

RESEARCH

Open Access



IDS-attention: an efficient algorithm for intrusion detection systems using attention mechanism

FatimaEzzahra Laghrissi^{1*} , Samira Douzi², Khadija Douzi¹ and Badr Hssina¹

*Correspondence:
fatimaezzahra.laghrissi@etu.
fstm.ac.ma
¹ FSTM University Hassan II,
Casablanca, Morocco
Full list of author information
is available at the end of the
article

Abstract

Network attacks are illegal activities on digital resources within an organizational network with the express intention of compromising systems. A cyber attack can be directed by individuals, communities, states or even from an anonymous source. Hackers commonly conduct network attacks to alter, damage, or steal private data. Intrusion detection systems (IDS) are the best and most effective techniques when it comes to tackle these threats. An IDS is a software application or hardware device that monitors traffic to search for malevolent activity or policy breaches. Moreover, IDSs are designed to be deployed in different environments, and they can either be host-based or network-based. A host-based intrusion detection system is installed on the client computer, while a network-based intrusion detection system is located on the network. IDSs based on deep learning have been used in the past few years and proved their effectiveness. However, these approaches produce a big false negative rate, which impacts the performance and potency of network security. In this paper, a detection model based on long short-term memory (LSTM) and Attention mechanism is proposed. Furthermore, we used four reduction algorithms, namely: Chi-Square, UMAP, Principal Components Analysis (PCA), and Mutual information. In addition, we evaluated the proposed approaches on the NSL-KDD dataset. The experimental results demonstrate that using Attention with all features and using PCA with 03 components had the best performance, reaching an accuracy of 99.09% and 98.49% for binary and multiclass classification, respectively.

Keywords: Intrusion detection systems, Deep learning, Attention mechanism, LSTM, UMAP, Chi-Square, PCA, Mutual information

Introduction

The rapid growth of the internet has established an environment in which millions of machines around the world are connected. Thus, the data saved on our personal machines has become considerably more valuable. Furthermore, with most companies accepting working from home, networks become more exposed to information stealing, and destruction. Additionally, access to the internet network is omnipresent and relatively low-priced, allowing any cybercriminals in the world to lead a network attack, nevertheless to their physical position.

Network attacks are illegitimate acts against private resources that target computer information systems, infrastructures, and personal computers, with the purpose of modifying, destroying, or stealing sensitive data. We distinguish two main types of network attacks:

- Passive: Hackers obtain unauthorized access to networks, in order to examine and scan for open ports and vulnerabilities. In addition, hackers monitor all transmissions and copy the content of the messages. Nevertheless, there is no change to the data gathered or systems.
- Active: In which the attacker uses data gathered throughout a passive attack to compromise a computer or network. Furthermore, the hacker attempts to modify, delete, encrypt, or damage private data. These attacks affect the integrity and availability of the system.

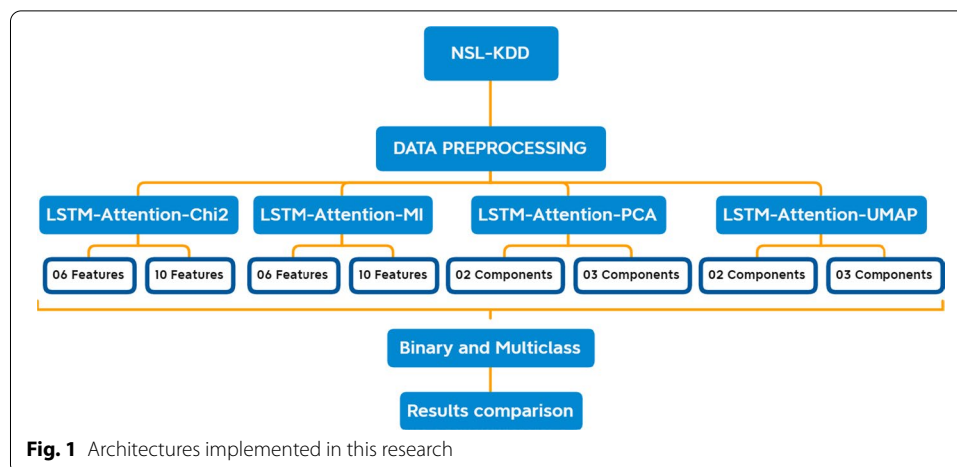
For all these reasons, cyber security and vigilance should be a priority in all industries. Fortunately, computer and network security products grow and expand in order to adapt and reflect the threats facing them. Among all these products, intrusion detection systems are the most important.

An intrusion detection system (IDS) is a vital element of a truly successful security solution. The general purpose of an IDS is to monitor network traffic for suspicious activity and known threats. Once any potential threats have been identified, IDS inform the IT manager that a network intrusion may be taking place. Reported information will usually contain the IP source address of the intrusion, the target/victim address, and the class of attack that is suspected. Additionally, an IDS comes in one of two types:

- A host intrusion detection system (HIDS): Runs on the computers on which it is installed, monitoring, and analyzing the processes and applications.
- A network intrusion detection system (NIDS): Implemented at a crucial point or points within the network, where it can analyze and examines the network traffic.

However, intrusion detection systems are prone to many challenges, among them: false positives rate and the false negatives rate. A false positive is a false alarm. It occurs when the IDS flags an activity as an attack, but the activity has acceptable behavior. Despite their failures, false positives do not generally cause grave damage to the network. On the other hand, a false negative is when the IDS fail to detect an attack. It occurs when the IDS identifies an activity as acceptable when the activity is actually an attack. Moreover, this is the most dangerous state, since IT professionals do not know that an attack is taking place.

In this study, our contribution consist of implementing an intrusion detection system based on LSTM neural network and Attention architecture (Fig. 1). Furthermore, in order to remove unimportant and noisy features that decrease the classification accuracy, four reduction algorithms were used, namely: Chi-square (Chi2), UMAP, Principal Components Analysis (PCA), and Mutual Information (MI). Moreover, the effectiveness of these approaches was tested on the well-known NSL-KDD dataset, for binary and multiclass classification.



Related works

Intrusion detection systems are a pure classification problem. Since the first IDS introduced by Denning [1], myriad methods have been used into network security fields. Moreover, with the consistent development of big data as well as the increase in computational power, several deep learning approaches have been used in intrusion detection. Furthermore, deep learning can handle big data efficiently, and has the ability to extract the representative characteristics from raw data, therefore, many researchers focused their efforts on deep learning techniques to create powerful IDSs.

Ramadan [2] proposed a hybrid IDS system where a pre-processing phase is utilized to reduce the required time. The feature selection process is done by using the Enhanced Shuffled Frog Leaping (ESFL) algorithm, and the selected features are classified using the Light Convolutional Neural Network with Gated Recurrent Neural Network (LCNN-GRNN) algorithm. Maha [3] Designed an intelligent BBFO-GRU intrusion detection systems in Industrial Cyber-Physical environment based on the Gated Recurrent Unit (GRU) model. In addition, in order to enhance the detection rate, NADAM optimizer is utilized to optimize the GRU hyperparameters. Derhab [4] designed a Temporal Convolution Neural Network (TCNN) in IoT, which combines the Convolution Neural Network (CNN) with a causal convolution. TCNN with Synthetic Minority Oversampling Technique-Nominal Continuous (SMOTE-NC) is evaluated on Bot-IoT dataset. Mulyanto [5] implemented a cost-sensitive neural network based on focal loss, called the focal loss network intrusion detection system (FL-NIDS), in order to overcome the problem of imbalanced data. FL-NIDS was applied using DNN and convolutional neural network (CNN). To evaluate this approach, three benchmark intrusion detection datasets that suffer from imbalanced distributions were used: NSL-KDD, UNSW-NB15, and Bot-IoT. Azmin [6] proposed a new paradigm of the synthesizing task based on Variational Laplace AutoEncoder (VLAe), and Deep Neural Network (DNN) classifier. The authors evaluated the model on the NSL-KDD dataset. Jie [7] proposed an Intrusion Detection System based on bidirectional simple recurrent unit. In addition, the skip connections is used to to alleviate the vanishing gradient problem and improve the training effectiveness. Mahboob [8] employed the butterfly optimization algorithm (BOA), and meta-heuristic to perform feature selection. A multilayer perceptron (MLP) classifier was

used to evaluate the capability of the selected features to predict attacks. In addition to the gradient descent (GD) training method, two other metaheuristic methods, particle swarm optimization (PSO) and genetic algorithm (GA) were used to optimize the classification structure. This approach was tested on the NSL-KDD dataset. Sahar [9] developed a network intrusion detection system based on deep learning, and implemented in the fog node for attack detection. The datasets used are UNSW-NB15 and NSL-KDD. Khan [10] conceived an intrusion detection system, based on convolutional neural network algorithm. The entire network consists of three hidden layers. Each hidden layer contains a convolutional layer and a pooling layer. Bediya [11] discussed many possible attacks at IoT networks and distributed denial of service (DDoS) attack. Then, the author proposed a blockchain-based IDS for the IoT network, called BIoTIDS. Khan [12] implemented a convolutional recurrent neural network (CRNN) to create a DL-based hybrid ID framework that predicts and classifies malicious cyberattacks in the network. In the HCRNNIDS, the convolutional neural network (CNN) performs the convolution to capture local features, and the recurrent neural network (RNN) captures temporal features to improve the ID system's performance and prediction. Experiments were carried out on the CSE-CIC-DS2018 dataset. Soumyadeep [13] presented an unique Generic-Specific autoencoder model where the generic one learns the features that are common across all forms of network intrusions, and the specific ones learn features that are pertaining only to that domain. Sekhar [14] applied a deep Autoencoder with Fruitfly Optimization. Firstly, the missing values in the dataset have been imputed with the Fuzzy C-Means Rough Parameter (FCMRP) algorithm, which handles the imprecision in datasets with the exploit of fuzzy and rough sets while preserving crucial information. Then, robust features are extracted from the Autoencoder with multiple hidden layers. Finally, the obtained features are fed to the backpropagation neural network (BPN) to classify the attacks. Experiments have been carried out on the NSL-KDD and UNSW-NB15 dataset. Khonde [15] proposed a hybrid method, based on semi-supervised machine learning classifiers. Moreover, classifiers used are Support vector machine, decision tree and k-nearest neighbor. Experiments were conducted on NSL-KDD dataset. Shen [16] proposed an ensemble method, combining the extreme learning machine (ELM) as a base classifier, and a pruning method based on the Bat Algorithm (BA) as an optimizer. Deepa [17] used the K-Means Algorithm features. Moreover, authors combined Cuckoo Search Optimization (CSO) and the K-Means clustering algorithm. This approach was tested on different datasets. Divakar [18] used an ensemble method based on XGB Classifier on UNSW-NB 15 dataset.

Basic concepts

Long short-term memory (LSTM)

A recurrent neural network (RNN) is a class of artificial neural networks where the data of the previous step is fed as input to the next step. However, the main issue with RNNs is gradient vanishing and exploding problems during back propagation. To overcome this problem, Hochreiter and Schmidhuber [19] introduced Long Short-Term Memory (LSTM) in 1997. Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks able to learn information from earlier time steps to later ones. Unlike conventional feedforward neural networks, with LSTM the data flows through a

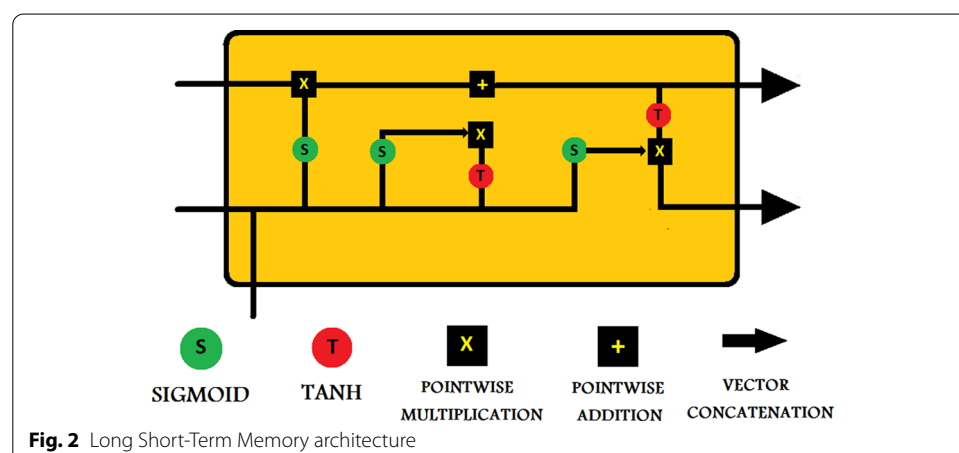
mechanism known as cell states. This way, LSTMs can selectively remember or forget information. Thus, its gating mechanism is what solved the “short-term memory” problem of RNNs. A common LSTM unit is made of a cell, an input gate, an output gate, and the forget gate. The cell remembers data over random time intervals, and the three gates control the stream of information into and out of the cell (Fig. 2). LSTM is suitable for myriad tasks such as: handwriting recognition [20], speech recognition [21], and anomaly detection in network traffic or IDSs (intrusion detection systems) [22].

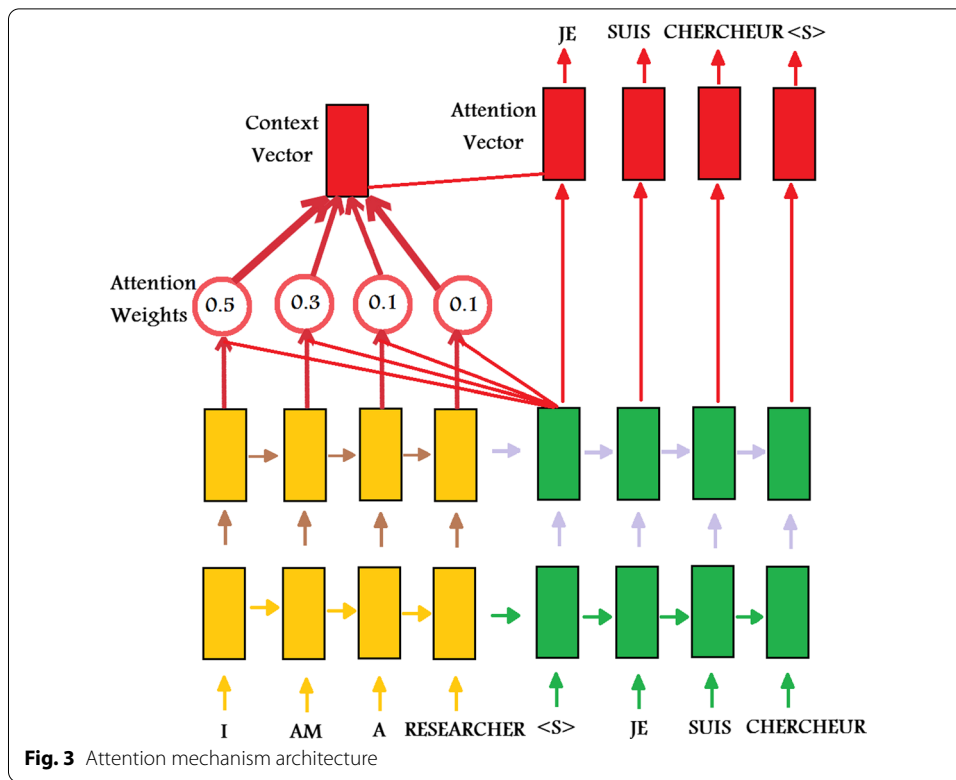
Attention mechanism

The attention mechanism is one of the most important ideas in deep learning research in the last decade. Moreover, it is an approach that imitates cognitive attention. Even though this technique is now used in a broad category of artificial intelligence models, including in natural language processing [23] and computer vision [24]. However, it was initially created over Seq2Seq models in the Neural Machine Translation domain. A basic seq2seq approach consists of an encoder-decoder model, where the encoder analyzes the input data and compresses the information into a context vector of a fixed length (sentence embedding), and the decoder is computed with the context vector to emit the transformed output. Furthermore, this architecture has shown its huge strengths in Seq2Seq challenges, still, it has one crucial drawback. The sentence embedding is generated in one vector; consequently, as the length of the input data increases, the more difficult it becomes for the model to capture the information in this vector. Thus, it has the inability to preserve longer input data as it tends to forget parts of it.

The attention mechanism was introduced by Bahdanau [25], in order to help memorize long source sentences in neural machine translation. Rather than constructing a single context vector, the attention mechanism creates shortcuts between the context vector and the entire source input. The weights of these shortcut connections are adjustable for each output feature (Fig. 3). The effect increases the important parts of the input data and fades out the rest.

Since not all the inputs would be used in generating the corresponding output, The attention mechanism calculates multiple attention weights marked by





$\alpha(t, 1), \alpha(t, 2), \dots, \alpha(t, t)$. The context vector C_i for the output result y_i is produced applying the weighted sum of the annotations:

$$C_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j.$$

The attention weights are computed by normalizing the output score of a feed-forward neural network described by the function that captures the alignment between input at j and output at i . The weights α_{ij} are computed by a softmax function given by the following equation:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})},$$

$$e_{ij} = a(s_{i-1}, h_j)$$

e_{ij} is the output score of a feedforward neural network described by the function a that attempts to capture the alignment between input at j and output at i .

Dimensionality reduction

Dimensionality reduction is the process of reducing the number of input data from a high-dimensional space to a low-dimensional space, so that the new input dimension contains most characteristics of the raw data.

High-dimensionality statistics and dimensionality reduction methods are commonly applied for data visualization. Nonetheless, these methods can be implemented in

machine learning to clarify the dataset in order to better fit a predictive model. Moreover, more input features usually make a predictive modeling task harder, more often called the curse of dimensionality. Thus, the higher the number of features, the harder it is to visualize the training set and then work on it. Furthermore, working in high-dimensional spaces can be undesirable for many reasons; raw data are often sparse, also most of these features are correlated, and hence redundant, therefore, analyzing the data is usually computationally expensive. This is where dimensionality reduction algorithms come into play. It is desirable to have simple models that generalize well and, in turn, input data with few input variables. Hence, it is often desirable to reduce the number of input features.

There two majors components of dimensionality reduction:

- Feature selection: is the process of identifying and selecting relevant features from the input variables using scoring or statistical methods.
- Feature extraction: is the process of generating, from the high-dimensional input data, new data of fewer dimensions.

Uniform manifold approximation and projection (UMAP)

UMAP (Uniform Manifold Approximation and Projection) is an innovative manifold learning algorithm for dimension reduction, invented by Leland McInnes et al. [26]. Moreover, UMAP is built from a theoretical framework based in Riemannian geometry and algebraic topology. Furthermore, the UMAP algorithm arguably conserves more of the global structure with higher performance, and no computational restrictions on embedding dimension. In addition, UMAP is among the fastest manifold learning application available. Moreover, UMAP consist of two principal stages:

- Creating a graph in high dimensions and calculating the bandwidth of the exponential probability, σ , through the binary search and the fixed number of the nearest neighbors.
- Applying Stochastic Gradient Descent (SGD) in order to optimize the low-dimensional representation, in order to improve the computation speed.

UMAP calculates the exponential probability distribution in high dimensions as:

$$p_{i|j} = e^{-\frac{d(x_i, x_j) - p_i}{\sigma_i}}$$

where p represents the distance from each i – th data point to its first nearest neighbor.

Moreover, UMAP uses the number of the nearest neighbors k as follows:

$$k = 2^{\sum_i p_{ij}}$$

Also, the symmetrization of the high-dimensional probability is calculated as :

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i}$$

Chi-Square (χ^2)

The Chi-Squared test or The Pearson's [27] Chi-Squared test is a statistical theory test, applied to check the independence of two variables. Furthermore, chi-square technique is applied in feature selection by calculating the chi-square statistics between all the features and the target variable, and examine the presence of a relationship between the features and the target. If the target variable is independent of the feature variable, we can throw away that feature variable. If they are dependent, the feature variable is significant and crucial. Likewise, the Chi-Squared statistics are calculated using the following formula:

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

where "O" stands for observed or actual values and "E" stands for expected values. If these two value are independent, O and E will be close, and if they have some association then the Chi-squared value will be high.

Principal component analysis (PCA)

Principal component analysis (PCA) is the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest. Moreover, it is a technique for feature extraction that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. Furthermore, PCA extracts the eigenvectors and eigenvalues from the covariance matrix (CM) using the following formula:

$$CM = \frac{1}{n-1} ((X - x')^T (X - x'))$$

where x' is the mean vector $x' = (\frac{1}{n}) \sum_{i=1}^n (x_i)$. And the covariance between two features :

$$Cv_{jk} = \left(\frac{1}{n-1} \right) \sum_{i=1}^n (x_{ij} - x'_j)(x_{ik} - x'_k)$$

Mutual information (MI)

Mutual information is one of many quantities that indicates how much information can be obtained from a random variable by observing another random variable. Furthermore, in probability theory and information theory, the mutual information (MI) of two random variables is a measure of the mutual dependence between the two variables. Therefore, a high mutual information value indicates a large reduction of uncertainty whereas a low value indicates a small reduction. If the mutual information is zero, that means that the two random variables are independent.

Moreover, the mutual information between two variables X AND Y denoted $I(X; Y)$, is defined by Shannon and Weaver [28] as:

$$I(X; Y) = \sum_{x,y} P_{XY}(x, y) \log \frac{P_{XY}(x, y)}{P_X(x)P_Y(y)} = E_{P_{XY}} \log \frac{P_{XY}}{P_X P_Y}$$

Here $P_X(x)$ and $P_Y(y)$ are the marginals: $P_X(x) = \sum_y P_{XY}(x, y)$

Our approach

This section includes the dataset description and preprocessing, calculation of the Chi-square scores, a data visualization using UMAP, calculation of PCA variance, calculation of the mutual information scores, implementation parameters of the proposed model, experimental results, and discussion.

The dataset and data preprocessing

NSL-KDD [29] is a dataset proposed to address some of the intrinsic issues of the KDD'99 [30] dataset [31]. Furthermore, the number of records in the NSL-KDD is lower; therefore, no data sampling or filtering is required. Consequently, the evaluation results of different research work will be consistent and comparable. Moreover, redundant and duplicate records are removed, so the classifiers will not be biased towards more frequent records.

Originally, the KDD99 dataset contained 3,925,650 attack record, 972,781 normal records, and a total of 4,898,431 records. However, The NSL-KDD dataset contains 262,178 attacks records, 812,814 normal records, and a total records number of 1,074,992, with a total reduction rate of 78.05%. The statistics of the NSL-KDD records are shown in Fig. 4.

In the NSL-KDD, we can find multiples attacks, each of them contains different sub-classes (Fig. 5). The four major attacks classes are:

- Denial of Service (DOS): DOS attacks are designed to exhaust the target system in order to shut down a machine or network, making it inaccessible to its intended users.
- Probing attacks (Probe): Probe attacks are designed to obtain more information about the target system.
- Remote to Local (R2L): R2L attacks are designed to give local access to target system; thus, they are more dangerous than DOS and probe attacks.

DATASET	Number of records					
	Total	Normal	DoS	Probe	U2R	R2L
KDDTrain+20%	25192	13449 (53%)	9234 (37%)	2289 (9.16%)	11 (0.04%)	209 (0.8%)
KDDTrain+	125973	67343 (53%)	45927 (37%)	11656 (9.11%)	52 (0.04%)	995 (0.85%)
KDDTest+	22544	9711 (43%)	7458 (33%)	2421 (11%)	200 (0.9%)	2654 (12.1%)

Fig. 4 Number of instances for all the attacks

Classes	DoS	Probe	U2R	R2L
Sub-Classes	<ul style="list-style-type: none"> • apache2 • back • land • neptune • mailbomb • pod • processtable • smurf • teardrop • udpstorm • worm 	<ul style="list-style-type: none"> • ipsweep • mscan • nmap • portsweep • saint • satan 	<ul style="list-style-type: none"> • buffer_overflow • loadmodule • perl • ps • rootkit • sqlattack • xterm 	<ul style="list-style-type: none"> • ftp_write • guess_passwd • httptunnel • imap • multihop • named • phf • sendmail • Snmpgetattack • spy • snmpguess • warezclient • warezmaster • xlock • xsnoop
	11	6	7	15

Fig. 5 Different subclasses of each attack that exists in the data set

- User to Root (U2R): U2R attacks give root access (super-user) to the normal user. Initially attacker access normal user account, later gain access to the root by exploiting the vulnerabilities of the system. Since root can do anything in system, U2R attacks are the most dangerous of all attacks in this dataset.

The NSL-KDD dataset contains 43 features, that can be divided in 4 types:

- 4 Categorical (Features: 2, 3, 4, 42)
- 6 Binary (Features: 7, 12, 14, 20, 21, 22)
- 23 Discrete (Features: 8, 9, 15, 23–41, 43)
- 10 Continuous (Features: 1, 5, 6, 10, 11, 13, 16, 17, 18, 19)

In the preprocessing phase, we first changed all the subclasses to their respective classes, then, we OneHotEncode the protocol_type (feature number 2) and the flag(feature number 4). We choose to OneHotEncode only these two features because they 3 and 11 possible values. On the other hand, we LabelEncode the feature named service (feature number 3) because it contains 60 possible values. In this step, we get 77052 normal records, 53,386 DoS records, 14,077 Probe records, 3880 R2L records, and 119 U2R records. Next, we used the MinMaxScaler, which transform features by scaling each feature to a given range, we choose to set this range between 0 and 1. Afterwards, we shuffled the data which help us reducing variance and making sure that models remain general and overfit less. Consequently, after the preprocessing phase, we end up with 53 features. Finally, we prepared the data to fit the binary and multiclass classification.

Multiclass classification is the problem of classifying instances into one of three or more classes. Thereby, we sorted the attacks into five groups, which are: Normal (0), DoS (1), Probe (2), R2L (3), and U2R (4), and replaced these the feature named label (feature number 42) with these numbers instead of attacks names.

On the other hand, binary classification refers to the classification tasks that have two class labels. Therefore, our binary classifier should have the ability to judge whether a given input is a normal record or not. Thus, we encode the labels into two integers: 0 represents the normal records, while 1 represents the attack records.

Finally, two dataset were generated, the first one is “NSL-Binary.csv”, destined for binary classification, and the second one named “NSL-Multiclass.csv” intended to multiclass classification.

In addition, normal records and Denial of Service (DoS) attacks represent the majority of the dataset, while R2L and U2R, are very rare in NSL-KDD (Fig. 6). Thus, this data set is widely imbalanced. Consequently, this issue affects the generalization of the model and reduces the classifier efficiency to predict minority classes, leading the model to fail in the classification task. This issue affects mostly the multiclass classification (as shown in Fig. 6).

Dimensionality reduction

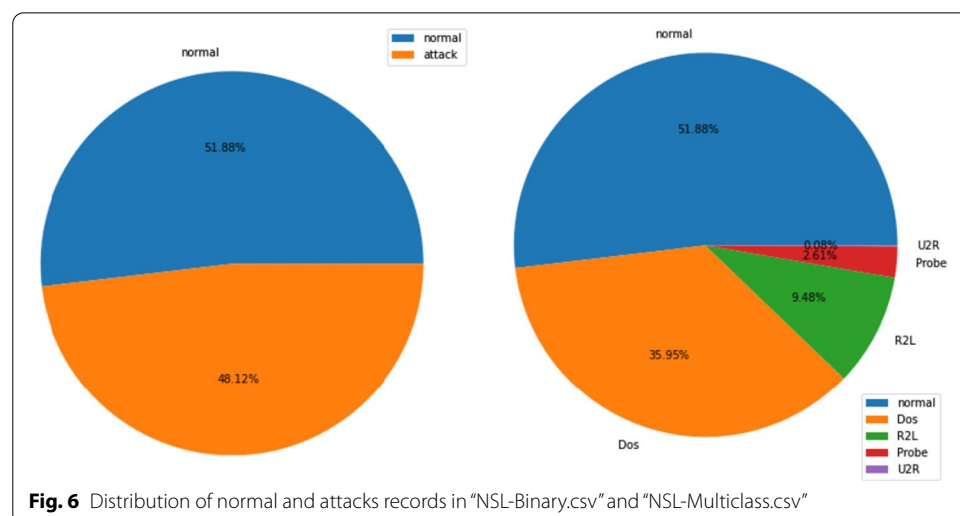
Chi-Square

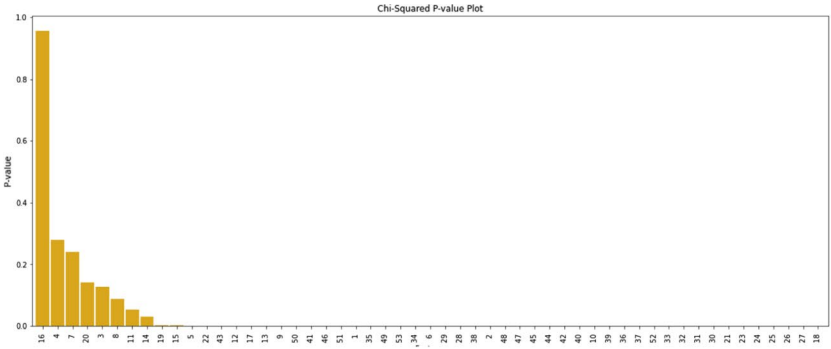
As we mention above, Chi-square is feature selection technique, that operate a statistical test, between every feature and the target variable, in order to investigate the presence of a relationship between the feature and the target. Furthermore, the Chi-square test has two important outcomes, namely: P-value and Chi Score.

When a p-value is higher, it means that the input feature is independent of the target and can not be considered for model training, thus, we can discard it. On the other hand, chi-score is a value attributed to every feature, demonstrating the impact of these features on the target variable.

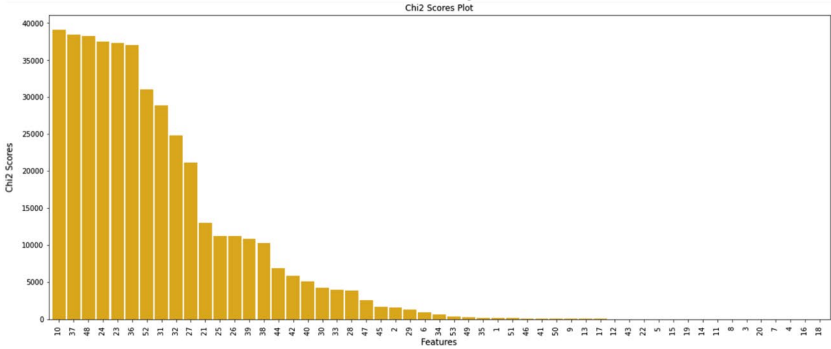
The p-value and chi-score are presented in Fig. 7, for both generated datasets.

From Fig. 7a, we can observe that features 16, 4, 7, 20, 3, 8, 11 and 14 have the highest p-values, which means that these variables and the output variable are independent. Thus, we can remove it. However, features 10, 37, 48, 24, 23, 36, 52, 31, 32, 27, 21, 25, 26, 39 and 38 in Fig. 7b have notably high chi scores, which means that the association between these variables and the target variable is statistically significant.

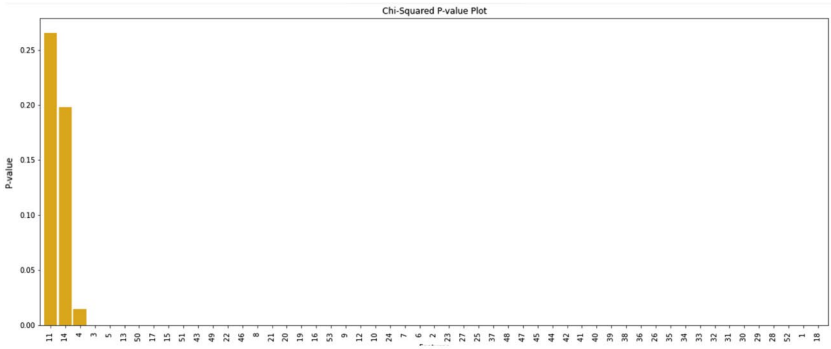




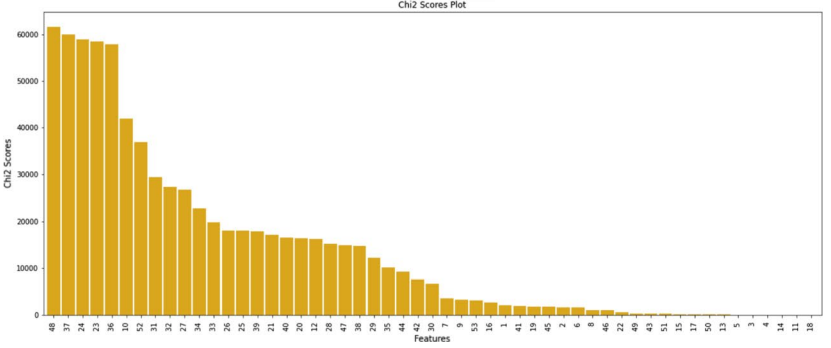
(a) Chi-square p-values using "NSL-Binary.csv" scores.png



(b) Chi-square scores using "NSL-Binary.csv"



(c) Chi-square p-values using "NSL-Multiclass.csv" scores.png



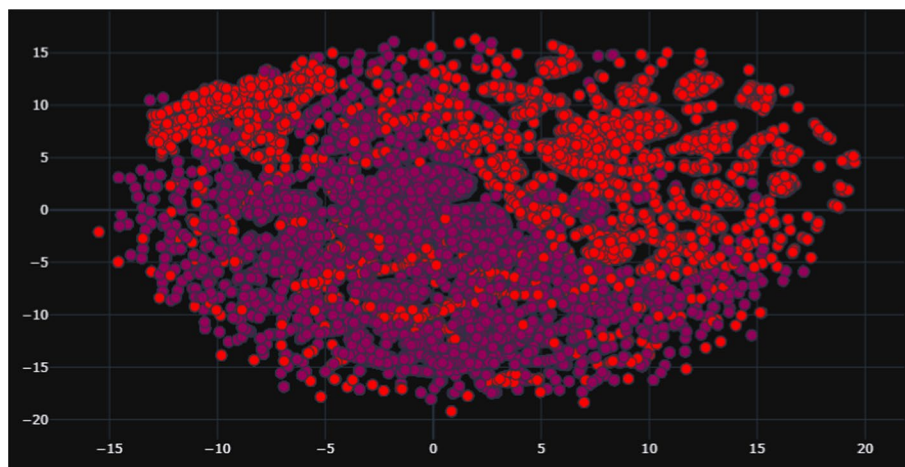
(d) Chi-square scores using "NSL-Multiclass.csv"

Fig. 7 Chi-Square calculus

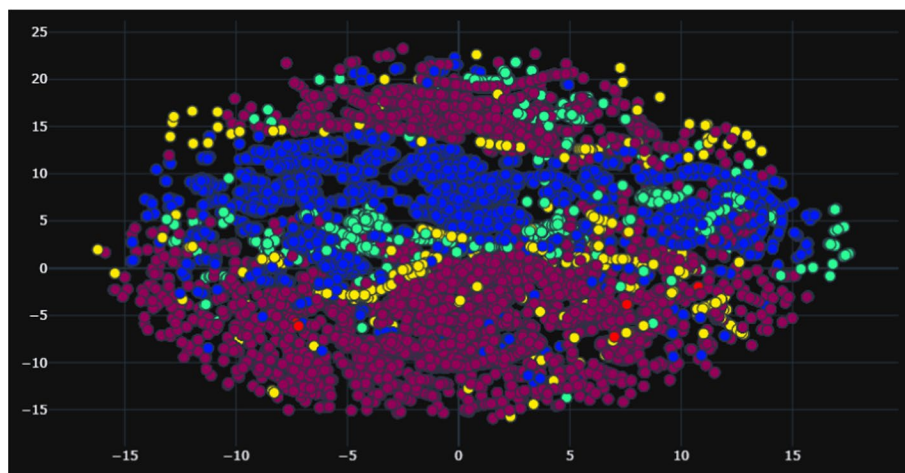
On the other hand, using “NSL-Multiclass.csv”, features 11, 14 and 4 have a higher p-value, therefore, these variables are independent and have no impact on the results (Fig. 7c). While features 48, 37, 24, 23, 36, 10, 52, 31, 32, 27, 34, 33, 26, 25, 39, 21, 40, 20, 12, 28, 47, 38, 29, 35, 44, 42, and 30 have a slightly higher chi-square score which means that these variables impact considerably the final score (Fig. 7c).

UMAP

UMAP is another method for data visualization and dimensionality reduction. Furthermore, it employs graph layout algorithms to organize data in low-dimensional space. The Fig. 8 shows a projection of the 53-dimensional NSL-KDD dataset show to 2 dimensions, using “NSL-Binary.csv” (Fig. 8a) and using “NSL-Multiclass.csv” (Fig. 8b). As we can see, UMAP can’t split clearly these output categories from each other, especially using “NSL-Multiclass.csv”. Consequently, there are no big clusters between the sign



(a) UMAP projection in “NSL-Binary.csv”



(b) UMAP projection in “NSL-Multiclass.csv”

Fig. 8 UMAP projection

sufficiently, thus, there are similar data points agglomerated together in other parts too from a 2d prospective.

PCA

As we mention above, PCA is a technique to reduce the dataset dimensionality. Therefore, we calculated the PCA covariance using “NSL-Binary.csv” and “NSL-Multiclass.csv”.

As we can see in Fig. 9, that the first the first 03 components represent more than 80% of the dataset.

Mutual information

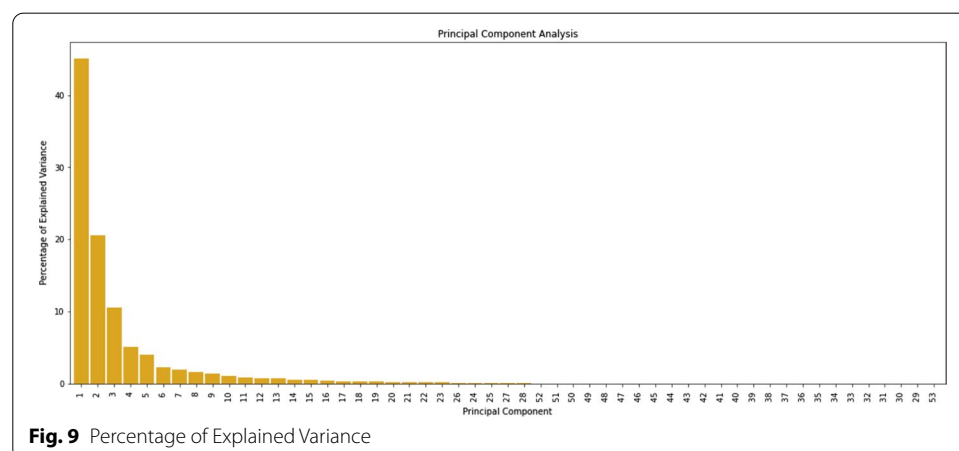
As we mentionned earlier, the mutual information calculates the statistical dependence between two variables. Thus, a score is assigned to each feature, showing how much the latter impacts the result.

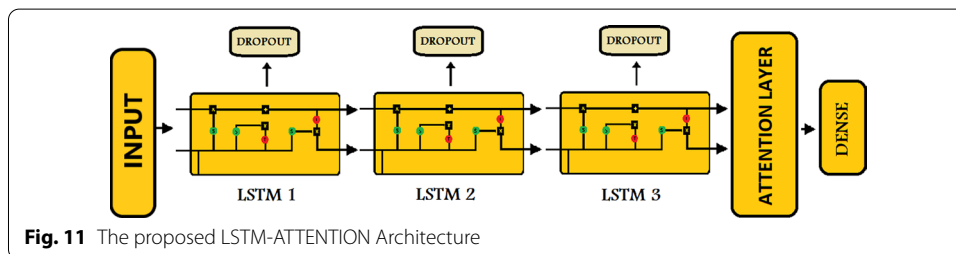
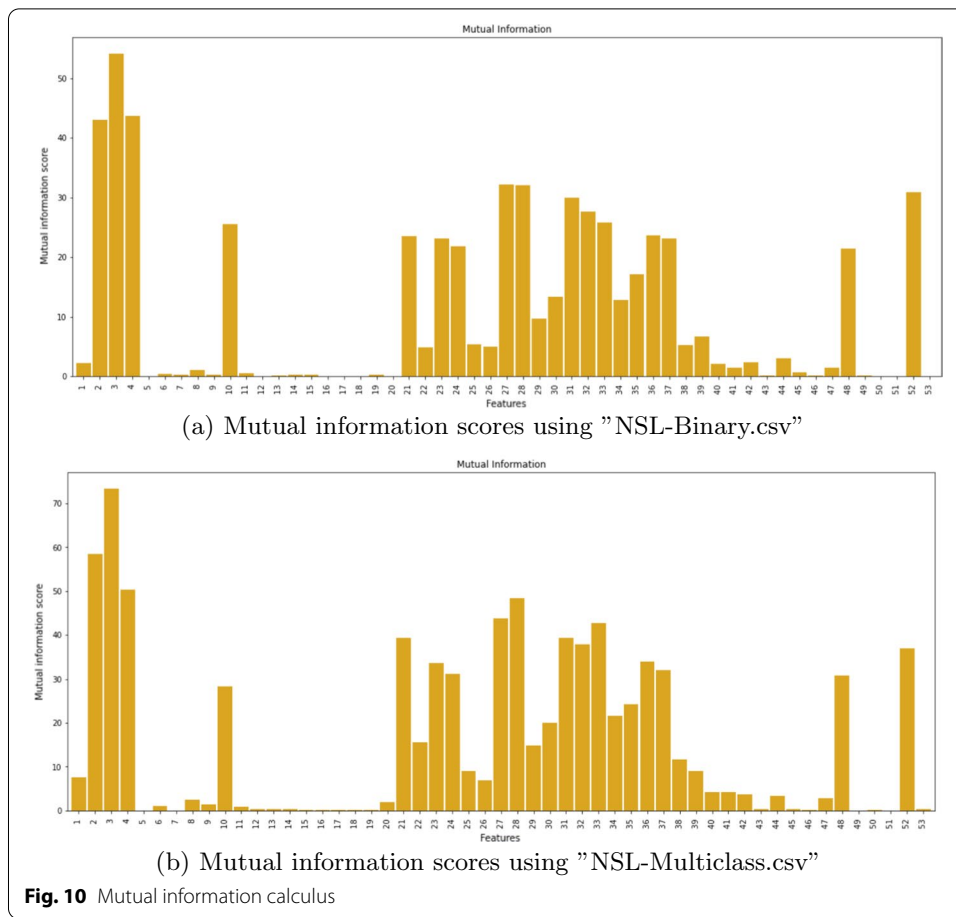
Therefore, we calculated, the mutual information scores, using “NSL-Binary.csv” and “NSL-Multiclass.csv” (Fig. 10).

Implementation and evaluation metrics

For the experiments, we implemented the proposed models using Keras Library. Keras is an open-source software library that provides a Python interface for artificial neural networks. However, the dimensionality reduction algorithms were applied using Scikit-learn, which is a free software machine learning library for the Python programming language. Furthermore, our tests were executed on Google Colab. Meanwhile, we divided the preprocessed dataset into train set, validation set, and test set, according to 60%, 20%, and 20% respectively.

In order to train our models, the dropout is set to 0.1, the number of epochs is set to 100, the schedule decay is set to 0.004, the epsilon is set to $1e-08$, the learning rate is set to 0.002, and the optimizer used is Adam. On the other hand, Sigmoid and Binary cross entropy are used as loss and activation function for binary classification, while Softmax and Sparse categorical cross entropy are used as loss and activation





function for multiclass classification. In addition, the proposed Lstm-Attention model is presented in Fig. 11.

To evaluate our detection models, the performances of the proposed architectures were calculated. Therefore, using the confusion matrix, we considered the true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), such as:

- TP: Actual attack is classified as attack.
- FP: Actual normal record is classified as attack.
- FN: Actual attack is classified as normal.
- TN: Actual normal is classified as normal.

Furthermore, the confusion matrix allows us to calculate more metrics, namely: Accuracy, recall, precision, f1 score, and misclassification rate. These metrics are measured as:

- The accuracy is the ratio of correctly predicted observations. Where:
 $Accuracy = (TP + TN) / AllPredictions$
- The recall is a proportion of correctly predicted positive events. Where:
 $Recall = TP / (FN + TP)$
- The precision means a ratio of correct positive observations. Where:
 $Precision = TP / (TP + FP)$
- The F1 score signifies the weighted average of precision and recall. Where:
 $F1Score = 2 * (Precision * Sensitivity) / (Precision + Sensitivity)$
- The misclassification rate is the percentage of incorrectly classified instances. Where:
 $Misclassification = (FP + FN) / AllPredictions$

Experimental results and discussion

In this research, we implemented LSTM classifier with multiple parameters. Firstly, several dimensionality reduction algorithms were applied with various parameters, namely: UMAP was used with 02 and 03 components, Chi-square was used with 06 and 10 features, PCA was applied using 02 and 03 components, whereas mutual information was employed using 06 and 10 features. Moreover, we added an attention layer to verify the impact of this architecture on the classification task, especially on the reduction of the false negative rate. Furthermore, all the models were applied in binary and multiclass classification.

Binary classification

Figure 12 shows the performance of all the models, using: Maximum training accuracy, testing accuracy, recall, precision, and the F1-score. Furthermore, Fig. 13 summarizes the confusion matrices for these architectures.

In this research, we observed that the classifier using the attention layer and all the features presented the best accuracy, recall, and f1-score. On the other hand, the LSTM model without attention obtained the highest precision score. Meanwhile, the Attention-PCA model using 02 components and Attention-MI with 06 features performed well.

As we mention earlier, the false negative rate generally cause grave damage to the network, and it is the most important metric to monitor. From Fig. 13, the model

	Attention-UMAP		Attention-Chi2		Attention-PCA		Attention-MI		All features	
	02 COMPONENTS	03 COMPONENTS	06 FEATURES	10 FEATURES	02 COMPONENTS	03 COMPONENTS	06 FEATURES	10 FEATURES	Without Attention	With Attention
Max Train Attention accuracy	94,86	90,85	98,14	98,01	98,33	98,11	97,77	98,78	98,56	99,09
Accuracy	91,09	75,93	96,80	97,44	98,18	98,07	97,66	96,24	98,74	98,84
Mis-Classification	0,08	0,24	0,03	0,02	0,01	0,01	0,02	0,03	0,01	0,01
Recall	89,03	85,56	95,72	97,71	97,81	98,25	96,87	97,22	98,10	98,40
Precision	92,99	66,99	97,80	97,19	98,53	97,89	98,39	95,34	99,34	99,25
F1-score	90,97	75,15	96,75	97,45	98,17	98,07	97,62	96,27	98,72	98,82


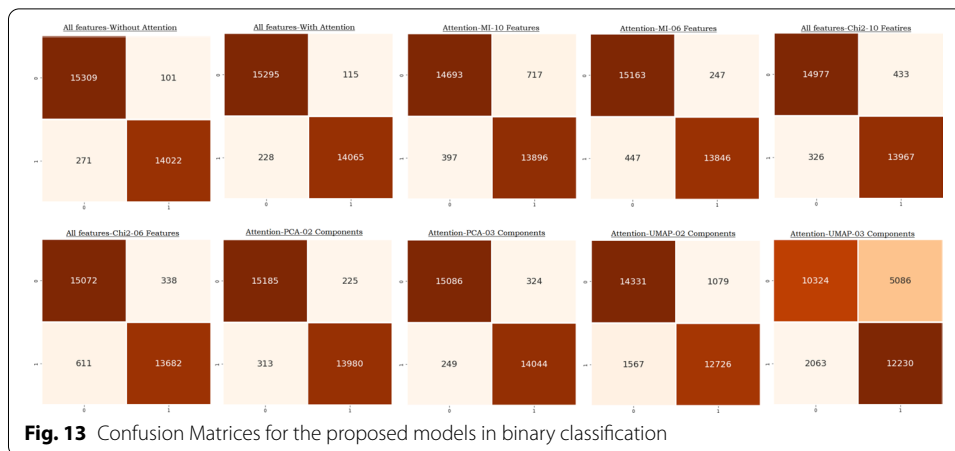

 Best scores Worst scores

Fig. 12 Results for binary classification



	Attention-UMAP		Attention-Chi2		Attention-PCA		Attention-MI		All features	
	02 COMPONENTS	03 COMPONENTS	06 FEATURES	10 FEATURES	02 COMPONENTS	03 COMPONENTS	06 FEATURES	10 FEATURES	Without Attention	With Attention
Max Train Attention accuracy	94,74	93,34	97,46	98,25	97,68	98,11	97,79	98,37	98,14	98,18
Accuracy	71,41	65,61	95,66	97,97	98,08	98,49	96,68	97,93	97,91	98,13
Recall	71,41	65,61	95,66	97,97	98,08	98,49	68,37	97,93	97,91	98,13
Precision	72,72	64,79	95,64	97,87	98,04	98,44	71,23	97,82	97,85	98,06
F1-score	69,63	64,08	95,41	97,76	98,06	98,44	69,66	97,79	97,77	98,01

Best scores Worst scores

Fig. 14 Results for multiclass classification

based on attention and using all features gets the best false negative rates, with only 228 attacks records detected as normal, from an initial 29.703 test records.

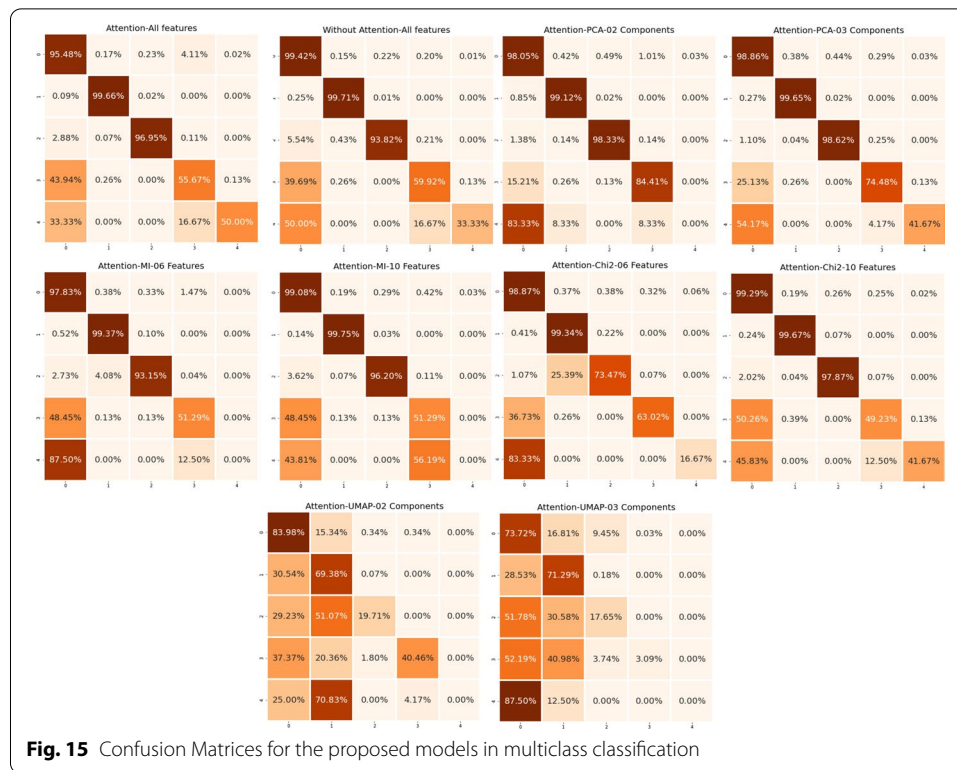
However, using the accuracy using UMAP with 03 components given the worst results. This infers that the UMAP-Attention has followed difficulty in learning the attack patterns. In addition, this architecture obtains the highest false negatives and true negatives rates, with 2063 and 5086 misclassified records respectively.

Multiclass classification

Figure 14 displays the achievements of all the architectures, using: Maximum training accuracy, Testing accuracy, Recall, Precision, and the F1-score.

Again, adding the attention layer enhance the all the metrics, especially the false negative rate which is the most important metric in our case.

Furthermore, the Attention-PCA model using 03 components offered the best accuracy, precision, recall, and f1-score within all the others. Moreover, the Attention-MI using 10 features gets the best training accuracy. In addition, using 02 PCA components make the model predict very well, and reach an accuracy of 98,13, which is the third best accuracy score. Also, we can remark the impact of the attention layer in the models using all features. Adding this layer allow the model to classify attack much better.

**Table 1** Performance comparison

	Ieracitano [32]	T. Su [33]	Ieracitano [32]	Choudhary [34]	Proposed
Accuracy	83.65%	84.25%	87.00%	91.50%	98.49%

On the other hand, the architectures using UMAP get the the worst scores, principally the model with 03 components. Furthermore, UMAP took the longest time to execute the reduction algorithms, and the longest time to train.

Moreover, Fig. 15 outlines the confusion matrices for these models. We can remark that the Attention-PCA architecture obtained the lowest false negatives rate and true negatives rate. On the other hand, the Attention-UMAP model with 03 components gets the highest false negatives and true negatives rates.

However, the only drawback of the PCA-03 components that the class number 4 which is U2R attacks, is frequently misclassified as normal. This may be due the record number of this attack, which represent only 0,08% of the dataset. The same problem happens to the model using attention and all feature, which is the second best model.

To improve the analysis of the experimental results, we compared our best model, which is the Attention-PCA using 03 components in multiclass classification, with those of previous researches (Table 1).

This kind of comparison is for reference purpose only, because IDSs differ in their execution environment, data preprocessing approaches, and interpretation process. Still, it can be seen that our model yields significantly better results than all the compared

models, which means that our approach is more adequate for this type of problem, and proves that our architecture has a better generalization and strength.

Conclusion and future works

In the study, an effective network attack detection strategy based on deep learning is presented. Moreover, Attention mechanism and Long Short-Term Memory(LSTM) were used as a classifier, in conjunction with numerous dimensionality reduction algorithms, namely: Chi-square, UMAP, PCA, and Mutual Information. Furthermore, multiple parameters were tested in order to obtain the best accuracy. Therefore, the model based on attention with all features and the model based on attention and PCA using 03 components obtained the best scores in binary and multiclass classification respectively, outperforming all the others. The Attention-PCA model is able to learn detailed features from the dataset in the training phase. This ability is important in learning characteristics to network traffic involved in anomaly intrusions to identify abnormal traffic from normal traffic.

The experimental results show that the proposed attack detection strategy achieves higher performance than previous strategies, using the NSL-KDD dataset, and it can also reduce the false negative rate.

Several avenues for future research have been identified. Firstly, we will apply more LSTM variants and evaluate the performance of complex LSTMs with dimensionality reduction algorithms. In addition, more experiments will be performed to further analyse the proposed Attention-PCA model using large data sets from published data sets. Also, the developed model will be improved to increase its detection accuracy further and the trade-offs between detection parameters.

Finally, we will try to overcome the unbalanced data problem, especially for the U2R and R2L attacks, which are the less represented classes in NSL-KDD dataset, using multiple numerical data augmentation techniques.

Acknowledgements

Not applicable.

Authors contributions

All authors read and approved the final manuscript.

Funding

Not applicable. This research received no specific grant from any funding agency.

Availability of data and materials

Not applicable. For any collaboration, please contact the authors.

Declarations

Ethics approval and consent to participate

The author confirms the sole responsibility for this manuscript. The author read and approved the final manuscript

Consent for publication

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

Competing interests

The authors declare that they have no competing interests.

Author details

¹FSTM University Hassan II, Casablanca, Morocco. ²FMPR University Mohammed V, Rabat, Morocco.

Received: 23 August 2021 Accepted: 16 November 2021

Published online: 29 November 2021

References

- Denning DE. An intrusion-detection model. *IEEE Trans Soft Eng.* 1987;SE-13(2):222–32.
- Ramadan RA, Yadav K. A novel hybrid intrusion detection system (IDS) for the detection of internet of things (IoT) network attacks. *Ann Emerg Technol Comput.* 2020. <https://doi.org/10.33166/aetic.2020.05.004>.
- Maha M, Althobaiti K, Mohan KP, Deepak G, Sachin K, Mansour RF. An intelligent cognitive computing based intrusion detection for industrial cyber-physical systems. *Measurement.* 2021;186(110145):0263–2241. <https://doi.org/10.1016/j.measurement.2021.110145>.
- Derhab A, Aldweesh A, Emam AZ, Khan FA. Intrusion detection system for internet of things based on temporal convolution neural network and efficient feature engineering. *Wireless Commun Mobile Comput.* 2020;16:6689134.
- Mulyanto M, Faisal M, Prakosa SW, Leu J-S. Effectiveness of focal loss for minority classification in network intrusion detection systems. *Symmetry.* 2021;13(1):4. <https://doi.org/10.3390/sym13010004>.
- Azmin S, Islam AAABM. A network intrusion detection system based on conditional variational laplace autoEncoder. 7th International Conference on Networking, Systems and Security; 2020. <https://doi.org/10.1145/3428363.3428371>.
- Jie L, Yu ZZ, Wang LH. An intrusion detection method for industrial control systems based on bidirectional simple recurrent unit. *Comput Electrical Eng.* 2021;91(107049):0045–7906. <https://doi.org/10.1016/j.compeleceng.2021.107049>.
- Mahboob AS, Moghaddam MRO. "An Anomaly-based Intrusion Detection System Using Butterfly Optimization Algorithm," 2020 6th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS), 2020; pp. 1–6. <https://doi.org/10.1109/ICSPIS51611.2020.9349537>.
- Sahar N, Mishra R, Kalam S. Deep learning approach-based network intrusion detection system for fog-assisted IoT. In: Tiwari S, Suryani E, Ng AK, Mishra KK, Singh N, eds. *Proceedings of International Conference on Big Data, Machine Learning and their Applications. Lecture Notes in Networks and Systems*, vol 150. 2021; Springer: Singapore. https://doi.org/10.1007/978-981-15-8377-3_4.
- Khan RU, Zhang X, Alazab M, Kumar R. IEEE 2019 Cybersecurity and Cyberforensics Conference (CCC). Melbourne, Australia (2019.5.8–2019.5.9) 2019 Cybersecurity and Cyberforensics Conference (CCC)—an improved convolutional neural network model for intrusion detection in networks. 2019;74–77. <https://doi.org/10.1109/CCC.2019.000-6>.
- Bediya AK, Kumar R. A novel intrusion detection system for internet of things network security. *J Inform Technol Res.* 2021;14:3. <https://doi.org/10.4018/JITR.2021070102>.
- Khan MA. HCRNNIDS: hybrid convolutional recurrent neural network-based network intrusion detection system. *Processes.* 2021; 9(5): 834. <https://doi.org/10.3390/pr9050834>.
- Thakur S, Chakraborty A, De R, Kumar N, Sarkar R. Intrusion detection in cyber-physical systems using a generic and domain specific deep autoencoder model. *Comput Electrical Eng.* 2021;91(107044):0045–7906. <https://doi.org/10.1016/j.compeleceng.2021.107044>.
- Sekhar R, Sasirekha K, Raja PS, et al. A novel GPU based intrusion detection system using deep autoencoder with Fruitfly optimization. *SN Appl Sci.* 2021;3:594. <https://doi.org/10.1007/s42452-021-04579-4>.
- Khonde SR, Ulagamuthalvi V. An hybrid architecture for distributed intrusion detection system using semi-supervised classifiers in ensemble approach. *Adv Model Anal B.* 2020. https://doi.org/10.18280/ama_b.631-403.
- Shen Y, Zheng K, Wu C, Zhang M, Niu X, Yang Y. An ensemble method based on selection using bat algorithm for intrusion detection. *Comput J.* 2018;61(4):526–38.
- Deepa M, Sumitra Dr P. An intrusion detection system using K-means based on cuckoo search optimization. *IOP Conf Series Mater Sci Eng.* 2020. <https://doi.org/10.1088/1757-899x/993/1/012049>.
- Divakar S, Priyadarshini R, Kishore M. "A Robust Intrusion Detection System using Ensemble Machine Learning," 2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE); 2020. pp. 344–347. <https://doi.org/10.1109/WIECON-ECE52138.2020.9397969>.
- Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput.* 1997;9(8):1735–80. <https://doi.org/10.1162/neco.1997.9.8.1735>.
- Carbune V, Gonnet P, Deselaers T, et al. Fast multi-language LSTM-based online handwriting recognition. *IJDAR.* 2020;23:89–102. <https://doi.org/10.1007/s10032-020-00350-4>.
- Ying W, Zhang L, Deng H. Sichuan dialect speech recognition with deep LSTM network. *Front Comput Sci.* 2020;14:378–87. <https://doi.org/10.1007/s11704-018-8030-z>.
- Preethi D, Khare N. EFS-LSTM (Ensemble-Based Feature Selection With LSTM) classifier for Intrusion Detection System. *IJEC.* 16.4. 2020; pp. 72–86. Web. 4 May 2021. <https://doi.org/10.4018/IJEC.2020100106>.
- Vasudevan AB, Dai D, Van Gool L. Talk2Nav: long-range vision-and-language navigation with dual attention and spatial memory. *Int J Comput Vis.* 2021;129:246–66. <https://doi.org/10.1007/s11263-020-01374-3>.
- Bian L, Zhang L, Zhao K, Wang H, Gong S. Image-based scam detection method using an attention capsule network. *IEEE Access.* 2021;9:33654–65. <https://doi.org/10.1109/ACCESS.2021.3059806>.
- Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. *ArXiv.* 1409; 2014
- Leland M, John H, Nathaniel S, Lukas G. UMAP: uniform manifold approximation and projection. *J Open Source Softw.* 2018;3: 861. <https://doi.org/10.21105/joss.00861>.
- Plackett R. Karl Pearson and the Chi-Squared test. *Int Stat Rev Revue Internationale De Statistique.* 1983;51(1):59–72. <https://doi.org/10.2307/1402731>.

28. Cover TM, Thomas JA. Information theory and statistics. In: Elements of information theory, 2nd edn. Wiley; 2005. <https://doi.org/10.1002/047174882X.ch11>.
29. NSL-KDD Dataset. <https://www.unb.ca/cic/datasets/nsl.html>.
30. KDDCUP99 Dataset. <http://kdd.ics.uci.edu/databases/kddcup99/>.
31. Tavallaei M, Bagheri E, Lu W, Ghorbani A. "A detailed analysis of the KDD CUP 99 Data Set," Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA); 2009.
32. Ieracitano C, Adeel A, Morabito FC, Hussain A. A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. *Neurocomputing*. 2020;387:51–62. <https://doi.org/10.1016/j.neucom.2019.11.016>.
33. Su T, Sun H, Zhu J, Wang S, Li Y. BAT: deep learning methods on network intrusion detection using NSL-KDD dataset. *IEEE Access*. 2020;8:29575–85. <https://doi.org/10.1109/ACCESS.2020.2972627>.
34. Choudhary S, Kesswani N. Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 datasets using deep learning in IoT. *Procedia Comput Sci*. 2020;167(2019):1561–73. <https://doi.org/10.1016/j.procs.2020.03.367>.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► [springeropen.com](https://www.springeropen.com)
