سیستم های اطلاعاتی هوشمند
نشریه علمی دانشگاه آزاد اسلامی
مجلد۱، شماره ۴
صفحات ۵۰-۵۸
**مقاله پژوهشی**

Intelligent Information Systems Journal
Islamic Azad University, Hamedan Branch

# Deep learning and a cost-sensitive strategy for intrusion detection system

Maryam Pournaghdi

Department of Computer Engineering, Hamedan Branch, Islamic Azad University, Hamedan, Iran

**Abstract**
Intrusions and attacks are critical problems in network security and privacy. There are many studies on intrusion detection, most of which use traditional data mining algorithms to detect intrusion. Intrusion detection using deep learning is a new approach to cyber security. Unbalanced data is one of the major challenges in intrusion detection in which the number of samples of some classes (majority) is much higher than other classes (minority) which increases the rate of incorrect classifications for minority classes and the created model tends towards the majority class. Although some studies have tried to address this issue by using resampling techniques, they are not effective. This research uses deep learning that combines the phases of classification and automatic feature extraction. Unlike previous approaches, the proposed method addresses the class imbalance problem during model training. The developed mechanism uses a cost-sensitive learning strategy and determines a cost for each misclassification based on the class distribution. These costs are considered during training when there are incorrect classifications to fit the data. The efficiency of the designed approach is assessed using different criteria such as accuracy, recall, precision, and F1-Score by the same method. Experiments have shown that the proposed method can improve the classification performance by an average of 3%.
*Keywords:* Intrusion detection, Imbalanced data, Deep learning, Cost-sensitive learning

## 1. Introduction

Security and privacy issues are the major challenges in computer networks because many computer networks (such as the Internet of Things) are based on the conventional Internet model without security [1][2]. Network attacks occur for a variety of reasons, such as obtaining sensitive information, disrupting data integrity, and damaging services in computer networks. Intrusion Detection System (IDS) is a second line of defense mechanism that is often used in conjunction with other security mechanisms such as access control and encryption techniques to secure IoT networks against cyber-attacks. To ensure the security of computer networks, the use of IDS is very important to prevent rapid attacks and protect privacy [3]. IDS components can be contracted in a hierarchical architecture to transfer the parsed data in layers.

Deep learning (DL) is one of subfields of machine learning has attracted attention in intrusion detection. This is because hand-crafted features are not required in DL algorithms that make them powerful extracting knowledge without the help of experts [4]. In computer networks, the distribution ratio between some classes is significantly higher than other classes. This issue refers to the class imbalance problem, where some attacks are low-frequent and cannot be detected carefully because there a bias towards the high-frequency classes. Thus, the accuracy of the classifier is high for the majority classes, while the accuracy is low for the minority attacks. Resampling strategies, such as undersampling and oversampling are simple techniques, but they introduce some challenges, such as information loss, high complexity, and over-fitting [1].

To address the challenges related to the resampling strategies, cost-sensitive learning is a useful strategy that builds DL models strong against unbalanced datasets. In this strategy, each misclassification is given a cost and these costs are considered during DL model training in loss function. A higher cost is often assigned to the minority class than the majority one. Cost-sensitive learning has achieved promising results in DL algorithms, such as auto-encoder [5][6][1], Deep Belief Networks (DBN) [7], Convolution neural network (CNN) [8], deep neural network (DNN) [9], and multilayer perceptron (MLP) [10].

In this study, an intrusion detection framework for computer network security is developed based on a cost-sensitive DL approach. We integrate cost-sensitive learning into two DL classifiers, including SAE and CNN. In both models, the cross-entropy cost function is improved considering class-dependent costs. In each epoch of DL training, diverse cost matrices are generated according data distribution. In this way, strong representations can

be trained. To demonstrate the superiority of our proposed approaches over other DL-based intrusion detection methods, the UNSW-NB15 and NSL-KDD datasets are used.

Structure of this study is as follows: Section 2 summarizes related works on deep learning-based intrusion detection methods. Section 3 presents a cost-sensitive learning-aided DL mechanism for mitigating impact of imbalanced data on intrusion detection systems. Some experiments are carried in Section 4 to evaluate performance of our developed model. Finally, Section 5 presents a conclusion of the paper.

## 2. Related works

In recent years, DL has attracted a great attention in network intrusion detection. Javaid et al. [11] developed an IDS using self-taught learning based on sparse autoencoder. They evaluated their system on NSL-KDD dataset. This system includes two phases: high-level feature representations are extracted from unlabeled data using unsupervised learning at the first stage. In the second phase, the learnt features are used for classification task on the labeled data. Non-symmetric deep auto-encoder was used by Shone et al. [12] for unsupervised learning. A classifier by combining stacked auto-encoder and random forest was developed. The authors analyzed their model using the KDD Cup99 and NSL-KDD datasets. A two-stage SAE model was developed by Khan et al. [13]. A binary classification between normal or abnormal traffic is done in the first phase and the attacks type is detected in the second phase.

Vinayakumar et al. [14] developed network-based IDS and host-based IDS by modeling a MLP. They used text representation methods in NLP (e.g., Bag-of-Words (BoW), N-grams, and Keras Embedding) to capture the contextual information in the form of feature vectors. Ge et al. [15] developed a feed-forward neural network (FNN) architecture with three hidden layers and 512 neurons in each layer. They also used three regularization techniques of L1, L2, and dropout.

An IDS using recurrent neural networks (RNN) was developed in [16]. Both binary detection and multiclass detection were considered. Diverse models with different learning rate and neurons are trained. The performance of the models was evaluated on NSL-KDD dataset. Tang et al. [17] built a DNN-based intrusion detection mechanism for Software Defined Networking (SDN). The DNN model included three hidden layers with 12, 6, and 3 neurons, respectively. The model was trained on the NSL-KDD Dataset.

Some studies have combined DL models to strengthen intrusion detection systems. A combination of CNN and weight-dropped Long Short-Term Memory (LSTM) was developed, where features are extracted using a two-layer CNN followed by a weight-dropped LSTM [18]. A deep blockchain framework based on Bidirectional LSTM was designed in [19] to provide secure data exchange and migration between multi-cloud IoT services. A two-stage mechanism was developed in [20], where a single-hidden layer was employed at the first stage to distinguish normal and intrusion traffics. In the second phase, DNN and LSTM techniques were applied to identify intrusion activities. A combination of DL and support vector machines (SVM) was introduced in [21]. Sparse autoencoder was employed for feature learning and dimensionality reduction. The classification is done using SVM.

Zhang et al. [22] developed a self-adaptive DBN model based on an improved version of the genetic algorithm to find the best values of hidden layers and neurons in each layer adaptively. DL-based Intrusion Detection System (DLIDS) [23] is a combination of Spider Monkey Optimization algorithm (SMO) and the Stacked-Deep Polynomial Network (SDPN) for IoT security. In the preprocessing step, Minkowski distance and nearest neighbour computation are applied to clean the dataset. The SMO algorithm is then employed for feature selection.

## 3. Proposed methodology

This section presents a cost-based DL model including four phases: data preprocessing, cost matrix generation, DL architecture, and improved loss function (Fig. 1).

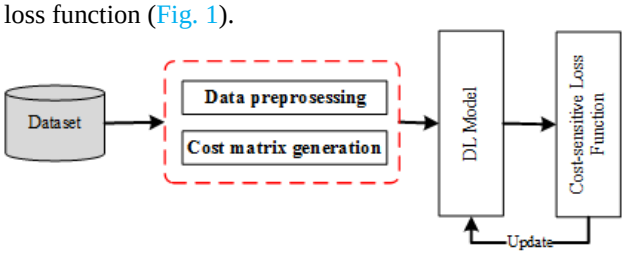



**Fig. 1.** The framework for our methodology

### *3.1. Data Preprocessing*

In the preprocessing stage, irrelevant features like the source/destination IPs and port numbers are first deleted. Nominal columns (e.g., “proto”, “state”, and “service”) are then converted to numerical values through the ordinal encoding technique. We apply Min-Max normalization to transform all values in the range [0,1] (Eq. 1). This is because the difference between values is large that affects the convergence rate and model training.

$$f_{i,j} = \frac{y_{i,j} - \min(y_i)}{\max(y_i) - \min(y_i)} \quad (1)$$

where $y_{i,j}$ is the value $j$ of the feature $i$ and $y_i$ represents the feature $i$.

### *3.2. Cost matrix generation*

Generating a cost matrix is a critical task in cost-sensitive learning to determine misclassification

cost of each category. The cost matrix is adopted when computing loss value. Unlike many previous approaches for cost matrix generation that the user/specialist manually determines the costs for each class, our presented technique uses a formulation to automatically define the costs without user intervention. These cost values are defined by calculating the data statistics of categories. Fig. 2 depicts a general framework of cost matrix generation.

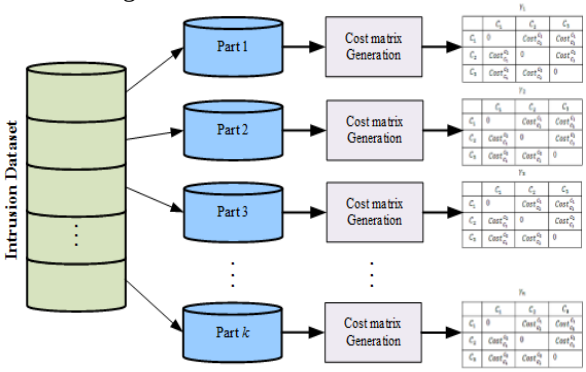



**Fig. 2** Cost matrix generation process

First, data distribution of each category is determined to be considered in generating the cost matrix based on a heuristic designed for cost definition. Low-frequency classes are given high costs and high-frequency classes are given low cost values. Each cost of classifying category $i$ as category $j$ is determined using Eq. 2.

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

The terms $\alpha_i$ and $\alpha_j$ are distribution of classes $i$ and $j$, respectively. If there is no sample for a class in a partition (i.e., $\alpha_i$ or $\alpha_j$=0), the costs related to the class is determined to be zero. Indeed, all cells in the corresponding row and column are set to be zero.

In the cost matrix, the diagonal row refers to the optimal vector. It indicates the correct classifications and the values in this vector are zero. The costs in other cells are non-negative (i.e., $\gamma_{i,j} > 0$). Table 1 presents an example of a cost matrix for a three-classification task.

Table 1: An example of a cost matrix with three 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 |

*3.3. Deep learning models architectures*

The architectures of our developed models are described in this section. Our designed CNN model consists of a 1D input layer and 3 convolution layers (Fig. 3). Each convolution layer is followed by a ReLU activator and a max-pooling layer. The size of each filter in the convolution layer is 8×1 with stride=1. Moreover, the size of the max-pooling is 4×1 with stride=2. Batch normalization and a dropout layer with the ratio of 0.05 are applied after each ReLu. At the end of the CNN model, two fully-connected layers are employed to detect attacks.

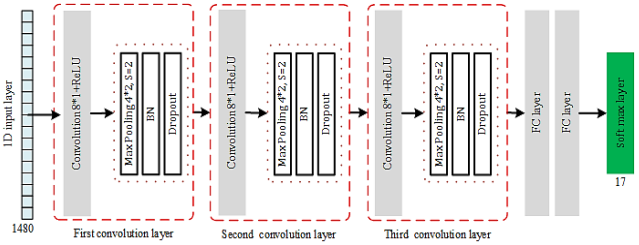



**Fig. 3** Our CNN architecture

Our SAE architecture proposed in this study has two auto-encoders, each with two encoding layers and two decoding layers (Fig. 4).

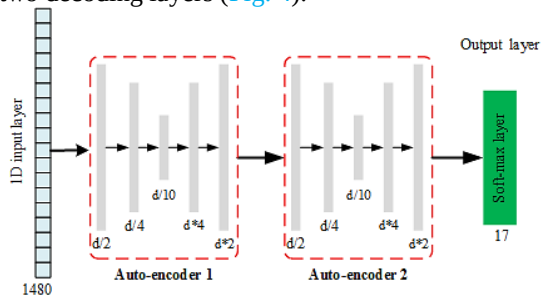



**Fig. 4.** The proposed SAE architecture

*3.4. Cost-Sensitive loss function*

A new cross-entropy loss function is proposed in this section with the aim of making DL models more sensitive to misclassification of the low-frequency attacks than the majority categories. During the training phase, class-related costs are considered to optimize DL parameters according to the class imbalance problem. Unlike data-level techniques (e.g., oversampling and undersampling), the developed models do not change the data distribution, which results in reducing computation time of the training process.

Our new loss function aims to penalize misclassifications according to corresponding costs. This penalty value is larger when a low-frequency sample is detected as a high-frequency category than when a high-frequency sample is wrongly detected as a low-frequency class. These misclassification costs are found the cost matrix. This technique intends to enhance the cross-entropy loss function by injecting the class-dependent costs. In fact, the output of the Softmax layer, which is in the form of probabilities, is considered as the input of the cost function to calculate the amount of cost-sensitive loss. The reason for choosing cross-entropy is that it

can perform better than other cost functions in most cases. Besides, cross-entropy can prevent the slowing down of learning, which is one of the problems with the mean square error (MSE) function in learning.

Before illustration of our cost-sensitive DL mechanism, how the Softmax layer works is explained. Let us 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 number of neurons in the last layer and $C$ is the number of categories). Softmax layer calculates the probability that object $i$ ($x_i$) belongs to each class, as expresses bellows:

$$f_\theta(x) = \frac{1}{\sum_{j=1}^{C} e^{\theta_j^T x_i}} \begin{bmatrix} e^{\theta_1^T x_i} \\ e^{\theta_2^T x_i} \\ \dots \\ e^{\theta_C^T x_i} \end{bmatrix} = \begin{bmatrix} p(y_i = 1|x_i;\theta_1) \\ p(y_i = 2|x_i;\theta_2) \\ \dots \\ p(y_i = C|x_i;\theta_C) \end{bmatrix} \quad (3)$$

where is the parameter mapping toward the $j$–the class ($s.t.\ b_j + W_j x$).

In our improved cross-entropy, misclassification errors are punished according the costs in cost matrix. The goal is to approximate the prediction value to the actual output:

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

Where $y_{o,c}$ is a binary value indicating the accurate prediction for instance $o$, where is $y_{o,c}$ is "1" when the prediction is correct, otherwise, the values is "0" for the actual category. The probability of the predicted class is changed by incorporating the related class-dependent cost:

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

## 4. Experimental setup

This section compares the performance of our developed cost-sensitive SAE (termed ICSAE) and CNN (termed ICSCNN) with other models. These models include non-sensitive versions for SAE and CNN, combining models with the SMOTE method, which is a preprocessing sampling technique, and the NADE method [12]. The Keras and Tensorflow libraries were used as backends to run DL models. The number of 100 epochs was considered for model training. An early stopping technique was applied to mitigate the overfitting problem, where the training phase stops when the amount of error on the validation set remains unchanged for certain iterations. The Adam is used as an optimizer function for DL models. The ratios of the training set, validation set, and the testing sets are set to be 0.8, 0.1, and 0.1, respectively.

### *4.1. Datasets description*

In this study, two datasets of UNSW-NB15 [24] and NSL-KDD are used to train and evaluate DL models for intrusion detection. The UNSW-NB15 dataset has been collected at the period of January and February 2015. Both attack and normal traffic against servers were created using an automatic attack generation tool (IXIA PerfectStorm). This dataset has nine types of attacks, including "Fuzzers", "Analysis", "Backdoors", "DoS", "Exploits", "Generic", "Reconnaissance", "Shellcode", and "Worms". NSL-KDD dataset consists of five classes of "DoS", "Probe", "U2R", "R2L", and "Normal".

Table 2 shows the statistics of these two datasets. The UNSW-NB15 dataset consists of 2.2 million samples with nine categories of attacks, of which eight are minority classes and the majority of instances belong to the normal class (around 88%). According to the statistics, Dos and Normal classes make the majority size of the dataset, 0.35 and 0.53, respectively. On the other hand, "Probe", "R2L", and "U2R" attacks have rarely happened.

**Table 2** Statistics of UNSW-NB15 and NSL-KDD datasets

| Dataset | Attack type | The number of samples | Ratio |
|---|---|---|---|
| **UNSW-NB15** | Normal | 1,958,467 | 0.879 |
| | Exploits | 33,422 | 0.015 |
| | DoS | 11,716 | 0.005 |
| | Backdoor | 1,959 | 0.0008 |
| | Analysis | 2,069 | 0.0009 |
| | Fuzzers | 19,578 | 0.00879 |
| | Generic | 187,598 | 0.084 |
| | Reconnaissance | 10,871 | 0.004 |
| | Shellcode | 1,187 | 0.0005 |
| | Worms | 134 | 0.00006 |
| | **Total** | **2,227,000** | |
| **NSL-KDD** | DoS | 53,383 | 0.356 |
| | Probe | 14,077 | 0.088 |
| | U2R | 3,751 | 0.022 |
| | R2L | 252 | 0.0000614 |
| | Normal | 77,053 | 0.532 |
| | **Total** | **148,516** | |

### *4.2. Evaluation criteria*

In our evaluation, five measures of accuracy (Eq. 4), recall (Eq. 5), precision (Eq. 6), and F1-score (Eq. 7) are employed for evaluation of intrusion classification approaches.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

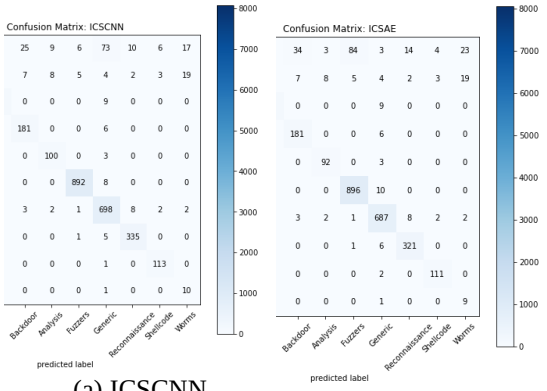
$$Precision = \frac{TP}{TP+FP} \quad (6)$$

$$F1\text{-}Score = \frac{2.\text{Recall.Precision}}{\text{Recall+Precision}} \quad (7)$$

In these equations, *TP*, *FP*, *TN*, and *FN* are the number of *True Positives, False Positives, True Negatives, and False Negatives*, respectively.

*4.3. Results on UNSW-NB15 dataset*

Fig. 5 shows the confusion matrix for the proposed cost-sensitive models, ICSCNN (5a) and ACSAE (5b).




(a) ICSCNN (b) ICSAE

Fig. 5 Confusion matrix for the UNSW-NB15 dataset

Table 3 shows recall evaluation of our developed models compared with the other classifiers. The recall is a major criterion for evaluating the effectiveness of classifiers because the number of detecting low-frequency samples as the majority class is great (i.e., the number of false negatives for the low-frequency class is high). Thus, the recall value for the minority classes is less than for the majority classes. Fig. 6 presents the average recall ratios for the low-frequency categories. The results demonstrate integrating cost-sensitive strategy into DL models could improve recall values for minority categories (such as "DoS", "Backdoor", "Shellcode", and "Exploits"). This means that our models were able to correctly identify minority classes. ICSCNN and ICSAE performed better than the other methods. In general, the proposed method was able to obtain an average recall ratio of 87% for low-frequency classes.

Table 3: Comparison of recall ratio for intrusion detection models on UNSW-NB15 dataset

| | **ICSAE** | **ICSCNN** | **SAE** | **CNN** | **SMOTE+SAE** | **SMOTE+CNN** | **NADE** |
|---|---|---|---|---|---|---|---|
| Normal | 0.998 | 0.999 | 0.99 | 0.992 | 0.996 | 0.997 | 0.997 |
| **Exploits** | 0.828 | 0.85 | 0.728 | 0.754 | 0.8 | 0.815 | 0.792 |
| **DoS** | 0.981 | 0.986 | 0.971 | 0.972 | 0.979 | 0.98 | 0.97 |
| **Backdoor** | 0.77 | 0.83 | 0.54 | 0.62 | 0.756 | 0.78 | 0.72 |
| **Analysis** | 0.891 | 0.916 | 0.773 | 0.842 | 0.884 | 0.89 | 0.858 |
| **Fuzzers** | 0.996 | 0.997 | 0.99 | 0.991 | 0.994 | 0.995 | 0.996 |
| Generic | 0.987 | 0.989 | 0.956 | 0.974 | 0.987 | 0.988 | 0.985 |
| **Reconnaissance** | 0.969 | 0.977 | 0.943 | 0.954 | 0.965 | 0.969 | 0.963 |
| **Shellcode** | 0.832 | 0.857 | 0.671 | 0.8 | 0.812 | 0.829 | 0.728 |
| **Worms** | 0.986 | 0.988 | 0.951 | 0.975 | 0.986 | 0.987 | 0.983 |
| **Average** | **0.923** | **0.938** | **0.851** | **0.887** | **0.915** | **0.923** | **0.899** |

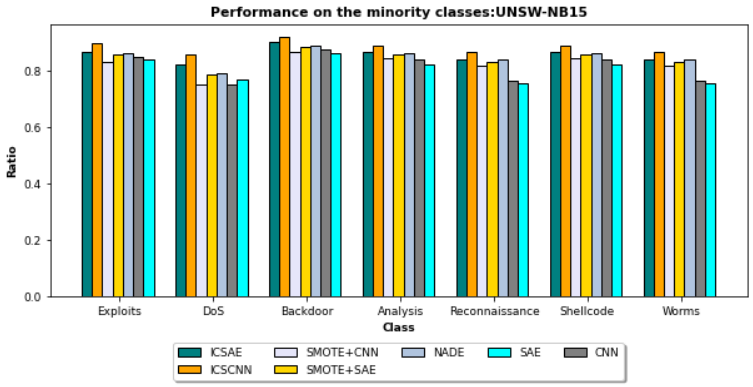



**Fig. 6** Performance of DL models for the minority classes on the UNSW-NB15 dataset

Since the number of false positive for high-frequency classes increases when training DL models on the imbalanced data, precision ratio of the low-frequency categories is larger than that of high-frequency classes. Table 4 confirms this issue, where the precision values are higher for the low-frequency attacks than their recall ratios. From the table, the precision ratio of ICSCNN with 99.4% is higher than other DL models.

**Table 4** Precision comparison of DL models for intrusion detection on UNSW-NB15 dataset

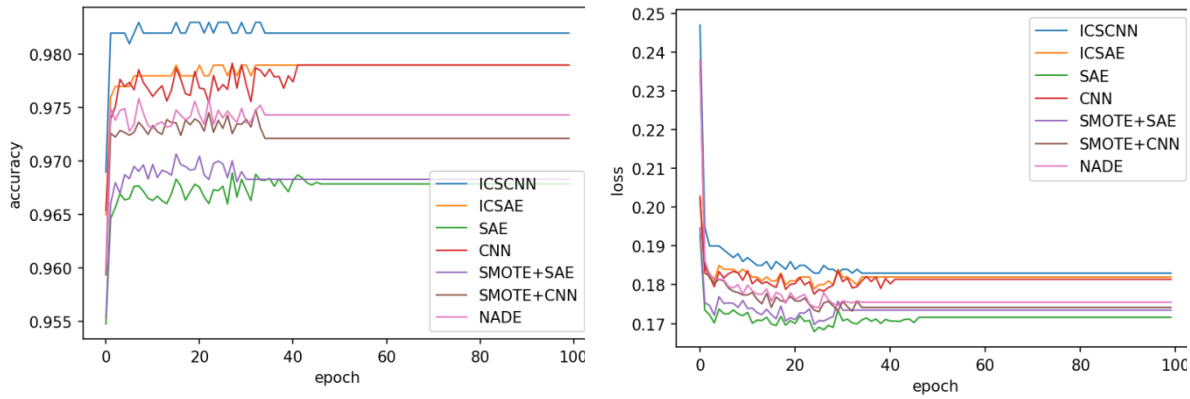
| | ICSAE | ICSCNN | SAE | CNN | SMOTE+SAE | SMOTE+CNN | NADE |
|---|---|---|---|---|---|---|---|
| Normal | 0.997 | 0.997 | 0.982 | 0.98 | 0.995 | 0.997 | 0.994 |
| **Exploits** | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **DoS** | 0.997 | 0.997 | 0.982 | 0.98 | 0.995 | 0.997 | 0.994 |
| **Backdoor** | 0.99 | 0.992 | 0.967 | 0.976 | 0.983 | 0.986 | 0.988 |
| **Analysis** | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Fuzzers** | 0.989 | 0.992 | 0.98 | 0.984 | 0.99 | 0.99 | 0.985 |
| Generic | 0.988 | 0.99 | 0.978 | 0.98 | 0.987 | 0.99 | 0.983 |
| **Reconnaissance** | 0.99 | 0.992 | 0.99 | 0.987 | 0.99 | 0.991 | 0.978 |
| **Shellcode** | 1.0 | 1.0 | 0.99 | 0.99 | 1.0 | 1.0 | 1.0 |
| **Worms** | 0.99 | 0.99 | 0.987 | 0.984 | 0.986 | 0.99 | 0.99 |
| **Average** | **0.993** | **0.994** | **0.985** | **0.987** | **0.991** | **0.993** | **0.991** |

The F1-Score is a harmonic mean of recall and precision criteria. Table 5 shows compares F1-Score ratios of intrusion detection models. According to the results, ICSCNN achieved the highest performance with 96.7%. Overall, the proposed models can achieve an average of 96.4% for the F1-Score. This shows that our cost-sensitive DL method can optimize classifiers by considering the misclassification costs when computing loss value. This enables classifiers to learn features with a sensitive trend towards low-frequency attacks.

**Table 5** F1-Score comparison of DL models for intrusion detection on UNSW-NB15 dataset

| | ICSAE | ICSCNN | SAE | CNN | SMOTE+SAE | SMOTE+CNN | NADE |
|---|---|---|---|---|---|---|---|
| Normal | 0.87 | 0.907 | 0.701 | 0.837 | 0.861 | 0.876 | 0.765 |
| **Exploits** | 0.906 | 0.919 | 0.843 | 0.860 | 0.889 | 0.898 | 0.884 |
| **DoS** | 0.989 | 0.991 | 0.976 | 0.976 | 0.987 | 0.988 | 0.982 |
| **Backdoor** | 0.994 | 0.995 | 0.978 | 0.984 | 0.989 | 0.991 | 0.992 |
| **Analysis** | 0.942 | 0.956 | 0.872 | 0.914 | 0.938 | 0.942 | 0.924 |
| **Fuzzers** | 0.992 | 0.994 | 0.985 | 0.987 | 0.992 | 0.992 | 0.990 |
| Generic | 0.908 | 0.923 | 0.800 | 0.885 | 0.896 | 0.907 | 0.843 |
| **Reconnaissance** | 0.979 | 0.984 | 0.966 | 0.970 | 0.977 | 0.980 | 0.970 |
| **Shellcode** | 0.987 | 0.989 | 0.967 | 0.977 | 0.987 | 0.989 | 0.984 |
| **Worms** | 0.988 | 0.989 | 0.969 | 0.979 | 0.986 | 0.988 | 0.986 |
| **Average** | **0.961** | **0.967** | **0.917** | **0.987** | **0.954** | **0.959** | **0.940** |

Fig. 7 shows the effect of the number of epochs on the training accuracy of intrusion detection methods. We can see that our cost-sensitive methods in the 20th period reached maximum accuracy, with 98% accuracy. Conversely, the training accuracy of the other methods reached a maximum accuracy in more epochs 97%.



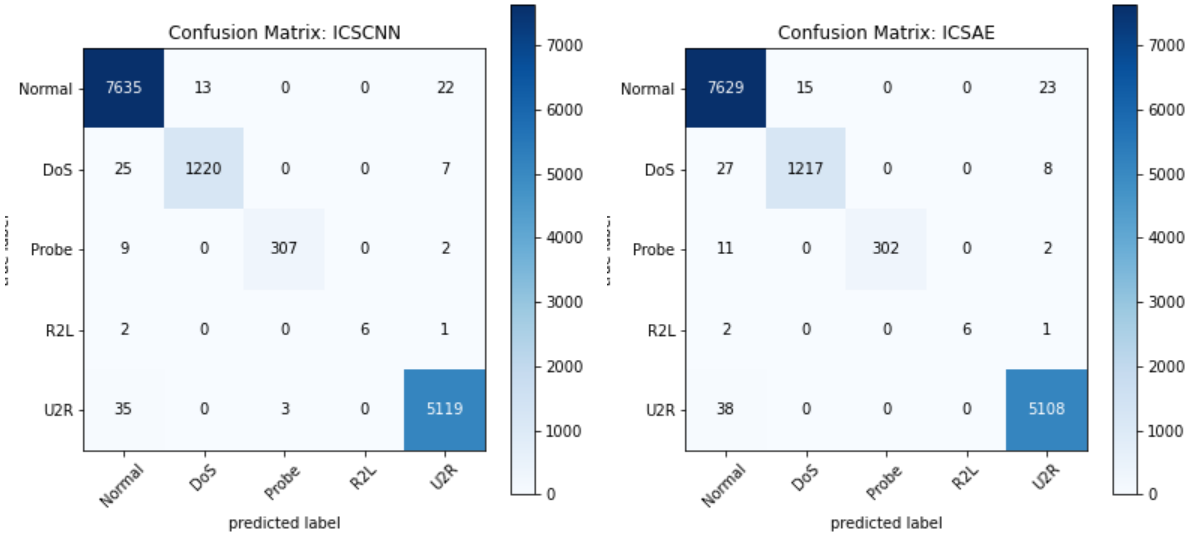

(a) Training accuracy

(b) Training loss

**Fig. 7** Training accuracy and loss value of intrusion detection models for UNSW-NB15 dataset

*4.4. Results on NSL-KDD dataset*

This section investigates the performance of the developed models in comparison with similar methods for the NSL-KDD dataset. Figs. 8a and 8b show the confusion matrices of the proposed methods for the NSL-KDD dataset.




(a) ICSCNN (b) ICSAE

**Fig. 8** Confusion matrix for the NSL-KDD dataset

Table 6 shows the recall of DL-based intrusion detection models for the NSL-KDD dataset. The experimental results showed that the performance of the models is low for the minority classes ("Probe", "R2L", and "U2R"). The models had the lowest recall rates for the U2R class because the number of events for this class is very low, 71.6% on average. The ICSCNN model with 83% and the SAE model with 54% have the highest and lowest performance for the U2R class, respectively. Overall, the ICSCNN model was able to provide the highest call rate with 0.9.

Table 6: Comparison of recall ratio for intrusion detection models on the NSL-KDD dataset

| | ICSAE | ICSCNN | SAE | CNN | SMOTE+SAE | SMOTE+CNN | NADE |
|---|---|---|---|---|---|---|---|
| DoS | 0.998 | 0.999 | 0.99 | 0.992 | 0.996 | 0.997 | 0.997 |
| **Probe** | 0.828 | 0.85 | 0.728 | 0.754 | 0.8 | 0.815 | 0.792 |
| **R2L** | 0.832 | 0.857 | 0.671 | 0.8 | 0.812 | 0.829 | 0.728 |
| **U2R** | 0.77 | 0.83 | 0.54 | 0.62 | 0.756 | 0.78 | 0.72 |
| Normal | 0.996 | 0.997 | 0.99 | 0.991 | 0.994 | 0.995 | 0.996 |
| **Average** | **0.8848** | **0.9066** | **0.7838** | **0.8314** | **0.8716** | **0.8832** | **0.8466** |

Fig. 9 shows the performance of intrusion detection models for the minority classes. The proposed methods had the highest recall ratio, which is about 80%.

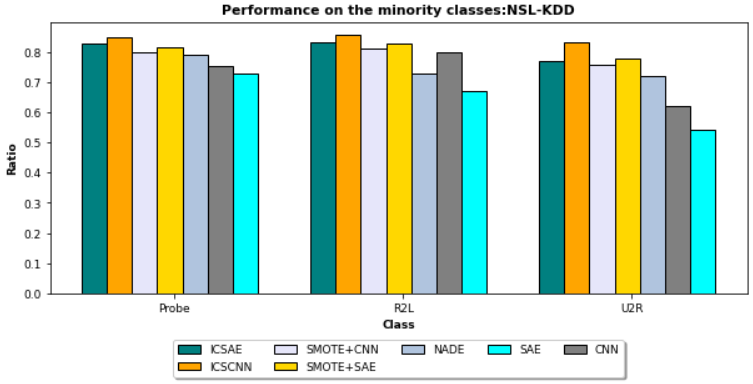



**Fig. 9** Performance of DL models for the minority classes on the NSL-KDD dataset

Table 7 compares the performance of our proposed cost-sensitive models with the other models in terms of precision criteria is presented in Table 7. Minority classes have a higher degree of precision than the

majority classes because the number of samples that are mistakenly classified as the majority classes is high.

Table 7: Comparison of precision ratio for intrusion detection models on the NSL-KDD dataset

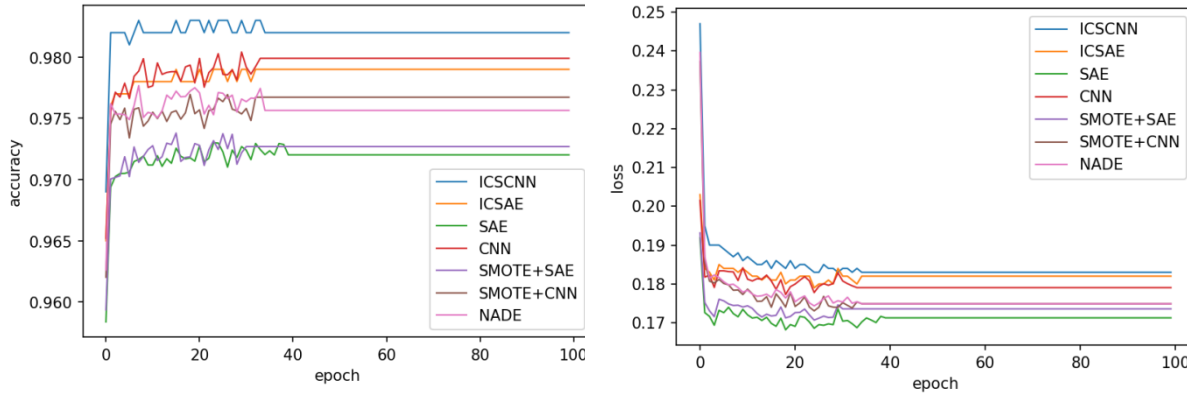
| | ICSAE | ICSCNN | SAE | CNN | SMOTE+SAE | SMOTE+CNN | NADE |
|---|---|---|---|---|---|---|---|
| DoS | 0.997 | 0.997 | 0.982 | 0.98 | 0.995 | 0.997 | 0.994 |
| **Probe** | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **R2L** | 0.999 | 0.998 | 0.982 | 0.98 | 0.995 | 0.997 | 0.994 |
| **U2R** | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| Normal | 0.99 | 0.992 | 0.967 | 0.976 | 0.983 | 0.986 | 0.988 |
| **Average** | 0.9972 | 0.9974 | 0.9862 | 0.9872 | 0.9946 | 0.996 | 0.9952 |

Table 8 shows the F1-Score values obtained from intrusion detection models. The ICSCNN model had the highest value (94.7%) while the SAE model had the lowest performance with 86%.

Table 8: Comparison of F1-Score ratio for intrusion detection models on the NSL-KDD dataset

| | ICSAE | ICSCNN | SAE | CNN | SMOTE+SAE | SMOTE+CNN | NADE |
|---|---|---|---|---|---|---|---|
| DoS | 0.9975 | 0.997 | 0.985 | 0.985 | 0.995 | 0.997 | 0.995 |
| **Probe** | 0.905 | 0.918 | 0.842 | 0.859 | 0.888 | 0.898 | 0.883 |
| **R2L** | 0.907 | 0.922 | 0.797 | 0.88 | 0.894 | 0.905 | 0.84 |
| **U2R** | 0.87 | 0.907 | 0.701 | 0.765 | 0.861 | 0.876 | 0.837 |
| Normal | 0.992 | 0.994 | 0.978 | 0.983 | 0.988 | 0.99 | 0.991 |
| **Average** | **0.934** | **0.947** | **0.86** | **0.894** | **0.925** | **0.933** | **0.909** |

Fig. 10 shows the accuracy ratio and error ratio of DL models for detecting intrusion in the training phase. As can be seen, the ICSCNN algorithm has achieved the highest training accuracy and all models have reached maximum performance in epochs between 35 and 40.




(a) Training accuracy

(b) Training loss

**Fig. 10** Training accuracy and loss of intrusion detection models for the NSL-KDD dataset

## 5. Conclusions

This paper addressed the class imbalance problem in intrusion detection systems. A modified loss function was proposed using the cost-sensitive strategy, which considers the costs assigned to different classes in the classification error calculation step. In the proposed method, the categorical cross-entropy loss function was improved for the CNN and SAE models. To build DL models resistant against unbalanced data, a varied cost matrix generation approach is proposed. In this approach, the dataset is divided into several parts and a cost matrix is generated based on the distribution of samples for each part. The UNSW-NB15 and NSL-KDD datasets were used to assess the developed approach. Accuracy, recall, precision, and F1-Score criteria were used to compare the proposed method with similar models. The two cost-sensitive algorithms of CNN and SAE were compared with their insensitive versions as well as in combination with SMOTE technique and NADE method. The results showed that the proposed method has higher detection ability for the minority classes. On average, the proposed model has increased the intrusion detection performance by about 3%.

Computer networks, especially IoT environments, generate a huge amount of data at tremendous speed. Besides, the traffic capacity of IoT networks has been greatly increased in order to comfort the transfer of data in the networks. Therefore, intrusion detection systems need to be performed rapidly.

Developing a system that can scan data at high speed is a basic need in intrusion detection. In fact, both the volume of data and the speed of data production must be managed. Reducing IDS computation time for Big data using parallel/distributed processing can be considered for future work. Large data processing technologies such as CPU/GPU, Hadoop, and Spark can be used to increase the performance of the proposed model.

**References**


[1] Yao, H., Gao, P., Wang, J., Zhang, P., Jiang, C., & Han, Z. (2019). Capsule network assisted IoT traffic classification mechanism for smart cities. *IEEE Internet of Things Journal*, *6*(5), 7515-7525.

[2] Rezaei, S., & Liu, X. (2019). Deep learning for encrypted traffic classification: An overview. *IEEE communications magazine*, *57*(5), 76-81.

[3] P. Wang, X. Chen, F. Ye, and Z. Sun, "A survey of techniques for mobile service encrypted traffic classification using deep learning," *IEEE Access*, vol. 7, pp. 54024-54033, 2019.

[4] J. Höchst, L. Baumgärtner, M. Hollick, and B. Freisleben, "Unsupervised traffic flow classification using a neural autoencoder," in *2017 IEEE 42nd Conference on Local Computer Networks (LCN)*, 2017, pp. 523-526.

[5] Y. Fan, C. Zhang, Z. Liu, Z. Qiu, Y. He, Cost-sensitive stacked sparse auto-encoder models to detect striped stem borer infestation on rice based on hyperspectral imaging, Knowledge-Based Systems 168 (2019) 49-58.

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

[7] Zhang, C., Tan, K. C., Li, H., & Hong, G. S. (2018). A cost-sensitive deep belief network for imbalanced classification. *IEEE transactions on neural networks and learning systems*, *30*(1), 109-122.

[8] S.H. Khan, M. Hayat, M. Bennamoun, F.A. Sohel, R. Togneri, Cost-sensitive learning of deep feature representations from imbalanced data, IEEE transactions on neural networks and learning systems 29(8) (2017) 3573–3587.

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

[10] Wang, H., Cui, Z., Chen, Y., Avidan, M., Abdallah, A. B., & Kronzer, A. (2018). Predicting hospital readmission via cost-sensitive deep learning. *IEEE/ACM transactions on computational biology and bioinformatics*, *15*(6), 1968-1978.

[11] Javaid, A., Niyaz, Q., Sun, W. and Alam, M., 2016. A deep learning approach for network intrusion detection system. *Eai Endorsed Transactions on Security and Safety*, *3*(9), p.e2.

[12] Shone, N., Ngoc, T.N., Phai, V.D. and Shi, Q., 2018. A deep learning approach to network intrusion detection. *IEEE transactions on emerging topics in computational intelligence*, *2*(1), pp.41-50.

[13] Khan, F.A., Gumaei, A., Derhab, A. and Hussain, A., 2019. A novel two-stage deep learning model for efficient network intrusion detection. *IEEE Access*, *7*, pp.30373-30385.

[14] Vinayakumar, R., Alazab, M., Soman, K.P., Poornachandran, P., Al-Nemrat, A. and Venkatraman, S., 2019. Deep learning approach for intelligent intrusion detection system. *IEEE Access*, *7*, pp.41525-41550.

[15] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, "Towards a deep learning-driven intrusion detection approach for Internet of Things," *Computer Networks*, vol. 186, p. 107784, 2021.

[16] Yin, C., Zhu, Y., Fei, J. and He, X., 2017. A deep learning approach for intrusion detection using recurrent neural networks. *Ieee Access*, *5*, pp.21954-21961.

[17] Tang, T.A., Mhamdi, L., McLernon, D., Zaidi, S.A.R. and Ghogho, M., 2016, October. Deep learning approach for network intrusion detection in software defined networking. In *2016 international conference on wireless networks and mobile communications (WINCOM)* (pp. 258-263). IEEE.

[18] Hassan, M. M., Gumaei, A., Alsanad, A., Alrubaian, M., & Fortino, G. (2020). A hybrid deep learning model for efficient intrusion detection in big data environment. *Information Sciences*, *513*, 386-396.

[19] Alkadi, O., Moustafa, N., Turnbull, B., & Choo, K. K. R. (2020). A deep blockchain framework-enabled collaborative intrusion detection for protecting iot and cloud networks. *IEEE Internet of Things Journal*.

[20] R. Qaddoura, M. Al-Zoubi, H. Faris, and I. Almomani, "A Multi-Layer Classification Approach for Intrusion Detection in IoT Networks Based on Deep Learning," *Sensors*, vol. 21, no. 9, p. 2987, 2021.

[21] Al-Qatf, M., Lasheng, Y., Al-Habib, M. and Al-Sabahi, K., 2018. Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. *IEEE Access*, *6*, pp.52843-52856.

[22] Zhang, Y., Li, P., & Wang, X. (2019). Intrusion detection for IoT based on improved genetic algorithm and deep belief network. *IEEE Access*, *7*, 31711-31722.

[23] Otoum, Y., Liu, D., & Nayak, A. (2019). DL-IDS: a deep learning–based intrusion detection framework for securing IoT. *Transactions on Emerging Telecommunications Technologies*, e3803.

[24] Moustafa, Nour, and Jill Slay. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)." *Military Communications and Information Systems Conference (MilCIS)*, 2015. IEEE, 2015.