JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

An intrusion detection method based on transformer - LSTM model

Mr. RAMBABU ATMAKURI - Head, Department of CSE, Anurag College of Engineering (Aushapur, Ghatkesar, Telangana 501301)

Ms. LALAPUDI TARUNI- Student, Department of CSE, Anurag College of Engineering (Aushapur, Ghatkesar, Telangana 501301)

Mr. KUSUMA VINAY KUMAR- Student, Department of CSE, Anurag College of Engineering (Aushapur, Ghatkesar, Telangana 501301)

Mr. KONDA BHARATH KUMAR- Student, Department of CSE, Anurag College of Engineering (Aushapur, Ghatkesar, Telangana 501301)

Abstract

This paper proposes an enhanced Transformer-based intrusion detection model to tackle the challenges of lengthy training time, inaccurate detection of overlapping classes, and poor performance in multi-class classification of current intrusion detection models. Specifically, the proposed model includes the following: (i) A data processing strategy that initially reduces the data dimension using a stacked auto-encoder to speed up training. In addition, a novel under-sampling method based on the KNN principle is introduced, along with the Borderline-SMOTE over-sampling method, for hybrid data sampling that balances the dataset while addressing the issue of low detection accuracy in overlapping data classes. (ii) An improved position encoding method for the Transformer model that effectively learns the dependencies between features by embedding the position information of features, resulting in better classification accuracy. (iii) A two-stage learning strategy in which the model first performs rough binary prediction (determining whether it is an illegal intrusion) and then inputs the prediction value and original features together for further multi-class prediction (predicting the intrusion category), addressing the issue of low accuracy in multi-class classification. Experimental results on the official NSL-KDD test set demonstrate that the proposed model achieves an accuracy of 88.7% and an F1-score of 88.2% in binary classification and an accuracy of 84.1% and an F1-score of 83.8% in multi-class classification. Compared to existing intrusion detection models, our model exhibits higher accuracy and F1-score and trains faster than other models.

1. INTRODUCTION

The rapid growth of the internet has brought significant convenience, but it has also led to an increasing number of network security problems. In today's world, security is of paramount concern as intruders have become more sophisticated with the advancement of technology [1]. Hackers employ various techniques to bypass firewalls, enabling them to infiltrate network systems and cause damage to the internal infrastructure or collect individuals' private information. Given the rising threats posed by intruders, network intrusion detection has emerged as a critical research direction in network security.

Intrusion detection systems can be divided into two categories: network-based intrusion detection systems (NIDSs) and host-based intrusion detection systems (HIDSs), depending on the type of intrusion behavior being monitored [2]. NIDSs monitor local network traffic by examining data packets to detect intrusion behavior, while HIDSs analyze multiple sources of information collected on the local host,

such as system data, log files, and disk resources. Traditional intrusion detection techniques include methods such as entropy-based approaches and redundancy optimization. Entropy-based approaches are used to detect anomalies, such as DDoS attacks in IEEE802.16-based networks, by calculating the entropy of network traffic [3]. This method analyzes statistical and entropy-based features of incoming traffic to determine whether an attack has occurred. However, this approach has some limitations, such as being less effective when dealing with encrypted traffic or low traffic volume. Redundancy optimization is another commonly used technique in intrusion detection, which improves the accuracy and reliability of detection by performing the same detection algorithm multiple times. The most widely used technique is Triple Modular Redundancy (TMR) [4], which processes input data through three detection modules and compares the output. If there is inconsistency, an attack can be detected. However, the main drawback of redundancy optimization is its high computational cost, requiring a significant amount of hardware resources and time.

In recent years, machine learning techniques have gained popularity in the field of intrusion detection. By analyzing large amounts of data, these techniques can discover intrusion features and patterns, leading to improved detection accuracy and speed.

Deep learning has gained widespread use in the field of intrusion detection. Compared to traditional machine learning algorithms, deep learning algorithms are better suited to handle high-dimensional data and complex features. Intrusion detection models based on deep learning commonly use convolutional neural networks (CNNs), recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and generative adversarial networks (GANs) as underlying models. In reference [12], a CNN-based intrusion detection model was built, and extensive experiments were conducted to optimize its structure, resulting in improved performance. In reference [13], a combination of a CNN and RNN was used to improve the performance of intrusion detection systems. The CNN was used for convolution to capture local features, and the RNN was used to capture temporal features. This model achieved good accuracy in both binary and multi-class classification tasks. In reference [14], a BiLSTM-based intrusion detection model was proposed, which outperformed traditional LSTM and other state-of-the-art models in terms of accuracy, precision, recall, and F1-score. In reference [15], a technique for oversampling based on GANs and feature selection was proposed. This technique first modeled a complex high-dimensional attack distribution using Gradient Penalty Wasserstein GAN to generate additional attack samples. Then, a subset of features representing the entire dataset was selected using variance analysis. Finally, a rebalanced low-dimensional dataset was generated for machine learning training. In summary, deep learning algorithms have made significant breakthroughs in the field of intrusion detection and have become an important technology. These algorithms can automatically learn features from input data and are used to detect potential network intrusion behaviors.

2. LITERATURE SURVEY

Despite significant progress in intrusion detection using machine learning and deep learning, three key challenges remain: long model training times, imbalanced datasets, and poor performance in multi-class classification. While many researchers have proposed improvements to address these challenges, the impact of these improvements has been relatively modest. In dealing with class imbalances, intrusion detection datasets often exhibit significant class imbalance, which may cause algorithms to favor predicting the more numerous class and thus lower detection accuracy. To address this issue, researchers often perform sampling on the dataset to balance it. Jiang et al. [16] suggested a detection framework that combines deep hierarchical networks with hybrid sampling techniques. In particular, they employed the one-sided selection algorithm and SMOTE technique to perform under-sampling and over-sampling, respectively, in order to balance the dataset. Zhang et al. [17] proposed another technique for processing unbalanced datasets, which combines SMOTE over-sampling with clustering-based under-sampling using a Gaussian mixture model. Their intrusion detection framework effectively addresses the class imbalance problem and improves detection accuracy. Yan et al. [18] employed an enhanced local adaptive synthetic minority over-sampling

technique to address dataset imbalance and utilized an RNN for detecting various types of traffic anomalies, leading to improved accuracy in the detection process. Zhou et al. [19] achieved good accuracy performance by combining an auto-encoder and a residual network. They reconstructed the network using an auto-encoder to perform feature extraction, and then used the extracted features to train a designed residual network. Similarly, Liu et al. [20] employed Principal Component Analysis (PCA) to reduce dimensionality, extracting a subset of principal component features that contain maximum information. The processed data was then fed to the recurrent neural networks for classification, resulting in a high accuracy rate. However, these dimensionality reduction methods do not take into account the loss of information caused by dimensionality reduction, which in turn results in a decrease in the model's classification ability. Moreover, slow model training speed is not necessarily only due to the increase in the number and dimensions of the dataset, as deep learning models with a deeper hierarchy and a larger number of trainable parameters can also lead to slow model training. In dealing with the low multi-classification ability of intrusion detection models, researchers usually improve the model's multi-class detection ability through optimization. Guo et al. [22] proposed a method for detecting attacks without any prior knowledge by combining Sub-Space Clustering (SSC) and One-Class Support Vector Machine (OCSVM Consequently, researchers have begun to apply Transformers to the field of intrusion detection. In [26], a Robust Transformation-based Intrusion Detection System (RTIDS) was proposed, which used position embedding to associate sequence information between features and stacked the encoder and decoder variants of the Transformer to learn low-dimensional feature representations from high-dimensional raw data. The self-attention mechanism was applied to facilitate the classification of network traffic types. However, the encoding of the input features into low-dimensional representations by the encoder becomes ineffective when the low-dimensional features are fed into the decoder, as the data is transformed back into high dimensions.

3. OVERVIEW OF THE SYSTEM

3.1 Existing System

The existing system utilizes deep learning techniques for intrusion detection, leveraging the theory proposed by Lecun et al. Deep learning has demonstrated remarkable performance in various domains such as Computer Vision (CV) and Natural Language Processing (NLP). In intrusion detection, deep learning methods aim to extract meaningful features from high-dimensional data through model training, treating network traffic anomaly detection as a classification problem. However, despite the effectiveness of deep learning, the issue of class imbalance in multi-classification of network traffic remains a challenge.

3.1.1 Disadvantages of Existing System

Deep learning-based intrusion detection systems have shown excellent performance in computer vision and natural language processing.

These systems convert network traffic anomaly detection problems into classification problems and enhance real-time performance through adaptive learning.

However, the multi-classification of network traffic still faces challenges due to the imbalance of classification.

3.2 Proposed System

We use the classic NSL-KDD and the up-to-date CSECIC-IDS2018 as benchmark datasets and conduct detailed analysis and data cleaning. This work proposes a machine learning algorithm, reducing the majority samples and augmenting the minority samples in the difficult set, tackling the class imbalance problem in intrusion detection so that the classifier learns the differences better in training. The classification model uses Random Forest (RF), Support Vector Machine (SVM), XGBoost, NLP with other methods, we divide the experiment into 30 methods.

3.2.1 Advantages of Proposed System

The proposed system addresses the class imbalance problem in intrusion detection by employing a machine learning algorithm.

It utilizes the NSL-KDD and CSECIC-IDS2018 datasets for benchmarking and conducts detailed analysis and data cleaning.

The proposed algorithm focuses on reducing majority samples and augmenting minority samples in the difficult set to improve classifier learning during training.

Various classification models such as Random Forest (RF), Support Vector Machine (SVM), XGBoost, and

NLP are employed, and the experiment is divided into 30 methods.
By mitigating class imbalance and employing diverse classification techniques, the proposed system aims to enhance the accuracy and effectiveness of intrusion detection in multi-class scenarios.

3.3 Proposed System Design

In this project work, there are two modules and each module has specific functions, they are:

1. Admin Module

2. User Module

3.3.1 ADMIN Module

Data Collection:

There are three symbolic data types in NSL-KDD data features: protocol type, flag and service. We use one-hot encoder mapping these features into binary vectors. One-Hot Processing: NSL-KDD dataset is processed by one-hot method to transform symbolic features into numerical features. For example, the second feature of the NSL-KDD data sample is protocol type. The protocol type has three values: tcp, udp, and icmp. One-hot method is processed into a binary code that can be recognized by a computer, where tcp is [1, 0, 0], udp is [0, 1, 0], and icmp is [0, 0, 1]

Pre-processing:

When the dataset is extracted, part of the data contains some noisy data, duplicate values, missing values, infinity values, etc. due to extraction errors or input errors. Therefore, we first perform data preprocessing. The main work is as follows.

1. Duplicate values: delete the sample's duplicate value, only keep one valid data.

2. Outliers: in the sample data, the sample size of missing values(Not a Number, NaN) and Infinite values(Inf) is small, so we delete this.

3. Features delete and transform: In CSE-CIC-IDS2018, we delete features such as ''Timestamp'', ''Destination Address'', ''Source Address'', ''Source Port'', etc. If features ''Init Bwd Win Byts'' and features ''Init Fwd Win Byts'' have a value of −1, we add two check dimensions. The mark of −1 is 1. Otherwise, it is 0. In NSL-KDD, we use the One Hot encoder to complete this conversion. For example, ''TCP'', ''UDP'' and ''ICMP'' are functions of three protocol types. After OneHot encoding, they become binary vectors (1, 0, 0), (0, 1, 0), (0, 0, 1). The protocol type function can be divided into three categories, including 11 categories for flag function and 70 categories for service function. Therefore, the 41 dimensions initial feature vector becomes 122 dimensions.

4. Numerical standardization: In order to eliminate the dimensional influence between indicators and accelerate the gradient descent and model convergence, the data is standardized, that is, the method of obtaining Z-Score, so that the average value of each feature becomes 0 and the standard deviation becomes 1, converted to a standard normal distribution, which is related to the overall sample distribution, and each sample point can have an impact on standardization. The standardization formula is as follows, u is the mean of each feature, s is the standard deviation of each feature, and x 0 i is the element corresponding to each column's features.

Train-Test Split and Model FITTING:

Now, we divide our dataset into training and testing data. Our objective for doing this split is to assess the performance of our model on unseen data and to determine how well our model has generalized on training data. This is followed by a model fitting which is an essential step in the model building process.

Model Evaluation and Predictions:

This is the final step, in which we assess how well our model has performed on testing data using certain scoring metrics, I have used 'accuracy score' to evaluate my model. First, we create a model instance, this is followed by fitting the training data on the model using a fit method and then we will use the predict method to make predictions on x_test or the testing data, these predictions will be stored in a variable called y_test_hat. For model evaluation, we will feed the y_test and y_test_hat into the accuracy_score function and store it in a variable called test_accuracy, a variable that will hold the testing accuracy of our model. We followed these steps for a variety of classification algorithm models and obtained corresponding test accuracy scores

3.3.2 User Module

Flask webapp

We develop a webapp using flask framework.User can view data analysis for different type of attacks with graphical representation and feature description along with attack type prediction.3.4 Architecture

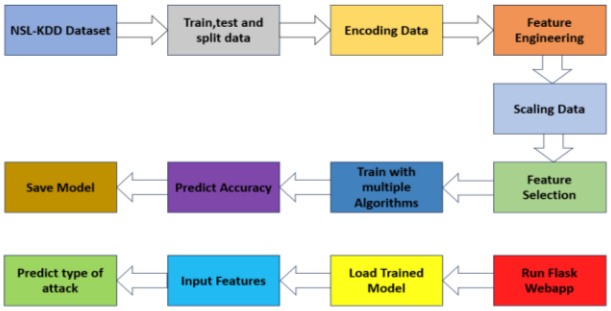
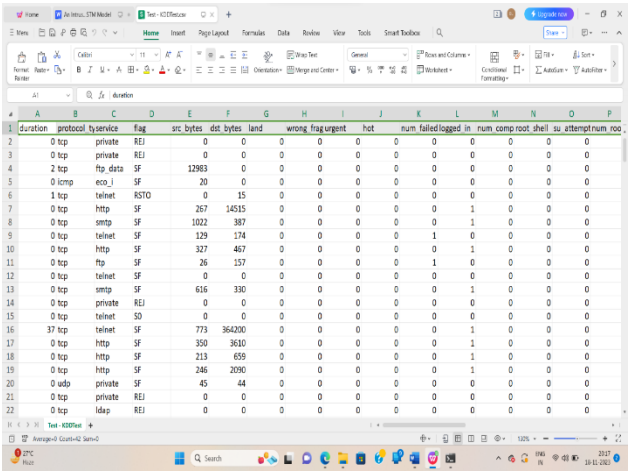
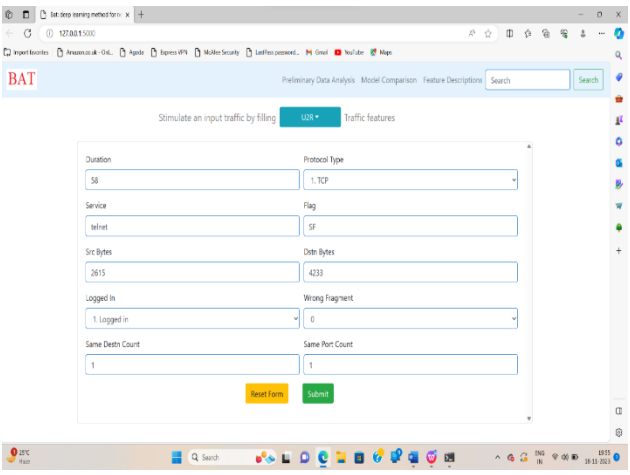
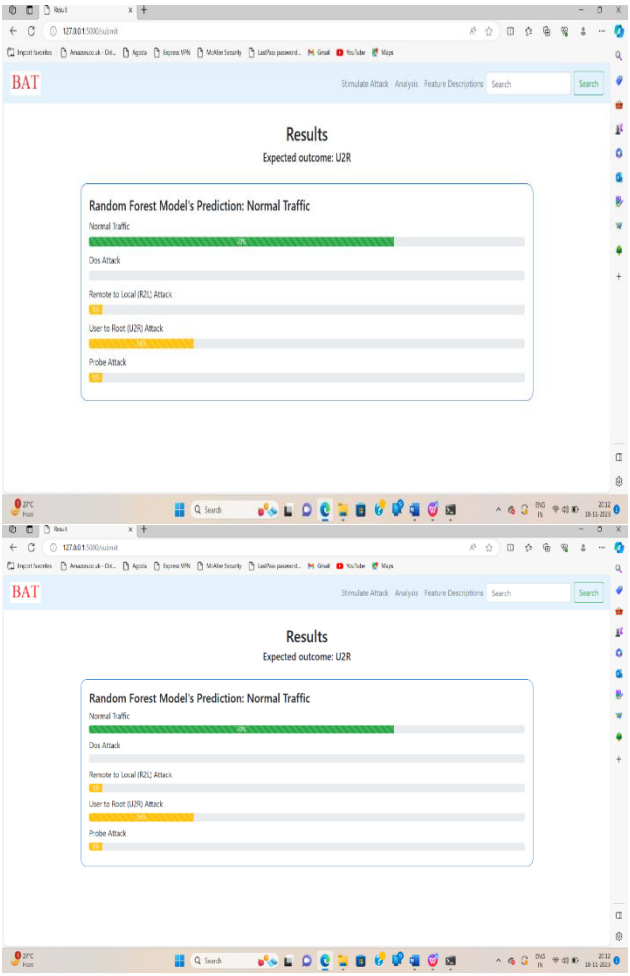


Fig 1: System Architecture

4. RESULT SCREEN SHOTS







5. CONCLUSION

As network intrusion continues to evolve, the pressure on network intrusion detection is also increasing. In particular, the problems caused by imbalanced network traffic make it difficult for intrusion detection systems to predict the distribution of malicious attacks, making cyberspace security face a considerable threat. This project proposed a novel Difficult Set Sampling Technique, which enables the classification model to strengthen imbalanced network data learning. A targeted increase in the number of minority samples that need to be learned can reduce the imbalance of network traffic and strengthen the minority's learning under challenging samples to improve the classification accuracy.

We used six classical classification methods in machine learning and deep learning and combined them with other sampling techniques. Experiments show that our method can accurately determine the samples that need to be expanded in the imbalanced network traffic and improve the attack recognition more effectively. In the experiment, we found that deep learning performance is better than machine learning after sampling the imbalanced training set samples through the MLP algorithm. Although the neural networks strengthen data expression, the

current public datasets have already extracted the data features in advance, which is more limited for deep learning to learn the pre-processed features and cannot take advantage of its automatic feature extraction. Therefore, in the next step, we plan to directly use the deep learning model for feature extraction and model training on the original network traffic data, performance the advantages of deep learning in feature extraction, reduce the impact of imbalanced data and achieve more accurate classification.

6. REFERENCES

[1] D. E. Denning, ‘‘An intrusion-detection model,’’ IEEE Trans. Softw. Eng., vol. SE-13, no. 2, pp. 222–232, Feb. 1987.

[2] N. B. Amor, S. Benferhat, and Z. Elouedi, ‘‘Naive Bayes vs decision trees in intrusion detection systems,’’ in Proc. ACM Symp. Appl. Comput. (SAC), 2004, pp. 420–424.

[3] M. Panda and M. R. Patra, ‘‘Network intrusion detection using Naive Bayes,’’ Int. J. Comput. Sci. Netw. Secur., vol. 7, no. 12, pp. 258–263, 2007.

[4] M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, ‘‘Support vector machine and random forest modeling for intrusion detection system (IDS),’’ J. Intell. Learn. Syst. Appl., vol. 6, no. 1, pp. 45–52, 2014.

[5] N. Japkowicz, ‘‘The class imbalance problem: Significance and strategies,’’ in Proc. Int. Conf. Artif. Intell., vol. 56, 2000, pp. 111–117.

[6] Y. LeCun, Y. Bengio, and G. Hinton, ‘‘Deep learning,’’ Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[7] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, ‘‘Deep learning for visual understanding: A review,’’ Neurocomputing, vol. 187, pp. 27–48, Apr. 2016.

[8] T. Young, D. Hazarika, S. Poria, and E. Cambria, ‘‘Recent trends in deep learning based natural language processing [review article],’’ IEEE Comput. Intell. Mag., vol. 13, no. 3, pp. 55–75, Aug. 2018.

[9] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, ‘‘A deep learning approach to network intrusion detection,’’ IEEE Trans. Emerg. Topics Comput. Intell., vol. 2, no. 1, pp. 41–50, Feb. 2018.

[10] D. A. Cieslak, N. V. Chawla, and A. Striegel, ‘‘Combating imbalance in network intrusion datasets,’’ in Proc. IEEE Int. Conf. Granular Comput., May 2006, pp. 732–737.