Applied Soft Computing 121 (2022) 108768

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A two-stage intrusion detection system with auto-encoder and LSTMs

Earum Mushtaq, Aneela Zameer *, Muhammad Umer, Asima Akber Abbasi

Department of Computer & Information Sciences, Pakistan Institute of Engineering & Applied Sciences (PIEAS), Nilore, Islamabad 45650, Pakistan

ARTICLE INFO

Article history:
Received 9 November 2021
Received in revised form 7 March 2022
Accepted 18 March 2022
Available online 24 March 2022

Keywords:
Long short term memory
Bi-directional long short term memory
Auto-encoder
False alarm rate
Recurrent neural network
Intrusion detection

ABSTRACT

'Curse of dimensionality' and the trade-off between low false alarm rate and high detection rate are the major concerns while designing an efficient intrusion detection system. In this study, we propose a hybrid framework comprising deep auto-encoder (AE) with the long short term memory (LSTM) and the bidirectional long short term memory (Bi-LSTM) for intrusion detection system by obtaining optimal features using AE and then LSTMs for classification into normal and anomaly samples. The performance of the proposed models is evaluated on the well-known dataset NSL-KDD in terms of error indices including precision, recall, F-score, accuracy, detection rate (DR), and false alarm rate (FAR). Experimental results indicate that the proposed AE-LSTM performance is significantly better with less prediction error as compared to other deep and shallow machine learning techniques including other recently reported methods. On the NSL-KDD dataset, AE-LSTM shows classification accuracy of 89% with DR of 89.84% and FAR of 11% which demonstrates the enhanced performance of the proposed model over recent state-of-the-art techniques.

1. Introduction

Digital World—a paradigm of digitization has significantly changed the means of communication and paved the way for innovative developments in technology and certain challenges [1]. This trend of dependence on internet-oriented services in all aspects of life including business, education, e-commerce and military affairs network security is an important concern in case of intrusion to protect the assets of sensitive organizations and individuals [2]. Recent security breaches in Aadhaar, British Airways, Exactis, Under Armour, Yahoo, etc. have inspired professionals from academia and industry towards cybersecurity as an exciting research area to protect critical information from devastating cyber-attacks. Conventionally, 'the first line of defense' such as 'antivirus, access control, encryption, decryption, firewalls', etc. are employed for intrusion detection and safeguard the network from hazardous cyber-attacks [3]. However, these conventional security techniques have the inadequate ability and are occasionally unable to protect the network from novel intrusion procedures [4]. The aforementioned challenge highlights the obligation to use defense-in-depth approaches [5]. Therefore, an efficient intrusion detection system (IDS) has been proposed by researchers.

An IDS monitors a network for malicious activities and offers protection from numerous prevailing and novel cyber-attacks that affect the confidentiality, integrity and availability (CIA) of the network. An IDS comprises several types and configurations which cover records information, notifies security administrators of unusual activity and, generate reports [6]. IDS software can be installed in several ways based on source and type of data analyzed such as Host-based IDS (HIDS), Network-based IDS (NIDS) and distributed or hybrid IDS. IDS software detects threats by applying either a single approach or using the approaches that are misuse or signature-based and anomaly-based [7]. Signature-based is used to recognize known threats whereas anomaly-based is used to recognize new threats. In anomaly-based rather than looking at a static database, it inspects standard activities and preserves a log that keeps an idea of what normal activities for a traditional system look like and generates an alert for the activity that deviates from normal data flow pattern [8]. Anomaly-based has an advantage in that it classifies zero-day attacks which are neglected and not identified in the signature-based but its disadvantage is that it generates more false-positive than signature-based [9]. A variety of classification methods are used for anomaly-based, which are broadly divided as statistical-based, knowledge-based, and machine learning (ML) based [10]. Machine learning-based approaches apply data mining methods to automatically generate a model using the labeled normal training data. A key limitation is that this requires time and computational resources but once the model is developed subsequent analysis is usually efficient [11].

An efficient IDS can deal with high dimensional data for fast and real-time decisions while maintaining a high detection rate and less false alarm rate [12]. High dimensional data, imbalanced

* Corresponding author.
E-mail addresses: Earum.mushtaq@gmail.com (E. Mushtaq), aneelaz@pieas.edu.pk (A. Zameer), malikumer00000@gmail.com (M. Umer), asima_akber@yahoo.com (A.A. Abbasi).

https://doi.org/10.1016/j.asoc.2022.108768
1568-4946/

Abbreviation

AE	Auto-encoder
LSTM	Long short term memory
Bi-LSTM	Bidirectional long short term memory
IDS	Intrusion detection system
CIA	Confidentiality, integrity and availability
HIDS	Host-based IDS
NIDS	Network-based IDS
SVM	Support Vector Machine
K-NN	k-Nearest Neighbor
ANN	Artificial neural network
SOM	Self-Organizing Map
DT	Decision tree
DL	Deep learning
MLP	Multilayer perceptron
CNN	Convolutional neural network
MCNN	Multiscale CNN
PCA	Principal component analysis
RNN	Recurrent neural network
σ	Sigmoid
SND	Standard normal distribution
TCP/IP	Transfer control protocol/internet protocol
DR	Detection rate
FAR	False alarm rate
ACC	Accuracy
ROC	Receiver operating characteristic
PR	Precision recall
GNB	Gaussian naïve Bayes
RBF	Radial basis function
BGRU	Bidirectional gated recurrent unit

datasets, ever-progressing nature of cyber-attack make the design of robust and efficient IDS a thought-provoking task [13]. The intrusion detection is considered as a classification problem; that can classify the network traffic as malicious or normal by models and rules has prompted the researchers to apply and incorporate artificial intelligence methods like support vector machine (SVM), k-Nearest Neighbor (k-NN), artificial neural network (ANN), naïve Bayes network, self-organizing map (SOM), decision tree (DT) and many others to enhance its performance [14–16]. Conventional ML methods are computationally inefficient and not able to learn complex and non-linear relationships of network traffic data [17]. However, deep learning (DL) techniques can automatically extract high-level abstraction from input through learning in a hierarchical way. DL has accomplished remarkable results in several research areas such as bioengineering [18,19], sentiment analysis [20,21], image recognition [22,23] and saliency detection [24]. Presently, it has also been in use for development of efficient and reliable intrusion detection systems.

Various machine learning and deep learning methods exploiting conventional feature selection techniques have been applied to identify malicious network traffic and detected unknown network attacks [25]. An intrusion detection method has been proposed by Zhou et al. which is built by selecting optimal features using CFS-BA technique and ensemble classifier. In their suggested method most relevant features are selected based on correlation among features and then ensemble classifier comprising C4.5, random forest (RF) and forest by penalizing attributes (Forest PA) is proposed, afterwards voting technique is employed for final classification. NSL-KDD dataset is used for simulation and their method has achieved an accuracy of 87.37% with DR 87.4% [26]. A deep learning approach using wrapper based feature extraction has been suggested by Kasongo and Sun, deep feed forward neural network is proposed whereas relevant features are extracted through Extra tree (ET) classifier and obtained accuracy of 85.48% on UNSW-NB15 test set. However, their method is computationally inefficient [27]. Sara A. Althubiti applied LSTM with rmsprop for network intrusions, they obtained an accuracy of 85.5% and assessed their model on the CIDDS-001 dataset also revealed that LSTM performed better than naïve Bayes, MLP (multilayer perceptron) and SVM techniques [28]. Boukhalfa *et al.* also proposed a method based on LSTM for binary and multiclass classification, the proposed method has a greater ability to memorize and discriminate in normal and malicious traffic [29]. Meng et al. proposed an innovative model for the detection of network attacks, their method is based on the kernel principal component analysis (PCA) and LSTM [30]. Yang et al. proposed a Bi-LSTM for efficient detection of known and unknown malicious network attacks by using the UNSW-NB15 dataset, the proposed method achieved high accuracy and recall rate in the detection of normal or attack traffic [31].

Deep learning with emerging feature selection techniques has recently been reported by various researchers in the domain of cybersecurity. Yan et al. introduced a network intrusion detection method based on auto-encoder (AE) and LSTM, UNSW-NB15 dataset is used for simulation and achieved an accuracy of 92% but its disadvantage is its high FAR [32]. Ahsan and Nygard proposed a novel hybrid algorithm for attack type classification based on CNN (convolutional neural network) and LSTM to detect network intrusion with high accuracy and low FAR [33]. Hsu et al. proposed a model using LSTM [34], they compared its performance with the LSTM-only, LSTM with CNN and BGRU+MLP proposed by Xu et al. [35], CNN-LSTM method achieved an accuracy that is better as compared to LSTM-only. Zhang et al. proposed a method for network intrusion detection by using multiscale CNN (MCNN) with LSTM [2]. The aforementioned methods which uses a full feature set for training the classifier requires sophisticated computational resources. AE has gained enough attention from professionals in the last decade. First time Hawskin et al. proposed a method for outlier detection and now that is common in use [36]. Recently use of AE for dimension reduction is very popular because of its good performance to produce abstract feature space of high dimensional data. It has considerably improved the accuracy in comparison to the kernel and linear principal component analysis (PCA) by identifying anomalies that linear PCA is unable to identify. Additionally, training of AE is easy and it is computationally efficient in comparison to the complex computation of PCA. Based on these inspiring results on AE for dimensionality reduction and LSTMs on intrusion detection, we propose a novel auto-encoder long short term memory (AE-LSTM) based scheme. Salient features of the proposed technique are highlighted in the following points:

1. An anomaly detection model is proposed by exploiting AE for optimal feature selection through eliminating the noisy and less informative features for binary classification with high detection rate and low false alarm rate.
2. The LSTM model is proposed with comparatively less parameters and number of layers for final classification using encoded features provided by AE.
3. Statistical analysis of the proposed model have been performed to ensure its stability and efficiency.

The remaining part of the paper is arranged as; Section 2 comprises relevant theory and details of the proposed methodology; in Section 3, experimental results and comparison with existing techniques are discussed; finally, Section 4 concludes the paper.

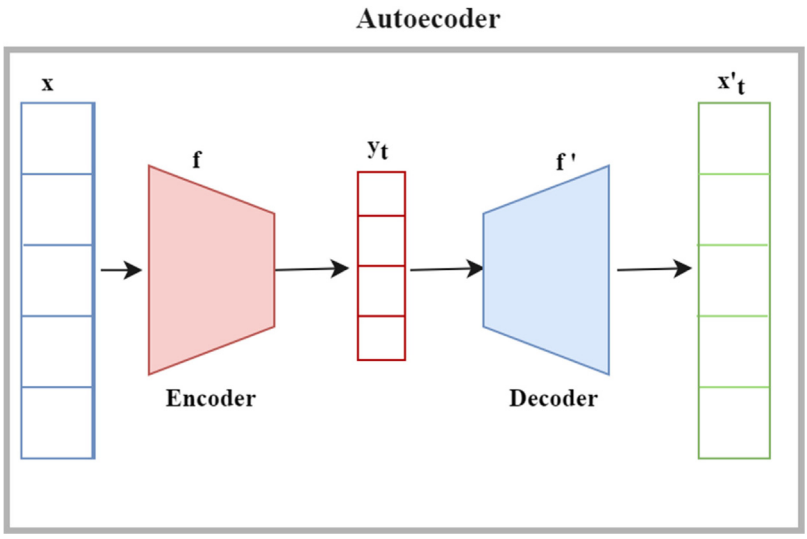


Fig. 1. An illustration of AE network.

2. Materials and methods

The proposed AE-LSTM comprises AE and LSTM to develop a novel approach with optimal feature selection. An overview of AE and LSTMs is represented in the following subsections. Main features of the dataset and performance indices are presented in the end.

2.1. Auto-encoder

Auto-encoder is a deep neural network that is employed to learn the best encoding–decoding representation of data [37]. Architecture of AE comprises an input layer, an encoder, latent space, decoder and an output layer as shown in Fig. 1. Data is passed to an input layer and the encoder module encodes the data into latent dimension by compressing it. Afterwards, decoder decodes the compressed encoding and reconstructs original representation at output layer. The objective of AE is not to replicate the input at output layer, also it reduces the dimensions of data while keeping the relevant information intact. AE finds the data representation in lower dimension, more precisely, it eliminates the noisy and less informative features that are not contributing towards classification.

The encoder module transforms the input x_t into latent dimensions y_t (hidden representation). Usually, it is a mapping function that can be written mathematically as:

$$y_t = f(\mathrm{W}\mathrm{x}_\mathrm{t} + \mathrm{b}) \tag{1}$$

Here, W is the weight between input x_t and latent representation y_t, b is the bias. Decoder reconstruct $\widehat{\mathrm{x}_\mathrm{t}}$ from the hidden representation y_t which can be expressed as:

$$\hat{x}_t = f'(\mathrm{W}'\mathrm{y}_\mathrm{t} + \mathrm{b}') \tag{2}$$

Where, W is weight between y_t the reconstructed output $\widehat{\mathrm{x}_\mathrm{t}}$. The reconstructed output is then matched with original data and error is back-propagated through the network to update the weights. The aim of training of auto-encoder is to minimize the reconstruction error that can be written as a cost function J below

$$\mathrm{J} = \frac{1}{\mathrm{p}} \sum_{\mathrm{i}=1}^{\mathrm{p}} \mathrm{L}[\mathrm{x}_\mathrm{t}, \widehat{\mathrm{x}_\mathrm{t}}] \tag{3}$$

Where p represents input signal, x_t is input signal at tth interval and $\widehat{\mathrm{x}_\mathrm{t}}$ is the reconstructed output. Here $\mathrm{L}[\mathrm{x}_\mathrm{t}, \widehat{\mathrm{x}_\mathrm{t}}]$ represents the reconstruction error that can be calculated by cross entropy or mean square error. In this work, we have used mean square error. Alternatively $\mathrm{L} = [\mathrm{x}_\mathrm{t}, \widehat{\mathrm{x}_\mathrm{t}}]$ can be written

$$\mathrm{L}[\mathrm{x}_\mathrm{t}, \widehat{\mathrm{x}_\mathrm{t}}] = \|\mathrm{x}_\mathrm{t} - \widehat{\mathrm{x}_\mathrm{t}}\|^2 \tag{4}$$

Multiple types of AE have been suggested by researchers such as Vanilla AE, convolutional AE and LSTM-AE. LSTM-AE is a type of auto-encoder that uses LSTM layers for both encoding and decoding. The capability of LSTM to learn long sequences in data makes them appropriate for anomaly detection and time series forecasting.

2.2. LSTMs

Recurrent neural networks (RNN) is a type of deep neural network with loops in its internal memory are specifically used to handle sequential data. The RNNs architecture is graphically presented in Fig. 2. For hidden layer of RNN, it receives input and produces the output vector Y. Working of RNN is shown in the right side of Fig. 2 for every time step t, the hidden layer preserves a hidden state, and update it by using current layer input and prior value of hidden state. x_1 input is passed to the network and hidden state h_1 is computed after that y_1 is its output, similarly for next time step h_2 is computed by using prior hidden state h_1 and input x_2 that produces output y_2. Mathematically it can be written as

$$h^t = f(\mathrm{h}^{(\mathrm{t}-1)}, \mathrm{x}^{(\mathrm{t})}; \theta) \tag{5}$$

Here $h^{(t)}$ is current hidden state that is equal to the some function f of the prior hidden state $h^{(t-1)}$ and current input $x^{(t)}$ whereas θ is parameter of function f [38].

Parameters of RNN are optimized by using back-propagation (BP). RNNs have great capability to handle nonlinear time series problems [39], but regular RNNs have the problem of vanishing and exploding gradients during BP thus unable to learn long time dependencies [40]. To overcome this problem, LSTM was suggested as it has the capability to remember values for time intervals, short or long [41]. Architecture of LSTM is graphically shown in Fig. 3.

LSTM has three gates: forget gate, input gate and output gate. Every gate is dependent upon state of preceding time step and current input signal, whereas sigmoid layer and multiplication operation is used in gates. The sigmoid layer produces the output between 1 and 0 that shows how much information can be

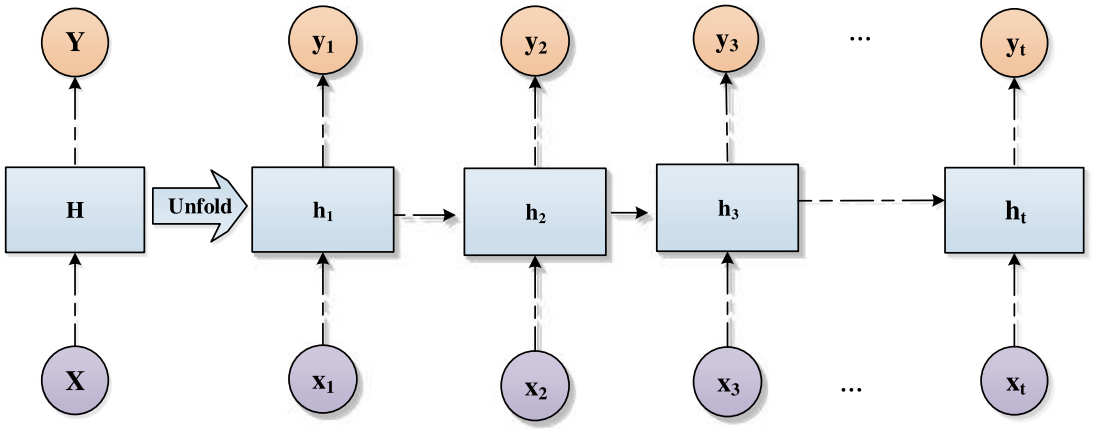


Fig. 2. RNN architecture.

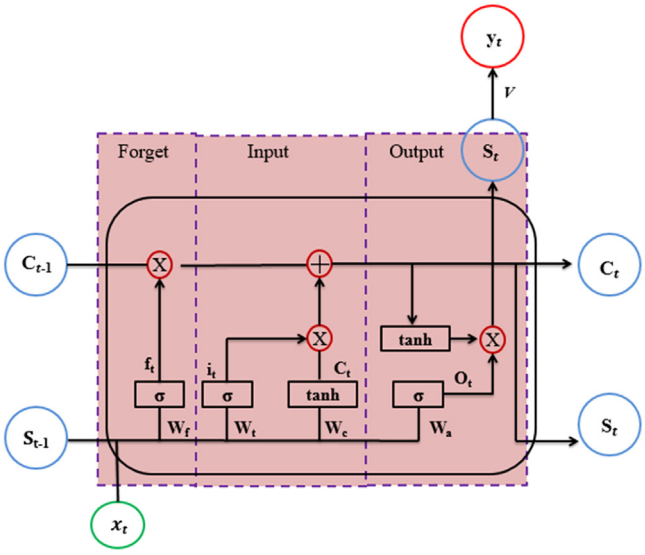


Fig. 3. Architecture of LSTM.

accepted. *Input gate* is responsible for selecting what information is required to store and *forget gate* decides what information need to be remove from unit state C and *output gate* specifies the new states [42]. Following equations represents the internal working of LSTM unit.

$$f_t = \sigma(W_f.[s_{t-1}, x_t] + b_f), \tag{6}$$

$$i_t = \sigma(W_i.[s_{t-1}, x_t] + b_i) \tag{7}$$

$$o_t = \sigma(W_o.[s_{t-1}, x_o] + b_o) \tag{8}$$

$$C_t^{\sim} = \tanh(W_c.[s_{t-1}, x_o] + b_c), \tag{9}$$

$$C_t = f_t.C_{t-1} + i_t.C_t^{\sim}, \tag{10}$$

$$s_t = o_t.\tanh(C_t), \tag{11}$$

Where, C_{t-1}, C_t and $C_t^{\sim}$ represents unit memory and W_f, W_i, W_c, and W_o are weight matrices then b_f, b_i, b_c and b_o are bias vectors. LSTM has certain limitations as it cannot predict well on future contents, to solve this problem of LSTM Schuster and Paliwal suggested bidirectional LSTM that consist of two LSTM hidden layers having same output but in reverse directions [43]. Fig. 4 represents the working of Bi-LSTM.

This architecture of LSTM can predict well for future predictions. By using Bi-LSTM an input sequence $X = (X_1, X_2, \ldots..X_n)$ is calculated in forward direction as $\overrightarrow{h_t} = (\overrightarrow{h_1}, \overrightarrow{h_2} \ldots \overrightarrow{h_n})$ and in backward direction by $\overleftarrow{h_t} = (\overleftarrow{h_1}, \overleftarrow{h_2} \ldots \overleftarrow{h_n})$ using these equations. Final result of this cell is y_t that is combination of both direction outputs. Final result of this cell is y_t, the combination of both direction output. Mathematically, it can be written as:

$$y_t = \partial(\overrightarrow{h_t}, \overleftarrow{h_t}) \tag{12}$$

Here ∂ is function that is used to merge these two outputs, it can be concatenation function, sum function, average function or multiplication. Output sequence will be like this $y = (y_1, y_2, \ldots..y_n)$ [42].

2.3. The proposed framework: AE-LSTM

We propose a novel anomaly detection framework, AE-LSTM exploiting deep auto-encoder and long short term memory for intrusion detection system. The graphical representation of the proposed methodology is shown in Fig. 5 whereas details are discussed in following subsections.

One hot encoding

The dataset has 41 features, among them three are categorical features; *protocol type, service and flag*. Most of machine learning models work on numeric data therefore we applied one hot encoding to convert categorical features into numeric values. This technique counts the distinct values for each feature and unique index is assigned for each value [44].

***Data Normalization*:** Data normalization process normalizes the feature values in the range of [−1, +1] or [0, +1] that depends upon the ML model applied. Data normalization also known as standardization that helps to lessens the training time and faster convergence of model. To achieve data normalization multiple techniques exist like min–max scaling, standard scaling and averaging. In this study, we have applied standard scalar to standardize the data. Standard scalar used standard normal distribution (SND) thus its mean is 0 and variance is 1. After applying one hot encoding we have 119 features, we have applied standard scalar to standardize the feature space. Mathematically it can be written as:

$$z = \frac{x - \mu}{sd} \tag{13}$$

Here z is the standardized feature space of input data samples x, μ is mean and sd is standard deviation. Mathematical representation of mean is $\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i)$ and standard deviation is $sd = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$ here x_i is input sample. Standardization

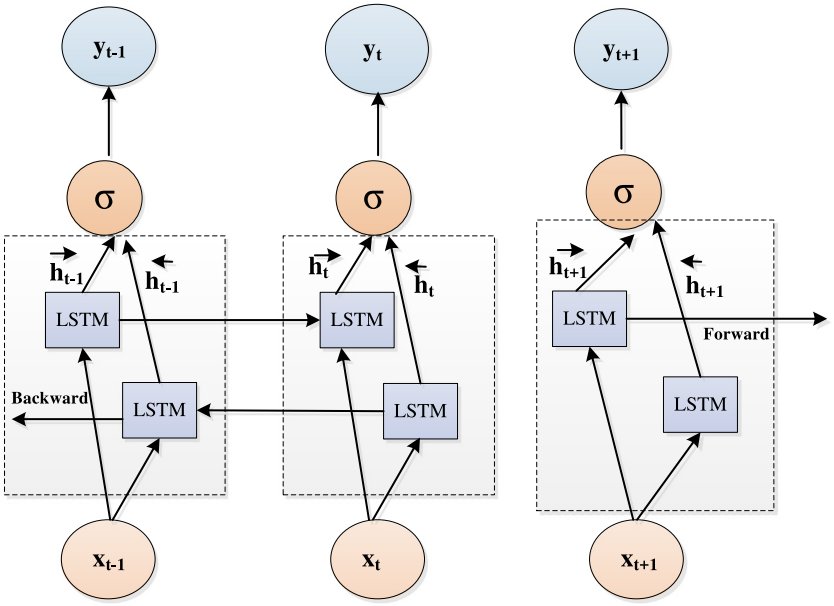


Fig. 4. Architecture of Bi-LSTM.

preserves information of data outliers and helps the model to become insensitive from them as compared to min–max scalar.

Classification: After conversion of categorical features into numeric features we have total of 118 features that needs to be reduced in order to lessen training time. In the proposed AE-LSTM, auto-encoder is used for extracting the robust and optimal feature subset. Thirty features are extracted from bottleneck layer of auto-encoder and fed into the LSTM model for the final classification into normal or anomaly as graphically elaborated in Fig. 6.

Although different deep learning techniques have been reported in the past for intrusion detection, most of them have employed full feature set. A deep learning technique, RNN-IDS, has been proposed by Yin et al. They used recurrent neural networks to classify network traffic into normal and malicious as well as attack type classification by using all the features of NSL-KDD dataset. Their suggested method achieved an accuracy of 81.29% on binary classification [45]. An intrusion detection method has been proposed by Qureshi et al. named as 'DST-TL'. They used deep learning approach of self-taught learning to classify network traffic and achieved an accuracy of 84.60% on KDDTest $^{+}$ variant of NSL-KDD dataset [46]. There are some shortcomings of traditional ML techniques for classification, for example feature selection is unable to select most distinguishing features in big data and thus effects the model's performance [47]. Since, in case of intrusion detection, it has to deal with massive network traffic data, in such scenarios traditional ML techniques of feature engineering and classification cannot perform well. Thus, the objective of the proposed framework using AE is to reduce the dimensionality and provide robust and optimal feature space to LSTM for good separability between two classes. Particularly it encodes the data into a low dimensional space in a way that class separability between two classes is high and dispersion of samples that belongs to same class is low.

2.4. Dataset

Since 1999, KDD'99 dataset has been frequently used to evaluate various proposed IDS. Stolfo et al. [48] prepared this dataset and is built by gathering data in DARPA'98 IDS assessment contest [49]. DARPA'98 comprises network traffic data of 7 weeks having 5 million records and two weeks of test data that is around 2 million records with 41 features and is labeled either normal or attack type. KDD'99 is based on DARPA'98 which is critically analyzed by McHugh [50], primarily due to synthetic data. It contains redundant samples and during training of classifiers they may become biased towards redundant samples and effects the model's generalization [46]. Later Tavallaee et al. proposed the refined version of KDD'99 named as NSL-KDD, where these deficiencies are eliminated by removing redundant records and difficulty level is also allotted for each sample. Dataset is divided into KDDTrain $^{+}$, 20%KDDTrain $^{+}$ and KDDTest $^{+}$. Features of NSL-KDD are described in Table 1.

Well-known intrusion detection dataset, NSL-KDD has been used to evaluate the proposed methodology in this study. We have used KDDTrain $^{+}$_20% for training and KDDTest $^{+}$ for testing of our proposed model. Distribution of normal and attack samples is graphically illustrated in

Fig. 7. The NSL-KDD dataset has four category of features as graphically illustrated in Fig. 8; (1) *Basic features* without seeing TCP/IP payload features are extracted comes under the category of basic features. 10 features are basic features in NSL-KDD dataset. (2) *Content features:* these are relevant to suspicious information of payload inside TCP packet, 12 features are content related features. (3) *Time based traffic features:* this type of features are extracted by setting the time of two seconds, 9 features are time based traffic features. (4) *Connection based features:* features extracted that have time greater than two seconds are known as connection based features, despite of time slice, "same host " and "same service" features are extracted by reviewing the history of previous connections (100 connections) [46–51].

2.5. Performance indices

Performance indices such as accuracy, precision, recall, F-score, detection rate, false alarm rate and area under the precision recall curve are used to evaluate the performance of proposed framework. All the outcomes are categorized into following four

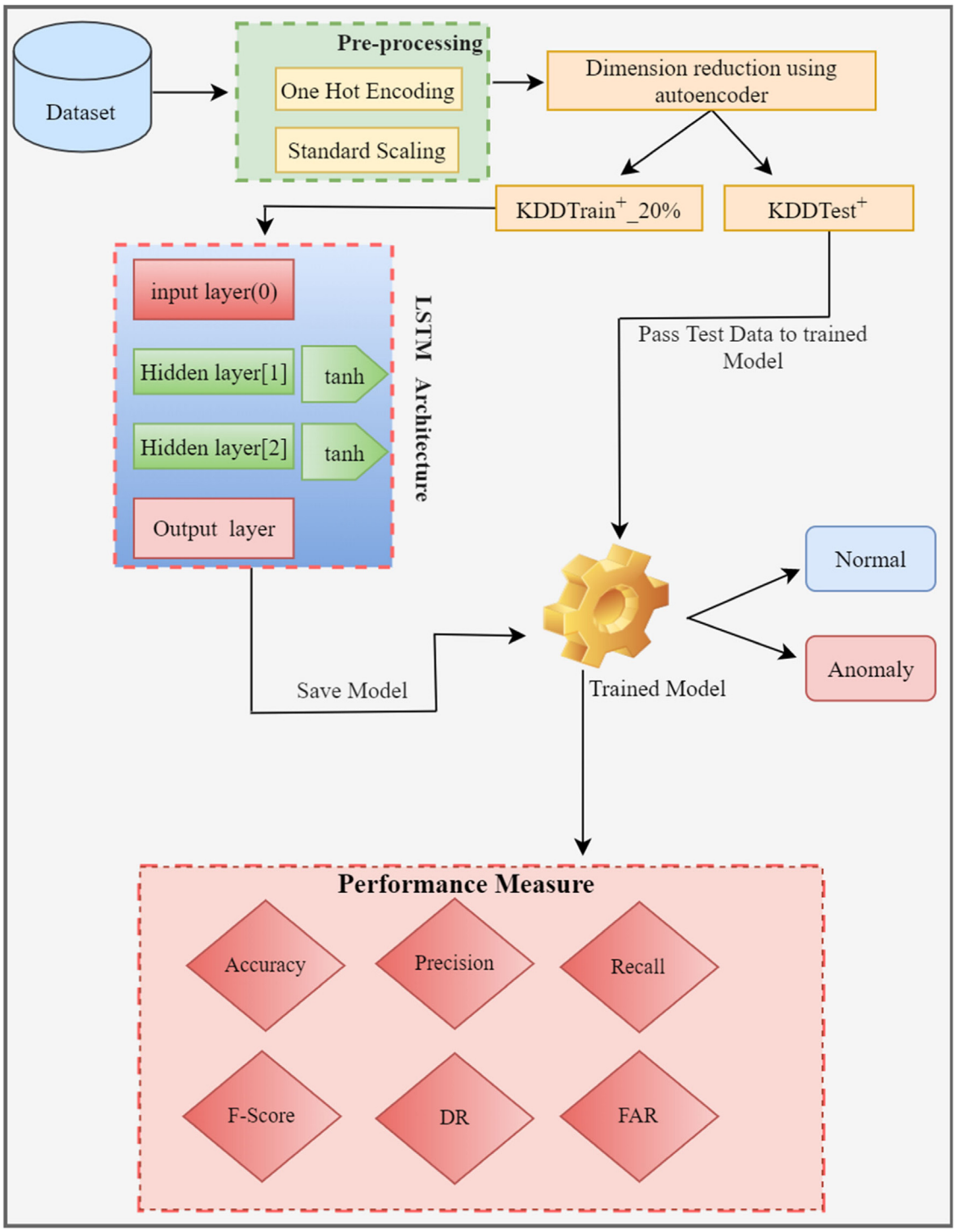


Fig. 5. Graphical abstract of the proposed AE-LSTM.

performance variables; **TP:** Correctly predicted anomaly samples, **TN:** Correctly predicted normal examples, **FP:** Incorrectly predicted anomaly samples, **FN:** Incorrectly predicted normal samples. Mathematical representation of all the performance indices are listed in Table 2.

The precision estimates the true classifications penalized by false classifications, recall estimates the total correctly classified examples, F-score is the harmonic mean of precision and recall, accuracy is the proportion of correct classified examples to all examples of dataset, detection rate depicts the number of samples truly identified as malware from dataset and false alarm rate is percentage of incorrect classifications divided by total normal samples. All the experiments have been carried out on Google Colab.

3. Experimental results and discussion

The parameter setting of LSTMs is listed in Table 3. Optimal hyper parameters are selected using trial and error method as well as keras tuner is employed.

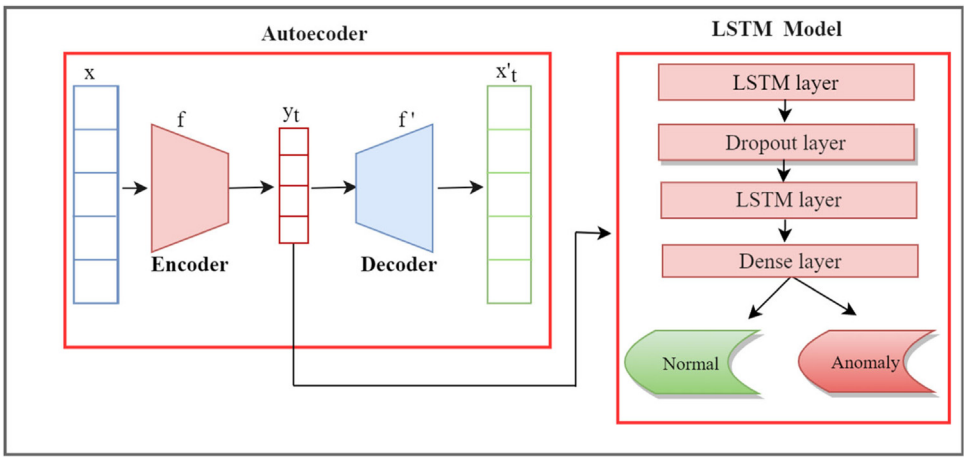


Fig. 6. The work flow of AE-LSTM scheme.

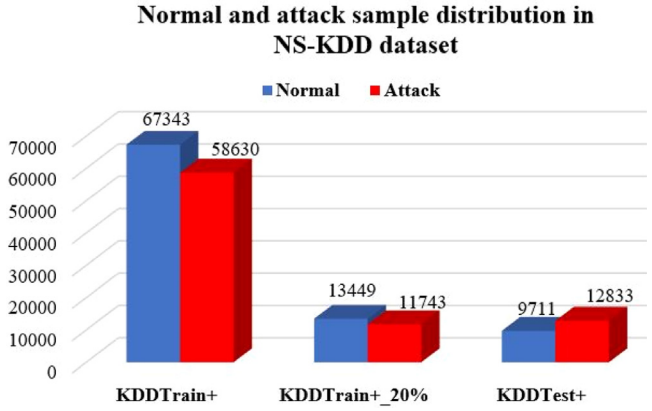


Fig. 7. Normal and anomaly samples distribution in NSL-KDD dataset.

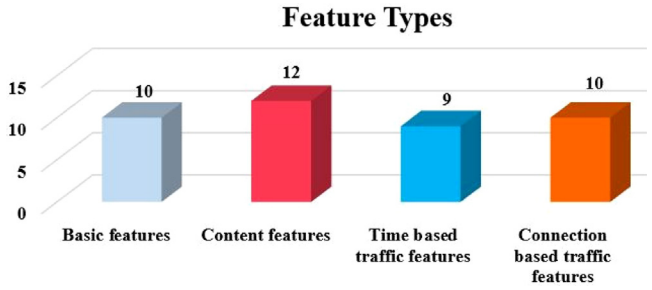


Fig. 8. Types of features in NSL-KDD dataset {Marin, 2005 #3}.

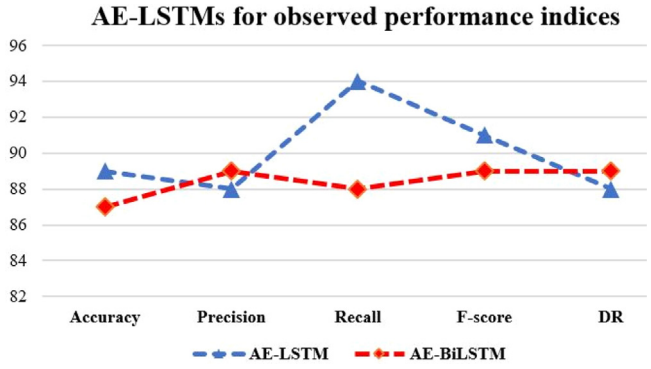


Fig. 9. Types of features in NSL-KDD dataset.

3.1. AE-LSTMs for observed performance indices

The proposed AE-LSTMs performance for several performance indices have been examined and Fig. 9 shows graphically. Fig. 9 depicts that AE-LSTM performs better in terms of accuracy 89%, recall 0.94 and F-score 0.91 whereas AE-BiLSTM has less accuracy 87% with less recall 0.88 and F-score 0.89.

3.1.1. AE-LSTM model accuracy and loss plots

The plot of accuracy and loss of the proposed model is plotted in Fig. 10. We can see in accuracy plot that the models are well trained as curves are now not rising after certain point. We can also see that the model has not yet over-learned the training dataset, showing comparable skill on both datasets. Loss plot shows that model have comparable performance on both train and test datasets and loss is decreasing when we move to higher number of epochs. If these parallel curves start to depart this is an indicator to stop training.

3.1.2. Multiple independent execution of the proposed AE-LSTMs

For more detailed analysis of the proposed AE-LSTMs ten independent executions have been carried out and results are plotted in the form of box plots as shown in Fig. 11. The top and bottom values are shown at upper and lower positions of box-plots whereas the median is shown by line inside the boxplots. Furthermore, it is evident from these boxplots that the proposed AE-LSTM has high accuracy values as compared to LSTM without AE and AE-BiLSTM. For detection rate LSTM without AE has 92% DR and the proposed AE-LSTM has 90.5%, which is slightly higher than AE-LSTM. Since it is a class imbalance problem, therefore, F-score is the standard measure to assess the performance of the model. F-score observed for AE-LSTM (0.91) is higher than that of LSTM only (0.79).

3.1.3. ROC and PR curve analysis of the AE-LSTM

ROC curve measures the performance of model at different threshold values by plotting false positive rate on *x*-axis and true positive rate on y-axis [52]. Fig. 12 illustrates the ROC and PR curves of proposed models. From Fig. 12, it can be seen that a), b) are the ROC and PR of the proposed model without AE and c), d) shows ROC and PR curves with AE. For ROC we can observe LSTM is showing AUC (area under curve) of 88.86 but with AE it is considerably improved for proposed model with AUC of 93.13. Similarly for PR curve the proposed model PR-AUC without AE is

Table 1
Description of NSL-KDD features.

Attribute #	Attribute name	Description
1	Duration	Time span of the connection
2	Protocol_type	Protocol that is used in the connection
3	Service	Service used by destination network
4	Flag	State of the connection
5	Src_bytes	Data bytes transmitted from source
6	Dst_bytes	Data bytes transmitted from destination to source
7	Land	This attribute set to 1 if source and destination IP'S are equal otherwise 0
8	Wrong_fragment	Number of wrong fragments in connection
9	Urgent	Urgent packets in connection (urgent packets have urgent bit activated).
10	Hot	Hot pointers for example directory and incoming of contents to system.
11	Num_failed_logins	Sum of unsuccessful login tries
12	Logged_in	Login Status : 1 if logged in else 0
13	Num_compromised	Total compromised situations
14	Root_shell	It is 1 for root shell acquired otherwise 0
15	Su_sttempted	1 for "su root" tried else 0
16	Num_root	Count of "root" accesses
17	Num_file_creations	count of file creation actions
18	Num_shells	Numeral of shell prompt
19	Num_access_files	Total operations on access control files
20	Num_outbound_cmds	Sum of outbound commands
21	Is-hot_login	It is 1 if host login else 0
22	Is_guest_login	It is 1 if guest login else 0
23	Count	Total connections to same destination host as the recent connection of past two seconds
24	Srv-count	Total connections to the same service as the recent connection of past two seconds
25	Serror_rate	The proportion of connection that triggered the flag in count (23)
26	Srv_rerror_rate	The proportion of connections that triggered the flag in srv_count (24)
27	Rerror_rate	The proportion of connections that triggered the flag (4) REJ, between the connections accumulated in count (23)
28	Srv_rerror_rate	The fraction of connections that triggered the flag (4) REJ, between the connections accumulated in srv_count (24)
29	Same_srv_rate	The fraction of connections that were to the same service, between the connections gathered in count (23)
30	Diff_srv_rate	The fraction of connections that were to different services, between the connections accumulated in count(23)
31	Srv_diff_host_rate	The fraction of connections that were to different destinations amongst the connections gathered in srv_count (24)
32	Dst_host_count	Total connections having similar destination and host IP's
33	Dst_host_srv_count	Total connections having similar port number
34	Dst_same_srv_rate	The fraction of connections that are using the same service amid the connections accumulated in dst_host_count (32)
35	Dst_host_diff_srv_rate	The fraction of connections that are using different service amid the connections accumulated in dst_host_count (32)
36	Dst_host_same_src_port_rate	The proportion of connections that use the same source port between the connections aggregated in dst_host_srv_count (33)
37	Dst_host_serve_diff_host_rate	The proportion of connections that use different destination machines between the connections accumulated in dst_host_srv_count(33)
38	Dst_host_serror_rate	The proportion of connections that triggered the flag (4) among the connections accumulated in dst_host_count (32)
39	Dst_host_srv_serror_rate	The proportion of connections that triggered the flag (4) among the connections gathered in dst_host_srv_count (33)
40	Dst_host_rerror_rate	The proportion of connections that triggered the flag (4) REJ, among connections in dst_host_count (32)
41	Dst_host_rerror_rate	The proportion of connections that triggered the flag (4) REJ, among the connections in dst_host_srv_count (33)
42	label	Label of sample either normal or anomaly

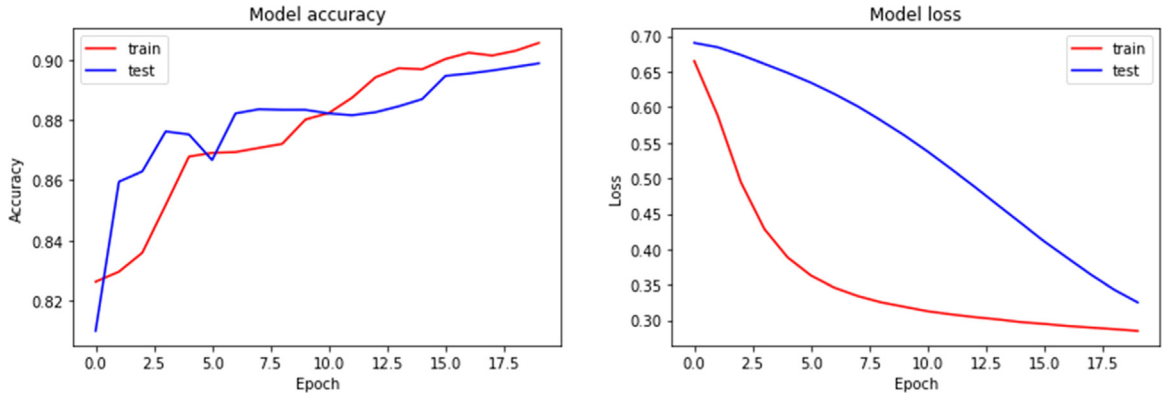
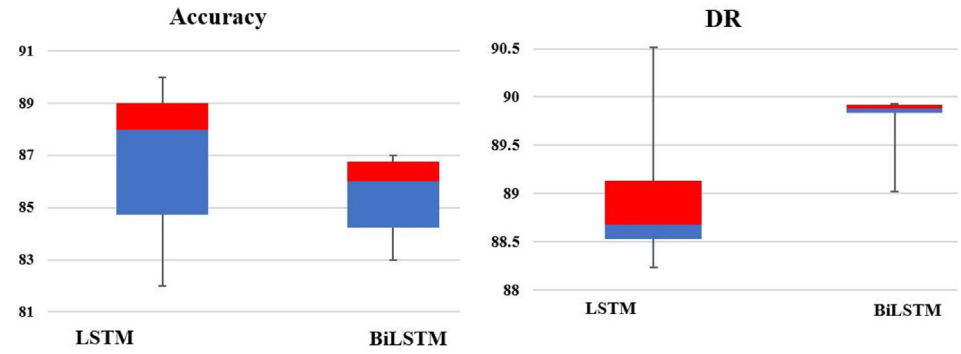
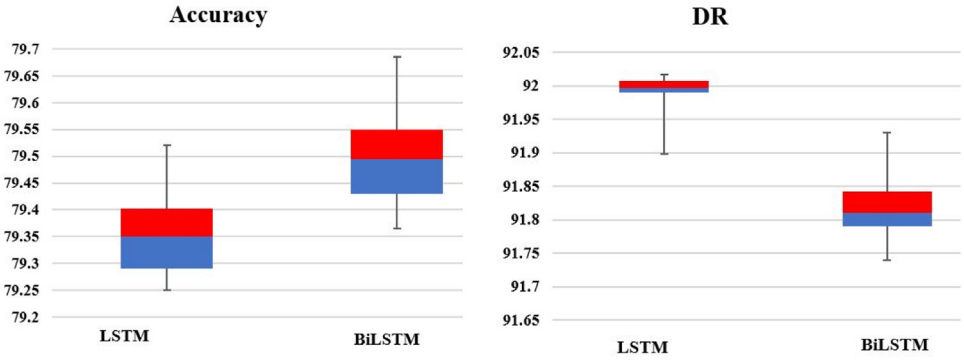


Fig. 10. Accuracy and loss plots for train and test data.



a) Accuracy and DR of proposed models with AE for ten independent runs



b) Accuracy and DR of proposed models without AE for ten independent runs

Fig. 11. Boxplots for accuracy and detection rate on multiple independent runs.

Table 2
Performance indices and their mathematical equations.

Metric	Symbol	Mathematical formulae
Precision	P	$\Pr ecison = \frac{TP}{TP+FP}$
Recall	R	$Recall = \frac{TP}{TP+FN}$
F-Score	F	$F-score = 2 \times \frac{precision \times recall}{precision+recall}$
Accuracy	ACC	$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100$
Detection rate	DR	$DetectionRate = \frac{TP}{TP+FP}$
False alarm rate	FAR	$FalseAlarmRate = \frac{FP}{TN+FP}$

Table 3
Parameter setting of LSTMs.

Parameters	Settings
No of layers	{1, 2}
No of neurons	{1024, 512}
Epochs	20
Optimizer	Adam
Batch size	850
Activation function	Softmax
Learning rate	0.0001

91.27 but with AE it has improved up to 93.99. Since our dataset is imbalanced and for this type of dataset PR curve is standard measure to check the performance of proposed model. It can further be observed that the proposed AE-LSTM has significant PR-AUC of 93.99 that indicates good capability of classifier.

3.1.4. Hyper parameter optimization of the proposed model

Multiple parameter settings of proposed models have been carried out to get the best results. We have optimized the number of epochs, number of layers, learning rate and different optimizers. Detail results are listed in Table 4 and Fig. 13. Experiments have been conducted by keeping constant Adam optimizer and learning rate of 0.0001 with varying number of layers. Experimentally it has been observed that AE-LSTM with only 2 layers showed best results for all performance indicators such as 89.0% accuracy, 0.88 precision, recall of 0.94 and F-score 0.91. After that when number of layers were increased performance of the model starts decreasing, up to 6 layers results were observed and results started gradually decreasing, by adding 6 layers accuracy dropped to 82% and F-score to 0.83. Similarly, with rmsprop optimizer same experiments were performed and best results are obtained with 3 layers of LSTM but that are low in comparison to Adam optimizer. Experiments with SGD are also performed and it has been observed that Adam is performing best for our proposed model. Effect of learning rate has also studied in this work and different learning rates are employed with optimizers and learning rate 0.0001 is performing most efficiently, with Adam optimizer it has achieved 89% accuracy and with rmsprop 84% .

3.1.5. AE-LSTM evaluation on full feature set

To check the effectiveness of the AE-LSTM comparison have been carried out with full feature set and results are presented in Table 5 and graphically in form of bar charts in Fig. 14. From Table 5 it can be seen that GNB has lowest performance for all performance indices such as 56% accuracy, 0.33 precision, recall 0.49 and F-score 0.36. However, SVC with rbf kernel is performing best with accuracy of 81%, precision of 0.82, recall of 0.82, F-score 0.81 and DR of 0.82. The proposed LSTM on full feature set is showing accuracy of 79% and F-score 0.79 that is lower than SVC (rbf). For precision and DR, the proposed LSTM is showing good performance with precision of 0.92 however, for imbalance

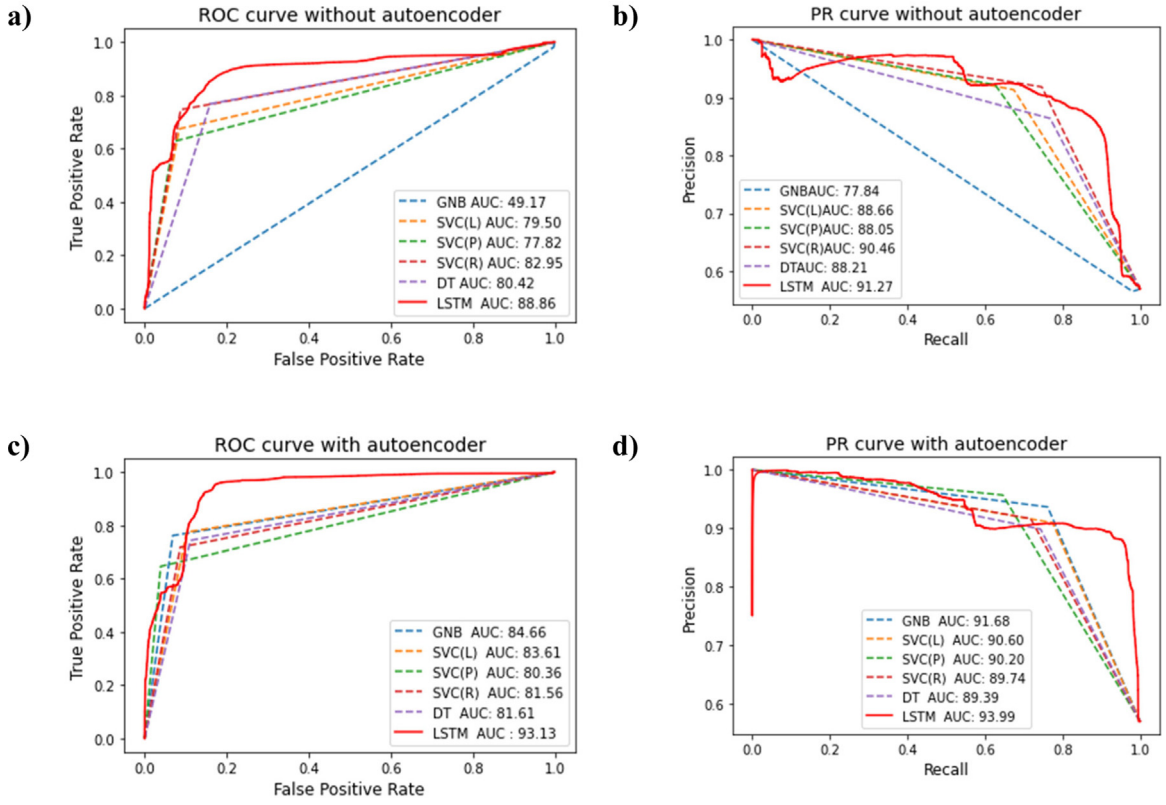


Fig. 12. **(a)** ROC-AUC without AE **(b)** PR-AUC without AE, **(c)** ROC-AUC with AE **(d)** PR-AUC with AE.

Table 4
Parameter setting for variants of LSTM.

Model	Optimizer	No of layers	Learning rate	Accuracy	Precision	Recall	F-score
AE-LSTM	Adam	1	0.0001	78.79	0.90	0.70	0.70
		2	**0.0001**	**89.0**	**0.88**	**0.94**	**0.91**
		3	0.0001	88.68	0.89	0.90	0.90
		4	0.0001	82.69	0.90	0.77	0.83
		5	0.0001	80.66	0.91	0.73	0.81
		6	0.0001	82	0.92	0.71	0.83
AE-LSTM	rmsprop	1	0.0001	81.04	0.90	0.74	0.81
		2	0.0001	86.77	0.89	0.86	0.88
		3	**0.0001**	**88.07**	**0.89**	**0.89**	**0.89**
		4	0.0001	84.9	0.90	0.80	0.85
		5	0.0001	83.61	0.90	0.79	0.82
		6	0.0001	81.16	0.91	0.74	0.81
AE-LSTM	SGD	**1**	**0.0001**	**67**	**0.81**	**0.56**	**0.66**
		2	0.0001	62	0.74	0.50	0.60
		3	0.0001	46	0.87	0.60	0.11
Effect of learning rate on the proposed model							
AE-LSTM	Adam	2	0.00001	77	0.87	0.69	0.77
		2	**0.0001**	**89**	**0.88**	**0.94**	**0.91**
		2	0.001	83	0.92	0.76	0.83
		2	0.01	83	0.91	0.77	0.84
AE-LSTM	rmsprop	2	0.00001	75	0.87	0.66	0.75
		2	**0.0001**	**84**	**0.90**	**0.79**	**0.89**
		2	0.001	82	0.91	0.74	0.82
		2	0.01	82	0.92	0.75	0.83
AE-LSTM	SGD	2	0.00001	56	0.78	0.31	0.44
		2	0.0001	46	0.9	0.7	0.13
		2	0.001	34	0.42	0.42	0.42

dataset comparison with other metrics such as recall and F-score are not showing best performance among all models. Decision tree and LSTM are almost performing similarly with full feature set in terms of accuracy, recall and F-score.

3.2. Performance comparison with ae assisted baseline models

Fig. 15 demonstrates the results of baseline models on selected feature set using AE graphically. Among other techniques, GNB

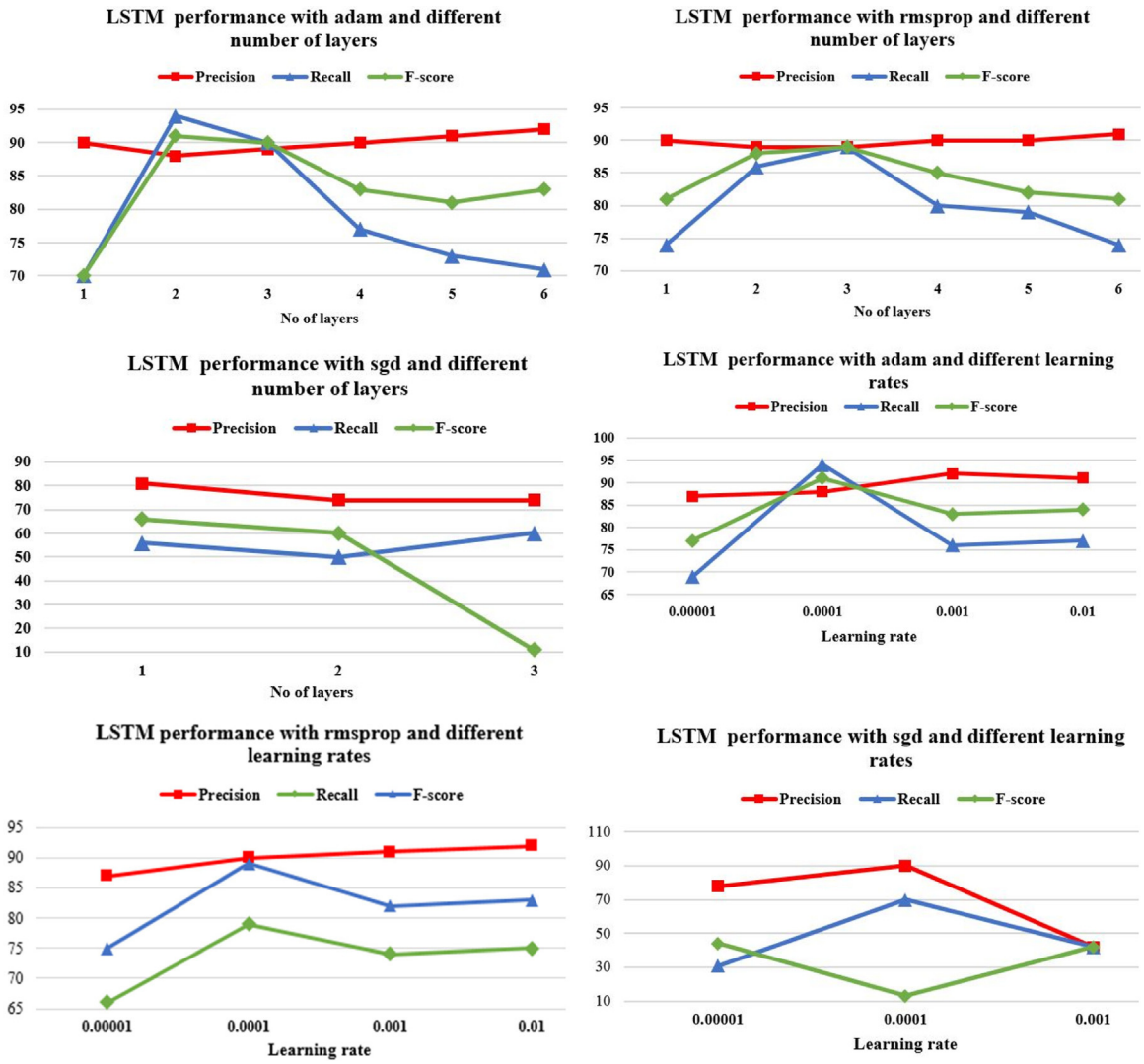


Fig. 13. Variants of LSTM under different settings.

Table 5
Proposed model results w.r.t precision, recall, F-score and detection rate on full feature set.

Proposed model results with full feature set					
Model	Accuracy	Precision	Recall	F-score	DR
GNB	56	0.33	0.49	0.36	0.33
SVC(linear)	77	0.79	0.79	0.77	0.79
SVC(polynomial)	75	0.78	0.77	0.75	0.78
SVC(rbf)	81	0.82	0.82	0.81	0.82
Decision tree	79	0.79	0.80	0.79	0.79
LSTM	**79**	**0.92**	**0.70**	**0.79**	**0.92**

is exhibiting good performance with accuracy of 83%, precision 0.84, recall 0.84, F-score 0.82 and DR 0.84 whereas the proposed AE-LSTM has an accuracy of 89%, precision 0.88, recall 0.94, F-score 0.91and DR 0.88. It is evident from the results that the proposed AE-LSTM performs significantly better than other techniques. Percentage improvement of the proposed AE-LSTM with existing methods in terms of accuracy is plotted in the form of bar charts in Fig. 15(f), in comparison to TSE-IDS, the AE-LSTM has 3.7% improved accuracy and from NBTree our proposed approach has 8.5% improved accuracy that shows effectiveness of AE-LSTM.

3.3. Comparison with the existing techniques

NSL-KDD has different sets for training and testing; in our experiments we used KDDTrain+_20% for training and KDDTest $^{+}$ is used for testing the proposed model. Comparison of the proposed AE-LSTM has been carried out with existing techniques and results are listed in Table 6. From this Table, it is evident that accuracy of the proposed AE-LSTM is highest among all the reported techniques including TDTC [32], FSSL [53], TSE-IDS [54] and SVM [55]. CFS-BA [26] is exhibiting the best performance among existing methods with accuracy of 87.37%, DR of 87.4% and FAR of 3.19% but the proposed AE-LSTM has achieved an accuracy of 89% with DR of 88% and FAR 11%. Although FS+GAR-forest [56] is performing significantly better in terms of FAR but it has used 32 features which are higher than the proposed AE-LSTM with 30 features only, also the proposed AE-LSTM has higher DR of 88% which is higher than its DR of 85.1%. Fuzzy [57] and TDTC [32] are also exhibiting comparatively low FAR in reported methods but fuzzy has used full features set and both methods have lower accuracy and DR then the proposed AE-LSTM. In RNN-IDS [58], RNN is employed for intrusion detection system. An accuracy of 83.28% has been achieved with RNN, which is lower than the proposed AE-LSTM. Similarly, DST-TL [46] constitutes sparse

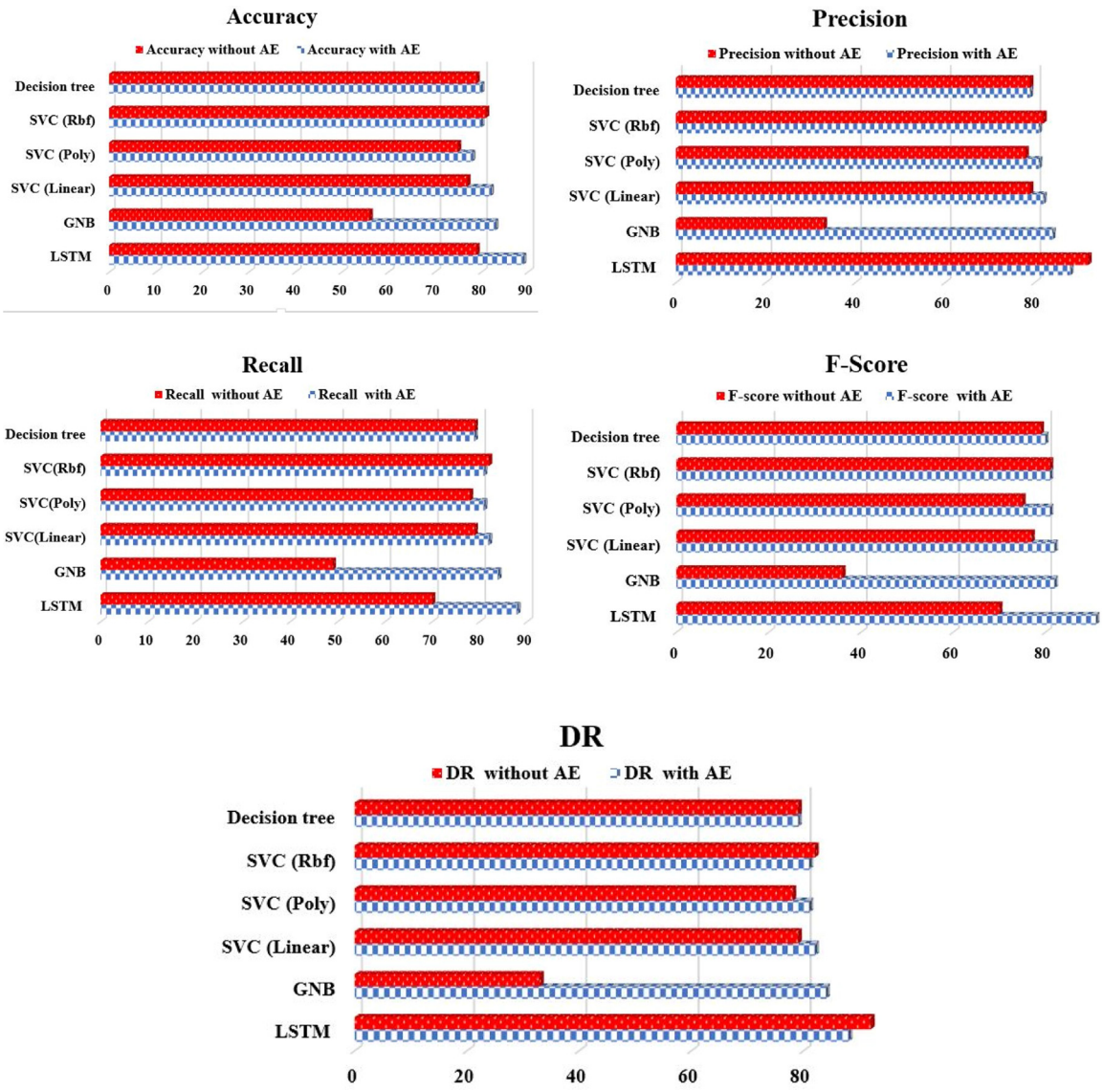


Fig. 14. Comparison of AE-LSTM with baseline classifiers in terms of accuracy, precision, recall, F-score and DR.

auto-encoder and proposed self-taught learning based IDS. It has shown good results on KDDTest+, however, in comparison to AE-LSTM, its accuracy and detection rate have smaller values. Another IDS based on deep learning architecture has been proposed, ICVAE-DNN [59], by exploiting improved conditional variational auto-encoder and deep neural network. This method has achieved improved accuracy of 85.97% but its DR is low 77.43% as compared to the reported techniques. BAT-MC [60] utilized Bi-LSTM with attention mechanism and obtained accuracy of 84.25% with 122 features which is less accurate than the proposed AE-LSTM with less number of features.

3.4. Complexity analysis

Training and testing time of proposed AE-LSTMs are plotted in Fig. 16 with baseline ML models as well. Results clearly depicts that training and testing time of baseline models is low with AE as compared to proposed models without AE, since full feature set needs more time to train and test. Only in case of AE-SVC we observed increased training time of 12 s and testing time of 6 s Training and testing time of SVC without AE is 09 s and 04 s respectively.

In the case of the proposed AE-LSTM, training time is 16 s and testing time is 3 s as compared to LSTM which has 33 s for training. It means, after applying AE its training time is reduced up to 17 s that has saved time. The training time of AE-BiLSTM and BiLSTM remains the same, *i.e.* 60 s, while testing time is reduced from 11 s to 9 s by incorporating AE with Bi-LSTM.

3.5. Statistical analysis

We have applied an independent sample t-test on results obtained from the proposed model such as accuracy, detection rate, and false alarm rate, and without applying AE on the proposed model for 20 independent runs and results are shown in Table 7.
H1: There is a significant difference in accuracy, DR and FAR of the proposed model with and without AE.

An independent sample t-test was conducted to compare the accuracy of proposed model with and without AE. There were significant differences between ($t = 34.672$ and p value $= 21.987$, $p<0.000$) in the scores with mean score for accuracy with AE ($M = 87.72$, $SD = 1.233$) was higher than that of accuracy without AE ($M = 77.40$, $SD = 1.698$). The magnitude of differences in the means (mean difference $= 10.318$, 95% CI: 9.368 to 11.267) was

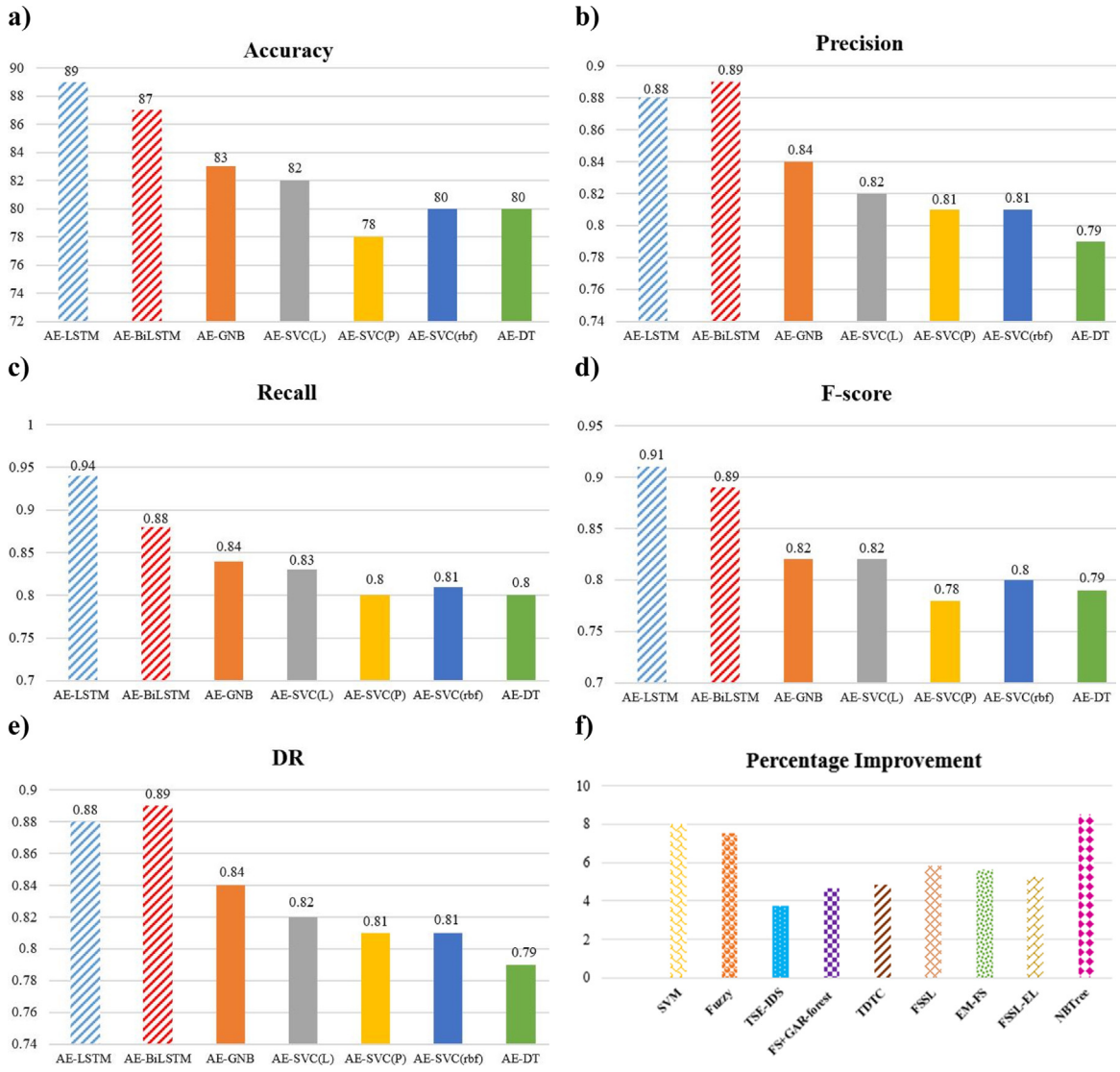


Fig. 15. Bar charts of proposed model performance measures and percentage improvement.

Table 6
Performance comparison of the proposed AE-LSTM with existing techniques.

Method	Dataset	Feature selection	Classifier	Features	ACC (%)	DR (%)	FAR (%)
FS+GAR-forest [56]	KDDTest$^+$	Symmetrical	GAR-forest	32	85.05	85.1	2.2
TDTC [32]	KDDTest$^+$	LDA+PCA	NB+CF-kNN	N/A	84.86	N/A	4.86
FSSL [53]	KDDTest$^+$	Clustering	FSSL	41	84.12	N/A	N/A
EM-FS [61]	KDDTest$^+$	IGR	Bagging(C4.5)	35	84.25	N/A	2.79
FSSL-EL [62]	KDDTest$^+$	PCA	Ensemble	20	84.54	N/A	5.31
TSE-IDS [54]	KDDTest$^+$	Hybrid	Two-stage ensemble	37	85.79	86.8	11.7
NBTree [63]	KDDTest$^+$	N/A	NBTree	41	82.02	N/A	N/A
Fuzzy [57]	KDDTest$^+$	N/A	Fuzzy classifier	41	82.74	86.7	3.9
SVM [55]	KDDTest$^+$	N/A	SVM	41	82.37	82	15
CFS-BA [26]	KDDTest$^+$	CFS-BA	Voting	10	87.37	87.4	3.19
RNN-IDS [58]	KDDTest$^+$	N/A	RNN	122	83.28	N/A	N/A
DST-TL [46]	KDDTest$^+$	N/A	Sparse auto-encoder	122	84.60	86	14
ICVAE-DNN [59]	KDDTest$^+$	N/A	DNN	122	85.97	77.43	N/A
BAT-MC [60]	KDDTest$^+$	122	Bi-LSTM with attention	122	84.25	N/A	N/A
AE-LSTM	**KDDTest$^+$**	**Auto-encoder**	**LSTM**	**30**	**89**	**88**	**11**

significant, hence H1 was supported. After that an independent sample t-test was conducted to compare the detection rate of the proposed model with and without AE. There were significant differences between ($t = 38$ and p value $= 53.331$, $p<0.000$) in the scores with mean score for DR with AE ($M = 0.9125$, $SD = 0.1832$) was higher than that of DR without AE ($M = 0.6910$, $SD = 0.00308$). The magnitude of differences in the means (mean difference $= 0.22150$, 95% CI: 0.21309 to 0.22991) was significant therefore H1 was supported. An independent sample t-test was conducted to compare the FAR of the proposed model with and without AE. There were significant differences between ($t = 38$ and p value $= 26.978$, $p<0.000$) in the scores with mean score for

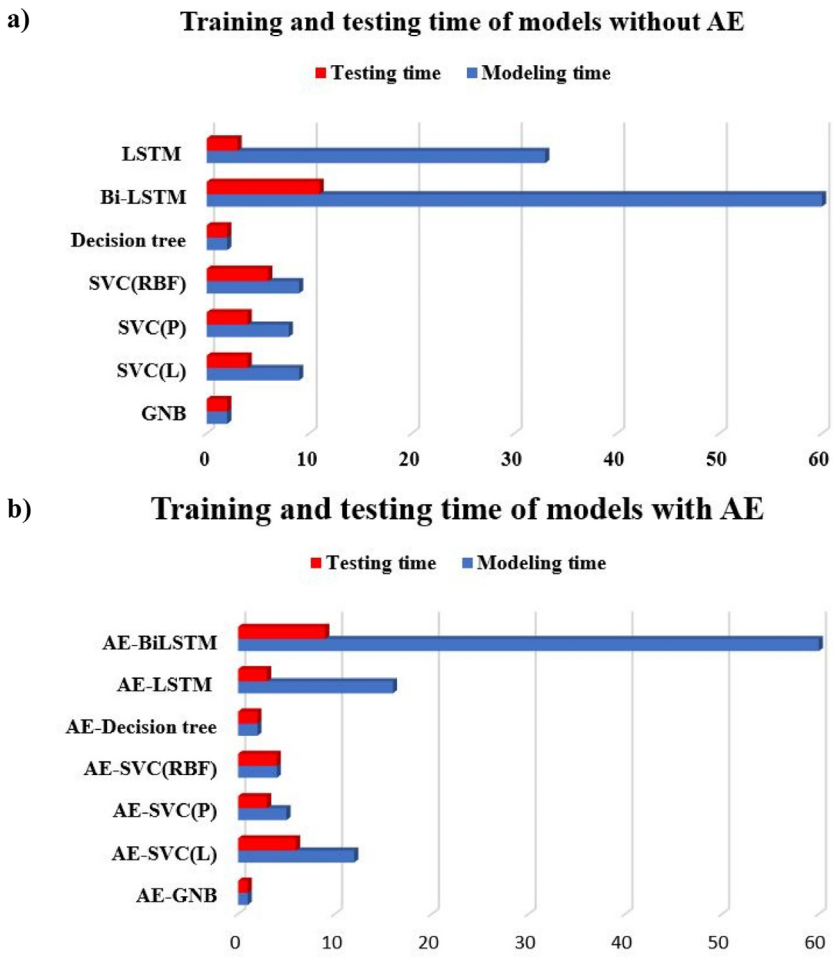


Fig. 16. Comparison of training and testing time of models, (a) without AE and (b) with AE.

Table 7
Independent sample t-test on the accuracy, DR and FAR of the proposed model with and without AE.

				Levene's Test for equality of variances						T-test for equality of means		
		Mean	SD	F	Sig	T	df	Sig (2-tailed)	Mean differences	Std Error Differences	95% CI of the differences	
											Lower	Upper
ACC	With AE	87.72	1.233	0.951	0.336	21.987	38	0.000	10.318	0.469	9.368	11.267
	Without AE	77.40	1.698									
DR	With AE	0.9125	0.01830	14.07	0.001	53.331	38	0.000	0.22150	0.00415	.21309	.22991
	Without AE	0.6910	0.00308									
FAR	With AE	0.1455	0.01050	6.424	0.015	26.978	38	0.000	0.07050	0.00261	.06521	.07579
	Without AE	0.0750	0.00513									

FAR with AE (M = 0.1455, SD = 0.01050) was higher than that of FAR without AE (M = 0.0750, SD = 0.00513). The magnitude of differences in the means (mean difference = 0.07050, 95% CI: 0.06521 to 0.07579) was significant, consequently H1 was supported.

4. Conclusion

This work proposed the novel anomaly detection system by applying AE for robust feature space selection and LSTMs model for classification by using benchmark dataset NSL-KDD. The outcomes of this study are listed as below:

- A novel intrusion detection system has been proposed AE-LSTM with enhanced accuracy of 89%, DR 88% and FAR 11% that significantly outperforms then recently reported techniques.
- The proposed model has been evaluated with full feature set as well as selected features obtained from AE. Experimental results demonstrate that the proposed model with reduced feature subset obtained from AE has better results on all the observed performance indices including accuracy, precision, recall and F-score.
- The proposed model is evaluated by comparing its results with the existing techniques in literature that confirms its efficiency and reliability.
- The proposed model can be used to detect network traffic into benign and malicious traffic with greater performance with experimental results and comparison with existing methods in terms of performance indices. Evaluation of the

proposed model suggest that it has greater capability to distinguish the two categories more accurately.

- Statistical analysis has been carried out by applying independent sample t-test on accuracy, DR and FAR of proposed model that shows the AE-LSTM's effectiveness.
- Despite these benefits, the proposed model has certain limitation: AE-LSTM and AE-BiLSTM have higher training time as compared to traditional ML methods owing to its sophisticated nature.
- In future, this work can be extended by applying some other feature selection technique and real time deployment of the model to classify network traffic. We intend to carry on attack type classification which represents incoming traffic as normal or attack type.

CRediT authorship contribution statement

Earum Mushtaq: Investigation, Writing, Visualization, Methodology. **Aneela Zameer:** Conceptualization, Methodology, Writing, Validation, Project administration. **Muhammad Umer:** Investigation, Visualization, Methodology. **Asima Akber Abbasi:** Visualization, Validation and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was carried out utilizing PIEAS resources. No other funding has been utilized to carry out this research.

References

[1] M.R. Gauthama Raman, Nivethitha Somu, Sahruday Jagarapu, Tina Manghnani, Thirumaran Selvam, Kannan Krithivasan, V.S. Shankar Sriram, An efficient intrusion detection technique based on support vector machine and improved binary gravitational search algorithm, Artif. Intell. Rev. (2020) 3255–3286.
[2] Jianwu Zhang, Yu Ling, Xingbing Fu, Xiongkun Yang, Gang Xiong, Rui Zhang, Model of the intrusion detection system based on the integration of spatial–temporal features, Comput. Secur. 89 (2020) 1–9.
[3] Akashdeep, Ishfaq Manzoor, Neeraj Kumar, A feature reduced intrusion detection system using ANN classifier, Expert Syst. Appl. (2017) 249–257.
[4] Wei Wang, Jiqiang Liu, Georgios Pitsilis, Xiangliang Zhang, Abstracting massive data for lightweight intrusion detection in computer networks, Inform. Sci. (2018) 417–430.
[5] G.A. Marin, Network security basics, IEEE Secur. Privacy 3 (6) (2005) 68–72.
[6] J. Jabez, B. Muthukumar, Intrusion detection system (IDS): Anomaly detection using outlier detection approach, Procedia Comput. Sci. 48 (2015) 338–346.
[7] Ozgur Depren, Murat Topallar, Emin Anarim, M. Kemal Ciliz, An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks, Expert Syst. Appl. 29 (4) (2005) 713–722.
[8] V. Jyothsna, V.V. Rama Prasad, K.M. Prasad, A review of anomaly based intrusion detection systems, Int. J. Comput. Appl. 28 (7) (2011) 26–35.
[9] Manasi Gyanchandani, J.L. Rana, R.N. Yadav, Taxonomy of anomaly based intrusion detection system: a review, Int. J. Sci. Res. Publ. 2 (12) (2012) 174–187.
[10] Sharmila Kishor Wagh, Vinod Pachghare, Satish Kolhe, Survey on intrusion detection system using machine learning techniques, Int. J. Comput. Appl. 78 (16) (2013) 30–37.
[11] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, Kuang-Yuan Tung, Intrusion detection system: a comprehensive review, J. Netw. Comput. Appl. 36 (1) (2013) 16–24.
[12] Seyed Mojtaba Hosseini Bamakan, Huadong Wang, Yong Shi, Ramp loss k-support vector classification-regression; A robust and sparse multi-class approach to the intrusion detection problem, Knowl.-Based Syst. 126 (2017) 113–126.
[13] Feng Jiang, Chen Yu-Ming, Outlier detection based on granular computing and rough set theory, Appl. Intell. 42 (2015) 303–322.
[14] Gulshan Kumar Ahuja, Krishan Kumar Saluja, Monika Sachdeva, The use of artificial intelligence based techniques for intrusion detection: a review, Artif. Intell. Rev. 34 (2010) 369–387.
[15] Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, Wei-Yang Lin, Intrusion detection by machine learning: A review, Expert Syst. Appl. 36 (10) (2009) 11994–12000.
[16] C. Kolias, G. Kambourakis, M. Maragoudakis, Swarm intelligence in intrusion detection: A survey, Comput. Secur. 30 (8) (2011) 625–642.
[17] Cosimo Ieracitanoa, A. Adeel, Francesco Carlo Morabito, Amir Hussain, A novel statistical analysis and autoencoder driven intelligent intrusion detection approach, Neurocomputing 387 (2020) 51–62.
[18] Sara Gasparini, Maurizio Campolo, Cosimo Ieracitano, Nadia Mammone, Edoardo Ferlazzo, Chiara ueri, Giovanbattista Gaspare Tripodi, Umberto Aguglia, Francesco Carlo Morabito, Information theoretic-based interpretation of a deep neural network approach in diagnosing psychogenic non-epileptic seizures, Entropy 20 (2) (2018) 1–12.
[19] Cosimo Ieracitanoa, Nadia Mammone, Alessia Bramanti, Amir Hussain, Francesco C. Morabito, A convolutional neural network approach for classification of dementia stages based on 2D-spectral representation of EEG recordings, Neurocomputing 323 (2019) 96–107.
[20] Kia Dashtipour, Mandar Gogate, Ahsan Adeel, Cosimo Ieracitano, Hadi Larijani, Amir Hussain, Exploiting deep learning for persian sentiment analysis, Int. Conf. Brain Insp. Cogn. Syst. (2018) 597–604.
[21] Yingjiao Ma, Jinglin Shi, Jinlong Hu, Reconfigurable remote radio head design and implementation for super base station applications, Ann. Telecommun. 73 (2018) 639–650.
[22] Xiao Sun, Man Lv, Facial expression recognition based on a hybrid model combining deep and shallow features, Cogn. Comput. 11 (2019) 587–597.
[23] Guoqiang Zhong, Shoujun Yan, Kaizhu Huang, Yajuan Cai, Junyu Dong, Reducing and stretching deep convolutional activation features for accurate image classification, Cogn. Comput. 10 (2018) 179–186.
[24] Lihua Wang, Bo Jiang, Zhengzheng Tu, Amir Hussain, Jin Tang, Robust pixelwise saliency detection via progressive graph rankings, Neurocomputing 329 (2019) 433–446.
[25] Radhika Chapaneri, Seema Shah, A comprehensive survey of machine learning-based network intrusion detection, Smart Intell. Comput. Appl. (2018) 345–356.
[26] Yuyang Zhoua, Guang Cheng, Shanqing Jianga, Mian Dai, Building an efficient intrusion detection system based on feature selection and ensemble classifier, Comput. Netw. 174 (2020) 107247.
[27] Sydney Mambwe Kasongo, Yanxia Sun, A deep learning method with wrapper based feature extraction for wireless intrusion detection system, Comput. Secur. 92 (2020) 101752.
[28] Sara A. Althubiti, Eric Marcell Jones, Kaushik Roy, Lstm for anomaly-based network intrusion detection, in: 28th International Telecommunication Networks and Applications Conference, ITNAC, 2018.
[29] Alaeddine Boukhalfa, Abderrahim Abdellaoui, Nabil Hmina, Habiba Chaoui, LSTM deep learning method for network intrusion detection system, Int. J. Electr. Comput. Eng. (IJECE) 10 (3) (2020) 3316–3322.
[30] Fanzhi Meng, Yunsheng Fu, Fang Lou, Zhiwen Chen, An effective network attack detection method based on kernel PCA and LSTM-rnn, in: 2017 International Conference on Computer Systems, Electronics and Control, ICCSEC, 2017, pp. 568–572.
[31] S. Yang, Research on network behavior anomaly analysis based on bidirectional LSTM, in: 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2019, 2019, pp. 798–802.
[32] Yu Yan, Lin Qi, Jie Wang, Yun Lin, Lei Chen, A network intrusion detection method based on stacked autoencoder and LSTM, in: ICC 2020-2020 IEEE International Conference on Communications, ICC, 2020, pp. 1–6.
[33] Mostofa Ahsan, Kendall Nygard, Convolutional Neural Networks with LSTM for Intrusion Detection CATA 2020 (EPiC Series in Computing), Vol. 69, 2020, pp. 69–79.
[34] Chia-Ming Hsu, Muhammad Zulfan Azhari, He-Yen Hsieh, Setya Widyawan Prakosa, Jenq-Shiou Leu, Robust network intrusion detection scheme using long-short term memory based convolutional neural networks, Mob. Netw. Appl. 26 (2021) 1137–1144.
[35] Congyuan Xu, Jizhong Shen, Xin Du, Fan Zhang, An intrusion detection system using a deep neural network with gated recurrent units, IEEE Access 6 (2018) 48697–48707.
[36] Simon Hawkins, Hongxing He, Graham Williams, Rohan Baxter, Outlier detection using replicator neural networks, Int. Conf. DataWarehousing Knowl. Discov. (2002) 170–180.
[37] H. Nguyen, Kim Phuc Tran, S. Thomassey, M. Hamad, Forecasting and anomaly detection approaches using LSTM and LSTM autoencoder techniques with the applications in supply chain management, Int. J. Inf. Manage. 57 (2021) 1–13.
[38] Ekaansh Khosla, Dharavath Ramesh, Rashmi Priya Sharma, Samuel Nyakotey, RNNs-RT: Flood based prediction of human and animal deaths in Bihar using recurrent neural networks and regression techniques, Procedia Comput. Sci. 132 (2018) 486–497.

[39] Xiaolei Ma, Zhimin Tao, Yinhai Wang, Haiyang Yu, Yuinpeng Wang, Long short-term memory neural network for traffic speed prediction using remote microwave sensor data, Transp. Res. C 54 (2015) 1876–1897.
[40] Yoshua Bengio, P. Simard, P. Frasconi, Learning long -term dependencies with gradient descent is difficult, IEEE Trans. Neural Netw. 05 (02) (1994) 157–166.
[41] B. Bakker, Reinforcement learning with long short term memory, in: Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 9–14, 2002, pp. 1475–1482.
[42] Farah Shahid, Aneela Zameer, Muhammad Muneeb, Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM, Chaos Solitons Fractals (2020).
[43] M. Schuster, K.K. Paliwal, Bidirectional recurrent neural networks, EEE Trans. Signal Process. 45 (11) (1997) 2673–2681.
[44] S. Sarraf, Analysis and detection of DDoS attacks using machine learning techniques, Am. Sci. Res. J. Eng. Technol. Sci. (ASRJETS) 66 (1) (2020) 95–104.
[45] Chuanlong Yin, Yuefei Zhu, Jinlong Efi, Xinzheng He, A deep learning approach for intrusion detection using recurrent neural networks, IEEE Access 5 (2017) 21954–21961.
[46] Aqsa Saeed Qureshi, Asifulla Khan, Nauman Shamim, Muhammad Hanif Durad, Intrusion detection using deep sparse auto-encoder and self-taught learning, Neural Comput. Appl. 32 (2019) 3135–3147.
[47] Quamar Niyaz, Weiqing Sun, Ahmad Y. Javaid, Mansoor Alam, A deep learning approach for network intrusion detection system, EAI Endorsed Trans. Secur. Saf. 3 (9) (2016).
[48] Salvatore J. Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, Philip K. Chan, Cost-based modeling for fraud and intrusion detection: results from the JAM project, in: Proceedings DARPA Information Survivability Conference and Exposition, DISCEX'00, 2000.
[49] Richard P. Lippmann, D.J. Fried, Isaac Graf, Joshua W. Haines, Kristopher R. Kendall, David McClung, Dan Weber, Seth E. Webster, Dan Wyschogrod, Robert K, Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation, in: Proceedings DARPA Information Survivability Conference and Exposition, DISCEX'00, 2000.
[50] John McHugh, Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by lincoln laboratory, ACM Trans. Inform. Syst. Secur. 3 (4) (2000) 262–294.
[51] Naveed Chouhan, Asifullah Khan, Haroon ur Rasheed Khan, Network anomaly detection using channel boosted and residual learning based deep convolutional neural network, Appl. Soft Comput. 83 (2019) 1–18.
[52] Kamran Shaukat, Suhuai Luo, Vijay Varadharajan, Ibrahim A. Hameed, Shan Chen, Dongxi Liu, Jiaming Li, Performance comparison and current challenges of using machine learning techniques in cybersecurity, Energies (2020).
[53] Rana Aamir Raza Ashfaqa, Xi-Zhao Wang, Joshua Zhexue Huanga, Haider Abbas, Yu-Lin He, Fuzziness based semi-supervised learning approach for intrusion detection system, Inform. Sci. (2017) 484–497.
[54] Bayu Adhi Tama, Marco Comuzzi, Kyung-Hyune Rhee, TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system, IEEE Access 7 (2019) 94497–94507.
[55] Muhammad Shakil Pervez, Dewan Md. Farid, Feature selection and intrusion classification in NSL-KDD cup 99 dataset employing SVMs, in: The 8th International Conference on Software, Knowledge, Information Management and Applications, SKIMA 2014, 2014.
[56] Navaneeth Kumar Kanakarajan, Kandasamy Muniasamy, Improving the accuracy of intrusion detection using GAR-forest with feature selection, in: Proceedings of the 4th International Conference on Frontiers in Intelligent Computing: Theory and Applications, FICTA, 2015, pp. 539–547.
[57] Pavel Kromer, Jan Platoš, Václav Snášel, Ajith Abraham, Fuzzy classification by evolutionary algorithms, IEEE Int. Conf. Syst. Man Cybern. (2011).
[58] Chuanlong Yin, Yuefei Zhu, Jinlong Fei, Xinzheng He, A deep learning approach for intrusion detection using recurrent neural networks, IEEE Access 5 (2017) 21954–21961.
[59] Yanqing Yang, Kangfeng Zheng, Chunhua Wu, Yixian Yang, Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network, Sensors 19 (11) (2019) 1–20.
[60] Tongtong Su, Huazhi Sun, Jinqi Zhu Sheng Wang, Yabo Li, BAT- deep learning methods on netwrok intrusion detection using NSL-KDD dataset, IEEE Access 8 (2020) 29576–29585.
[61] Ngoc Tu Pham, Ernest Foo, Suriadi Suriadi, Helen Jeffrey, Hassan Fareed M. Lahza, Improving performance of intrusion detection system using ensemble methods and feature selection, Proc. Austr. Comput. Sci. Week Multiconf. (2018) 1–6.
[62] Ying Gao, Yu Liu, Yaqia Jin, Juequan Chen, Hongrui Wu, A novel semi-supervised learning approach for network intrusion detection on cloud-based robotic system, IEEE Access 6 (2018) 50927–50938.
[63] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, Ali A. Ghorbani, A detailed analysis of the KDD CUP 99 data set, in: Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications, CISDA 2009, 2009.