o. The value of $y_{o,c}$ has a value of 1 for the erroneously predicted type and 0 for the real one. The probability of misclassification varies with class cost (Equation 5).

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

According to Equation 5, multiplying the cost of minority classes greatly reduces the amount of new probability and, therefore, leads to an increase in the amount of classification loss in relation 5. Thus, minority classes affect the loss function more than majority classes. Algorithm 2 shows the cost-sensitive cross-entropy cost entropy (CSCE) pseudocode designed for the cost-sensitive CNN model.

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

IV. Experimental results

performance of this article proposed cost-sensitive deep learning model (i.e., CSCNN) is compared with Deep Packet [2] and DFR [3]. Also, a balanced dataset was provided by performing SMOTE, which is an oversampling technique. A CNN model was built using this dataset, which called SMOTE+CNN. Keras library [4] and Tensorflow [5] was used as the backend for implementing deep learning models. All models are trained with 200 epochs. An early stopping strategy is used to keep away the over-fitting problem, in which the training stops when the loss value on the validation data is not changed for several epochs. The CSCNN used *Adam* optimizer for neural networks and loss function for cross-entropy. In all experiments, selected 80% of the data for training, 10% for validation, and 10% for testing. To a balanced distribution between all types of the testing set, 10% of instances of each class are randomly selected rather than selecting 10% of the entire dataset. This model first shows the confusion matrices of the developed models and then the present the models performance of four recall measures, precision, F1, and accuracy. Finally, also investigates the performance of the models at the training phase. Fig.4, Fig. 5 present the confusion matrices of CSCNN, Deep (CNN), DFR (CNN), and SMOTE+CNN models, respectively. In all figures, it is clear that the low-frequency classes (i.e. AIM chat, Email, Gmail, ICQ, Spotify, Torrent, and Vimeo) are often misclassified as the majority classes (i.e. Facebook, FTPS, Hangouts, and Skype). Diagonal entries represent the corrected classifications and non-diagonal entries show the misclassifications. The number of these types of misclassifications is high on unbalanced data and a cost-sensitive model aims to reduce these misclassifications.

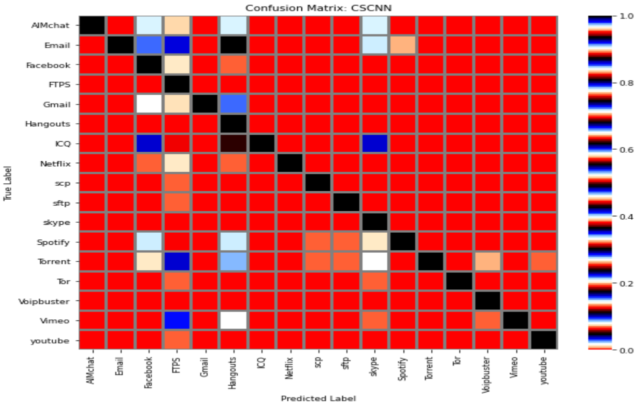

Fig. 4. Confusion matrices of CSCNN model

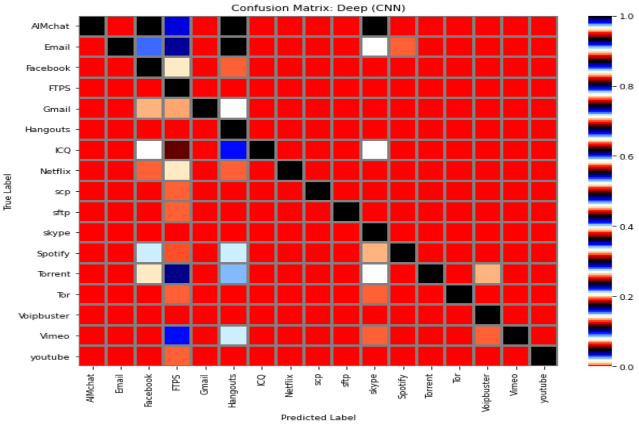

Fig. 5. Confusion matrices of Deep Packet model

Fig. 6 presents the average performance of deep learning-based traffic classification approaches. The CSCNN model outperforms the others regarding all measures, followed by the Deep Packet (CNN). A significant trend in the results is that CNN-based encrypted traffic classification 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.

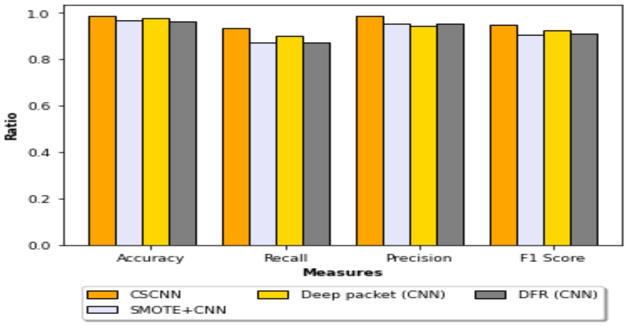

Fig. 6. Models Performance.

Table II presents the comparison of recall ratios. The recall is the best measure for assessing the classification models performance 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. 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 means that the CSCNN model is able to correctly detect the minority classes. It can be seen that CSCNN outperformed other classifiers. Overall, the proposed approach obtained a recall ratio of 0.944 for low-frequency classes on average, followed by Deep Packet (0.929) and DFR (0.923).

TABLE II. Recall comparison of traffic classification models

	CostCNN	SMOTE+CNN	Deep (CNN)	DFR (CNN)
AIM chat	0.83	0.62	0.78	0.8
Email	0.85	0.754	0.815	0.811
Facebook	0.986	0.972	0.98	0.976
FTPS	0.999	0.992	0.997	0.996
Gmail	0.916	0.842	0.89	0.874
Hangouts	0.997	0.991	0.995	0.995
ICQ	0.857	0.8	0.829	0.814
Netflix	0.977	0.954	0.969	0.964
SCP	0.989	0.974	0.988	0.98
SFTP	0.988	0.975	0.987	0.98
Skype	0.998	0.993	0.997	0.976
Spotify	0.9	0.827	0.87	0.793
Torrent	0.9	0.821	0.87	0.855
Tor	0.986	0.936	0.982	0.95
VoIP buster	0.993	0.989	0.99	0.979
Vimeo	0.933	0.817	0.894	0.864
YouTube	0.982	0.971	0.973	0.969
Average	**0.944**	**0.9**	**0.929**	**0.923**

When evaluating classification models on unbalanced data using precision criterion, the performance of the minority classes is higher than that of the majority ones because the FP increases for the majority class. Thus, classification models are thwart to the majority ones, and instances of the minority classes are classified incorrectly as the majority classes. This issue is confirmed by the results in Table III, where the precision of the minority classes (i.e., Chat, Email, Vpn: chat, and Vpn: email) is higher than their recall values. According to the results, CSCNN outperformed the other DL models with a precision measure of 0.994, followed by Deep (CNN) and DFR (CNN) models with precision measures of 0.993 and 0.991, respectively.

F1 is a trade-off between precision and recall measures, F1-Score evaluates the harmonic mean of these two values. Table IV provides a comparison between F1-Score values of traffic classification methods, which are the average of recall and precision. CSCNN achieved the highest performance with an F1-Score of 0.967, indicating that the cost-sensitive CNN approach can optimally train neural networks classifiers considering unbalanced distribution between different classes. In this way, classifiers learn discriminating features from the data to carefully distinguish each class

TABLE III. Precision comparison of deep learning models for traffic classification

	CostCNN	SMOTE +CNN	Deep (CNN)	DFR (CNN)
AIM chat	1.0	1.0	1.0	1.0
Email	1.0	1.0	1.0	1.0
Facebook	0.997	0.98	0.997	0.987
FTPS	0.992	0.976	0.986	0.983
Gmail	1.0	1.0	1.0	1.0
Hangouts	0.992	0.984	0.99	0.968
ICQ	1.0	0.99	1.0	1.0
Netflix	0.992	0.987	0.991	0.991
SCP	0.99	0.98	0.99	0.984
SFTP	0.99	0.984	0.99	0.991
Skype	0.995	0.988	0.993	0.993
Spotify	0.997	0.984	0.995	0.994
Torrent	1.0	1.0	1.0	1.0
Tor	0.991	0.99	0.989	0.991
VoIP buster	0.993	0.986	0.992	0.994
Vimeo	0.99	0.972	0.988	0.99
YouTube	0.986	0.981	0.987	0.99
Average	**0.994**	**0.987**	**0.993**	**0.991**

TABLE IV. F1-Score comparison of traffic classification models

	CostCNN	SMOTE+CNN	Deep (CNN)	DFR (CNN)
AIM chat	0.907	0.837	0.876	0.889
Email	0.919	0.860	0.898	0.953
Facebook	0.991	0.976	0.988	0.981
FTPS	0.995	0.984	0.991	0.989
Gmail	0.956	0.914	0.942	0.933
Hangouts	0.994	0.987	0.992	0.981
ICQ	0.923	0.885	0.907	0.897
Netflix	0.984	0.970	0.980	0.977
SCP	0.989	0.977	0.989	0.982
SFTP	0.989	0.979	0.988	0.985
Skype	0.996	0.990	0.995	0.984
Spotify	0.946	0.899	0.928	0.882
Torrent	0.947	0.902	0.930	0.922
Tor	0.988	0.962	0.985	0.970
VoIP buster	0.993	0.987	0.991	0.986
Vimeo	0.961	0.888	0.939	0.923
YouTube	0.984	0.981	0.980	0.979
Average	**0.967**	**0.987**	**0.959**	**0.955**

Fig. 7 illustrates the epoch number effect of on training of the encrypted traffic classification 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%.

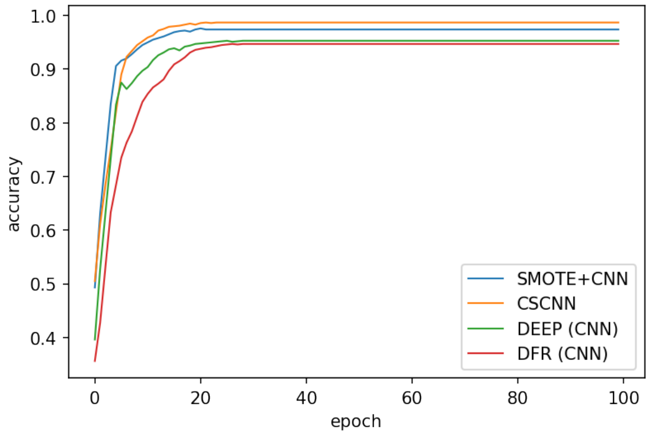

Fig. 7. Training accuracy of traffic classification models

Training Loss of deep learning models for traffic classification is illustrated in Fig. 8. The proposed CSCNN approach shows promising convergence in training unbalanced traffic dataset.

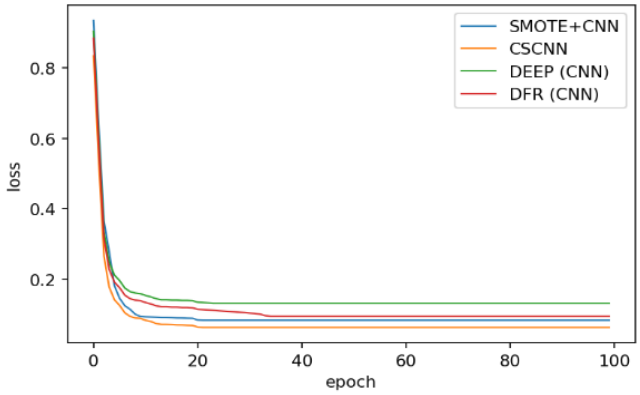

Fig. 8. Training loss of traffic classification models

V. Conclusion and Future Insights

This study proposed a CSCNN approach for traffic classification to determine the effect of the class imbalance problem on the low-frequency traffic data identification. This approach adapts misclassification costs while performing the training to minimize the cost of classifiers. Which are assigned based on the training data distribution, replacing defining a handcrafted cost matrix by expert judgment. To train deep neural networks with diverse misclassification costs, a cost matrix is generated for each epoch. In this strategy, the cost matrix is created according to the data distribution of each epoch rather than other cost-sensitive learning approaches that generate the matrix in the pre-processing phase using the entire training set. Learning with different value of costs enables deep learning models to be robust against unseen imbalanced datasets and not to be dependent on the training dataset. The CSCNN approach was adapted in the cross-entropy loss function of CNN (CSCNN). To show the superiority of the cost-sensitive traffic classification over other deep learning models, the "ISCX VPN-nonVPN" data was used for traffic description and application identification tasks. The results proved that the model can attain a high level of performance, particularly for low-frequency traffic data. CSCNN could significantly outperform other traffic classification methods.

For future work, the developed approach can be modified for complex tasks such as differentiating between two types of Skype data such as voice and chat. Applying a cost-sensitive approach with different other algorithms such as self-stacking (SAE) or Recurrent Neural Network (RNN) can also be employed for future work. Experiments have shown that applying an in-depth learning model to tens of millions of network traffic is very costly in terms of time, which would be very problematic for the internet environment, which generates a huge amount of traffic per second. Therefore, applying this model to real-time environments and bulk data is essential.

References

[1] Gil, G. D., Lashkari, A.H., Mamun, M., & Ghorbani, A. A. (2016). Characterization of encrypted and vpn traffic using time-related features. In Proceedings of 2nd International Conference on Information Systems Security and Privacy (pp. 407-414).

[2] Park, J., Tyan, H. R., & Kuo, C. C. J. (2006). Internet traffic classification for scalable QoS provision. In Proceedings of the IEEE International Conference on Multimedia and Expo (pp. 1221-1224).

[3] Khalife, J., Hajjar, A., & Diaz-Verdejo, J. (2014). A multilevel taxonomy and requirements for an optimal traffic-classification model. International Journal of Network Management, 24(2), 101-120.

[4] Wang, X., Jiang, J., Tang, Y., Liu, B., & Wang, X. (2011). StriD2FA: Scalable Regular Expression Matching for Deep Packet Inspection. In 2011 IEEE International Conference on Communications (pp. 1-5).

[5] 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, pp. 1999-2012, 2020.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[6] Tong, V., Tran, H. A., Souihi, S., & Mellouk, A. (2018). A novel QUIC traffic Classifier based on Convolutional Neural Networks. In Proceedings of the IEEE International Conference on Global Communications (pp. 1-6).

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

[8] Wang, W., Sheng, Y., Wang, J., Zeng, X., Ye, X., Huang, Y., & Zhu, M. (2018a). HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. IEEE Access, 6, 1792-1806.

[9] Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., & Lloret, J. (2017). Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. IEEE Access, 5, 18042-18050.

[10] Wang, P., Ye, F., Chen, X., & Qian, Y. (2018). Datanet: Deep learning based encrypted network traffic classification in sdn home gateway. IEEE Access 6.

[11] Höchst, J., Baumgärtner, L., Hollick, M., & Freisleben, B. (2017). Unsupervised traffic flow classification using a neural autoencoder. In Proceedings of IEEE 42nd Conference on Local Computer Networks (pp. 523-526).

[12] Hua, N., Song, H., & Lakshman, T. V. (2009). Variable-stride multi-pattern matching for scalable deep packet inspection. In Proceedings of IEEE INFOCOM (pp. 415-423).

[13] Sharif, M. S., Abbod, M., Al-Bayatti, A., Amira, A., Alfakeeh, A. S. and Sanghera B. (2020). "An Accurate Ensemble Classifier for Medical Volume Analysis: Phantom and Clinical PET Study," in IEEE Access, vol. 8, pp. 37482-37494, doi: 10.1109/ACCESS.2020.2975135.

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

[15] Ramentol, E. Caballero, Y. Bello, R. Herrera, F. (2012). SMOTE-RSB*: A hybrid preprocessing approach based on oversampling and under sampling for high imbalanced data-sets using SMOTE & rough sets theory. Knowledge and information systems, 33(2), 268-265.