Knowledge-Based Systems 226 (2021) 107132

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Unified Deep Learning approach for Efficient Intrusion Detection System using Integrated Spatial–Temporal Features

P Rajesh Kanna, M.E., Assistant Professor [a,*], P Santhi, M.E., Ph.D., Professor [b]

[a] Department of Information Science and Engineering, Bannari Amman Institute of Technology, India
[b] Department of Computer Science and Engineering, M.Kumarasamy College of Engineering, India

ARTICLE INFO

Article history:
Received 1 October 2020
Received in revised form 11 March 2021
Accepted 10 May 2021
Available online 14 May 2021

Keywords:
Intrusion detection systems
Deep learning
spatial–temporal features
Optimized Convolutional Neural Networks
Hierarchical Multi-scale LSTM
Lion Swarm Optimization
NSL-KDD
UNSWNB15

ABSTRACT

Intrusion detection systems (IDS) differentiate the malicious entries from the legitimate entries in network traffic data and helps in securing the networks. Deep learning algorithms have been greatly employed in the network security field for large scale data in modern cyberspace networks because of their ability to learn the deeply integrated features. However, learning both space and time aspects of system information are very challenging for any individual deep knowledge model. While Convolutional Neural Networks (CNN) effectively acquires the spatial aspects, the Long Short-Term Memory (LSTM) neural networks perform better for temporal features. Integrating the benefits of these models has the potential for improving the large scale IDS. In this paper, a high accurate IDS model is proposed by using a unified model of Optimized CNN (OCNN) and Hierarchical Multi-scale LSTM (HMLSTM) for effective extraction and learning of spatial–temporal features. The proposed IDS model performs the pre-processing, feature extraction through network training and network testing and final classification. In the OCNN–HMLSTM model, the Lion Swarm Optimization (LSO) is used to tune the hyper-parameters of CNN for the optimal configuration of learning spatial features. The HMLSTM learns the hierarchical relationships between the different features and extracts the time features. Lastly, the unified IDS approach utilizes the extracted spatial–temporal features for categorizing the network data. Tests are performed over public IDS datasets namely NSL-KDD, ISCX-IDS and UNSWNB15. Assessing the performance of OCNN–HMLSTM against the contemporary IDS methods, the proposed model performs better intrusion detection with high accuracy of above 90% with less false values and better classification coefficients.

1. Introduction

Recent developments in the field of information coupled with the advanced communication paradigms led to the increasing number of online services. These online service systems rely on network technology for advancements which increases network security issues. The critical constituents of the communication network are confidentiality, integrity and availability. Any abnormal activity is termed based on the impact it creates on the compromise of these constituents. These abnormal activities are called network intrusions and are considered illegal. Intrusion detection system (IDS) [1] has been the saviour against these intrusion attacks and often coupled with the general security framework of the network. The IDS model acts as the scrutiny model for monitoring the security threats in the network traffics. The typical IDS model monitors all the inbound and outbound traffic packets for finding the signs of abnormal traffic for intrusions. A robust IDS model can recognize maximum properties of the intrusion actions and can automatically warn the server system for alerting the entire network [2]. For ensuring sophisticated monitoring, the IDS model combines the software and hardware devices based on a specified set of security policies. An ideal IDS model must be able to detect the intrusions dynamically and enable effective protection against all different patterns of the attack strategies.

Based on the dynamic detection property of the IDS, they are categorized into three types namely, misuse detection approach or signature-based system (SBS) [3], anomaly-based system (ABS) [4] and stateful protocol analysis technique [5]. SBS has been built using the digital signature matching process which compares the packet signature with the attack signature database to identify the intrusion. If a traffic packet is suspected of intrusion, its signature is matched against the database and if there

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author.
E-mail addresses: mailmeatrajeshkanna@gmail.com (P Rajesh Kanna), santhipalanisamypec@gmail.com (P Santhi).

https://doi.org/10.1016/j.knosys.2021.107132
0950-7051/

is a similarity, the alarm is outstretched. As the SBS involves the use of a database for attack knowledge analysis, it is also called knowledge-based IDS. Due to these benefits, the SBS achieves high accuracy of detection with a minimum false positive rate. But it has limitations in handling the unknown attacks with new attack strategies. The other approach, ABS is based on the attack behaviour analysis where the attacks are identified through comparison of attack behaviour against the normal traffic behaviours. ABS helps in identifying unknown attacks more effectively than the SBS but it cannot differentiate the anomaly in certain cases where the attacker imitates the normal behaviour. The stateful protocol analysis technique incorporates the benefits of both the SBS and ABS techniques [6]. Unlike the SBS and ABS, this technique identifies the attacks using the signature as well as the behaviour at the communication protocol stage itself. The IDS model proposed in this article is also based on this idea of integrating both the misuse detection and anomaly detection approaches.

Conventional approaches such as encryption, access control, firewalls and anti-attack software models were the staple of traditional intrusion detection. These approaches were effective for small scale attack identification while has limitations in detecting a large number of attacks and results in high false detections. Particularly, the hackers initiate the Denial-of-Service (DoS) vulnerabilities [7] in large numbers which are hard to distinguish and protect via conventional techniques. Most recent studies have shifted their focus on integrating the machine learning (ML) algorithms for intrusion detection [8,9]. They tend to escalate the recognition rates besides diminishing the overhead in handling large scale attacks, unlike the conventional methods. Support Vector Machine (SVM) [10,11] can predict the target attacks in the test dataset using the attributes of the training data and is also memory efficient. The IDS model using SVM has the simpler design of the prediction model using the hyperplanes and kernel functions to determine the attack class. K-nearest neighbour (KNN) [12] provided a fast and efficient design with low complexity. It assigns the data as samples in search space and applies the single or multidimensional feature vector for determining the neighbouring samples having the closest resemblance as the normal or attack class. Naïve Bayes [13] employs the simplest form Bayesian probability model with independent assumptions on the attributes and the bias and variance to reduce the error rate in detecting the attacks. Random forests algorithm [14] utilizes the embedded feature selection method and intrinsic metrics to rank features for attack class categorization. It also reduces complexity and memory utilization. Other algorithms such as k-means clustering and logistic regression employ the clustering and regression designs for determining the attack classes. Still, the standard ML algorithms have serious limitations in learning the deeply integrated features of the attacks [15]. Additionally, these algorithms provide less performance for noisy and high dimensional traffic data. Hybrid ML algorithms were developed by merging two or more ML algorithms for IDS but the model complexity is high which makes them less efficient. Similarly, the artificial neural networks (ANN) [16] and extreme learning machine (ELM) [17] also increase the complexity issues. Some variants of ELM [18,19] were found to be effective for intrusion detection but the extensive training time is often a big problem.

Recently, deep learning (DL) algorithms are taking the centre-stage in intrusion detection. The intrusion recognition models that are constructed on non-linear structured DL systems like CNN [20], recurrent neural networks (RNN) [21], LSTM [22], etc. have provided higher learning behaviours and increased the intrusion detection accuracy rates [23–28]. Recent years has also seen upgrades in the hardware architectures to support the DL models for their generous security improvements.CNN based models have utilized shift-invariant, shared-weight architecture of convolution kernels to scan the hidden layers and other characteristics. CNN regularized the multi-layer perceptron and hence utilize fully connected networks to determine the attack class. However, certain aspects of this fully connected network result in over-fitting data. RNN based IDS models utilize the feed-forward neural network structure with the internal memory dedicated to processing the lengthy-time sequences of the input traffic data. It includes the feedback loops to control the state of operation and improve the learning rate. LSTM is an RNN model that can be used for IDS to process sequences of traffic data. Unlike standard RNN, LSTM constitutes the feedback connections that are well suited for intrusion detections with temporal features. Extensive researches on the DL based IDS models have shown some important revelations. The non-linear deep architecture of CNN and LSTM helps extract the features of network traffics [29–31]. CNN has provided high detection accuracy by learning the deep spatial features through spatial correlations with faster learning rates [32]. However, it is found that CNN does not automatically learn the temporal features and hence certain parameter and model tuning are required. As temporal features have become vital behavioural features for detecting the attack patterns, they need an additional architecture for learning the long-term dependencies in the temporal features [33]. LSTM is the special class of RNN that can learn the time features from network traffics more effectively with their short term sequence memory. However, LSTM takes more training time and also requires more resources than CNN and also has limited knowledge of past information [34].

These investigations reveal the efficiency of CNN namely high accuracy and faster training time while accurate and effective processing of time features are the strengths of LSTM. Integrating these two models can help in harvesting their benefits and even overcoming their limitations. Additionally, tuning the hyper-parameters of CNN can also be efficient. Many studies have used this strategy of optimizing the CNN parameters. In [35], an optimal trajectory using fuzzy multi-objective transcription is proposed and utilized to optimally train the deep neural network (DNN) to model it as the optimal command generator. In [36], pre-generated trajectory ensemble optimization is used to train the DNN for establishing optimal feedback actions for the functional relationship between optimized systems states and design controls. In [37], a DNN model is used with a desensitized trajectory optimization method to obtain the optimal parking trajectories with initial uncertainty. These approaches optimized the DNN to obtain the feedback actions for controlling the automatic parking. The common advantage of these techniques is the optimization of the parameters of DNN to improve design control. Based on these studies, a hybrid DL model is proposed in this paper in which CNN is optimized using Lion Swarm Optimization (LSO) [38]. This improves the training speed of CNN for large scale network data. Likewise, the LSTM model is modified by adapting the hierarchical structure for multi-scale network data. Comparing with [35–37] and other popular optimization algorithms, the LSO has better global probability convergence and has provided global optimal solutions with high precision [38]. These advantages of LSO over the other optimization algorithms have led to its usage in this paper. Therefore, an advanced unification model of OCNN–HMLSTM is developed in this study by jointly using the optimized CNN and HMLSTM models. The vital offerings of this study are:

- The development of efficient IDS using a novel unified DL model of OCNN–HMLSTM. It is a joint model that exploits the deep learning patterns of CNN and LSTM, for learning both the spatial–temporal features.

- The hyper-parameters in CNN of the proposed OCNN-HMLSTM are tuned using the new meta heuristic of Lion Swarm Optimization which increases the learning rate for spatial features while HMLSTM learns the temporal features.
- The evaluations are performed on three popular and large scale public IDS datasets of NSL-KDD, ISCX-IDS and UNSWNB15 using the MATLAB tool.
- The existing IDS models based on ML and DL models are evaluated for performance comparisons which showed that the proposed OCNN–HMLSTM accurately detects the attacks and outperforms the other models during cross-validation tests.

This research article is structured as: Related study in Section 2. The description of the proposed OCNN–HMLSTM and their implementations are explained in Section 3. Performance results and comparative investigation are illustrated in Section 4. Inferences of this research and possible future instructions are given in Section 5.

2. Related works

The last decade has seen a surge in the usage of ML and DL algorithms for IDS models. The ability to provide good predictive accuracy through effective feature learning has been the major advantage in these learning-based IDS models. Most existing IDS models have been based on supervised learning models. SVM has been used predominantly for IDS. Wang et al. [10] utilized SVM with feature augmentation for IDS and evaluated it on NSL-KDD with 99.18% accuracy, 99.85% detection rate and 2.96% false alarm rate. Similarly, Usha and Kavitha [11] also utilized the SVM based IDS model which was evaluated on the AWID dataset to achieve 99.25% accuracy, 99% precision, 1.2% false-positive rate within 1.6 h of learning time and 0.85 h testing time.K-nearest neighbour (KNN) has been used by Meng et al. [12] for constructing alarm filter of IDS model which was tested on DARPA1999 dataset with 88.6% accuracy and 86.3% f-measure with CPU workload threshold rate of 89%. Likewise, Serpenand Aghaei [39] also utilized KNN for the IDS model via Principal component analysis (PCA) feature selection for the ADFA-LD dataset and achieved 99.5% accuracy, 99.8% true positive rate, 99.7% true negative rate, 0.3% false-positive rate and 0.2% false-negative rate. Mukherjee and Sharma [13] used Naïve Bayes based IDS for the NSL-KDD dataset and achieved 97.78% detection accuracy, 97.8% true positive rate and 2.2% false-positive rate. K-means clustering has been employed for real-time live attack detection in wireless sensor networks [40] with 1.2% false-positive rates. Aunga and Min [41] applied the K-means algorithm for intrusion detection from the KDD99 dataset and provided a detection accuracy of 99.8% and 0.3% false-positive rate. Farnaazand Jabbar [14] developed IDS using Random forests and applied it on NSL-KDD to achieve 99.67% precision and 0.8% false-positive rate. Peng et al. [42] used the Decision tree (DT) for KDD99 and detected the intrusions with 88% accuracy and a calculation time of 3.4 s. Besharati et al. [43] utilized logistic regression NSL-KDD and detected the attacks with 97.5% accuracy. Although high overall accuracy rates are obtained, these machine learning algorithms have low detection rates for many attack types especially the U2R and R2L in NSL-KDD and KDD datasets and also the balance between false positives and negatives are uneven in most methods with false positives increasing with decreasing false negatives.

Some studies applied hybrid machine learning and advanced machine learning algorithms to reduce false positives. Teng et al. [44] developed SVM-DT-based collaborative IDS which were evaluated on KDD99 with 89.02% detection accuracy, 12% error rate and consuming 7.25 s training time. Tao et al. [45] developed Feature selected Weight and Parameter optimized-SVM (FWP-SVM) using the genetic algorithm for intrusion detection on KDD99 and achieved 96.61% accuracy, 0.07% false negatives and 3.39% false positives with a minimum classification time of 5.078 s and 2.4% error rate. Zhang et al. [46] developed Gaussian Naïve Bayes (GNB) for network IDS and tested it on the KDD99 dataset for which it achieved 91.06% accuracy and 0.494 s detection time. Mazini et al. [47] developed a hybrid artificial bee colony (ABC) and Ada Boost algorithms for anomaly detection in NSL-KDD and ISCXIDS2012 datasets with 98.9% detection accuracy, and 1.1% false-positive rate. Khraisat et al. [48] proposed an ensemble of hybrid IDS using C5 classifier and One-Class SVM classifier. This hybrid model was tested on the Bot-IoT dataset and achieved 94% malware detection and overall accuracy of 99.97%.

Advanced machine learning-based IDS techniques include neural networks based techniques. Shenfield et al. [16] utilized ANN on benign network traffic dataset and achieved 98% accuracy, sensitivity 95% and 1.8% false positives. Baig et al. [49] utilized multiclass cascaded ANN on KDD99 with 98.25% accuracy, the false-positive rate of 3.77% and the false-negative rate of 1.26% and the UNSWNB15 dataset with 86.4% accuracy and 2.8% false-positive rate. Sumaiya Thaseen et al. [50] used an integrated model of Neural Networks with correlation-based feature selection for IDS on NSL-KDD and UNSWNB15 datasets. This neural network resulted in 98.45% accuracy and 500 s of computation time for NSL-KDD and 96.4% accuracy and 660 s computation time for UNSWNB15.Atli et al. [17] used an extreme learning machine (ELM) on ISCX-IDS 2012 dataset with 99% detection accurateness and 1% false-positive rate with 42.12 s training time. Singh et al. [18] utilized online sequential ELM (OSELM) on NSL-KDD with an accuracy of 98.66% and a false-positive rate of 1.74% are achieved in 2.43 s detection time while Gao et al. [19] used incremental ELM (I-ELM) on NSL-KDD with 81.22% accuracy, 30.03% false alarm rate and 19.97 s detection time and UNSWNB15 dataset with 77.36% accuracy, 36.09% false alarm rate and 476.18 s detection time. Although efficient than the conventional and hybrid machine learning algorithms, the NN and ELM models have limitations in handling larger datasets due to their shallow architecture that reduces their ability to abstract the DL features of network data.

Numerous studies have been conducted using DL algorithms as an IDS tool. Yin et al. [23] developed RNN for intrusion detection which was applied on the NSL-KDD dataset and achieved 97% accuracy and 1765 s training time. Kim et al. [24] used CNN-based IDS for DoS attacks which were applied on KDD99 with 99% accuracy and CSE-CIC-IDS2018 with 91.5% accuracy. Nguyen and Kim [25] used genetic CNN based IDS for the NSL-KDD dataset in which the attacks were detected at 98.2% accuracy, 0.52% false-positive rate and 95.44% true positive rate. Althubiti et al. [26] employed LSTM for IDS which was tested on the CIDDS dataset with 85% accuracy and 17.22% false-positive rate. Likewise, Chawla et al. [27] employed Bidirectional LSTM for the ADFA dataset with 90% accuracy and 25% false alarm rate. Amar and Ouahidi [28] used Weighted LSTM on ISCX-UNB with 97% accuracy, 22% loss and 1.47% false alarm rate. Apart from CNN and RNN/LSTM, many other DL algorithms have also been used individually or joint with other algorithms for intrusion detection. However, CNN and LSTM have always provided better attack detection in most of the intrusion datasets. Still, they also suffer from certain limitations mentioned in the previous section. To alleviate their limitations, some authors have tried using hybrid models or unified models. Khan et al. [29] developed Convolutional LSTM (Conv-LSTM) for the IDS model which was tested on the ISCX-UNB dataset with 97.29% accuracy and 0.71% false alarm rate.

As described earlier, the extraction of spatial–temporal features can significantly enhance intrusion detection but it is quite

Table 1
Data distribution in the NSL-KDD dataset.

Set	Total	Normal	DoS	Probe	R2L	U2R
KDDTrain^{+}	125 973	67 343	45 927	11 656	995	52
KDDTest^{+}	22 544	9711	7458	2421	2754	200
KDDTest^{-21}	11 850	2152	4342	2402	2754	200

challenging. Wang et al. [30] employed DNN to learn spatial–temporal features. It was applied on DARPA1998 and ISCX2012 with 99.8% accuracy and 0.22% false alarm rate with 1.7 min detection time. Similarly, Zhang et al. [31] developed a combined model of multi-scale CNN (MSCNN) and LSTM for intrusion detection. It was tested on the UNSWNB15 dataset and achieved 95.6% accuracy, 9.8% false alarm rate and 1.6% false-negative rate with 1060 s computation time. However, both these models suffer from limitations in handling the imbalanced datasets. Considering the limitations of [30] and [31], and also analysing the benefits of [29], a unified DL model named OCNN–HMLSTM has been proposed in this article. It is intended to tackle these limitations and exploit the benefits of both CNN and LSTM for intrusion detection in three different datasets. This will be helpful especially in environments where multiple traffic patterns are experienced due to the multiple applications that are run at the same time.

3. Materials and methodology

The proposed OCNN–HMLSTM model for intrusion detection automatically uses the properties of CNN and LSTM for extracting the spatial–temporal aspects of network traffic data. Then HMLSTM performs the final classification using these two set of features. The spatial–temporal features integration using the OCNN–HMLSTM model is shown in Fig. 1.

3.1. Datasets

In most studies, the IDS models were developed focusing on multiple attacks on a single intrusion detection dataset. Not all the ideal IDS models performed similarly on different datasets. Most IDS models are suitable for specific datasets only while performs poorly for another dataset. Hence in this work, three prominent intrusion detection public datasets will be used to test the proposed OCNN–HMLSTM. The principal objective is designing adaptive IDS that can perform comparatively better in most datasets with various attacks. It is also important to select the appropriate datasets since they are vital in the evaluation process of the IDS. The three datasets selected for this work are NSL-KDD, ISCX-IDS 2012 and UNSWNB15.

NSL-KDD dataset: It was generated in 2009 as an alternative to the popular KDDCUP99 dataset. KDDCUP99 was been utilized for more than two decades. As KDDCUP99 has characteristic replicated entries, the researchers developed NSL-KDD as an improved KDD. It also does have a sensible quantity of data entries for preparation and assessment, so that the classification does not depend on frequent records. The NSL-KDD contains I training—KDDTrain^{+} and 2 testing—KDDTest^{+} and KDDTest^{-21} sets, containing diverse legitimate entries and four primary categories of attacks. Table 1 displays the classes and the sum of records in every category of NSL-KDD.

As shown above, the NSL-KDD has one normal class and four attack classes namely DoS, R2L (Root to Local attacks), U2R (User to Root attack), and Probe attacks. It contains 41 aspects and one class label for each traffic data. NSL-KDD includes 10 basic, 12 content and 19 traffic features.

ISCX-IDS 2012: This dataset was created with seven kinds of datasets collected for 7 days of a week in June 2010. It includes two profiles: α and β profiles. While α profile introduces anomalous behaviour in the network, the β profile represents the features and mathematical distributions of the procedures. It includes four attack scenarios namely intrusion from the inside, HTTP, DoS, DDoS, and brute force Secure Shell (SSH). It contains a total of 2,381,532 normal records and 68,792 malicious records. Table 2 shows the class distribution of ISCX-IDS.

Table 2
Data distribution in ISCX-IDS 2012.

Network flow	Training		Testing	
	Normal	Malicious	Normal	Malicious
Friday	55 640	0	24 500	0
Saturday	85 222	1353	45 889	1353
Sunday	220 024	9833	42 345	875
Monday	108 945	2451	58 664	1320
Tuesday	347 308	24 295	187 012	13 083
Wednesday	339 470	0	182 793	0
Thursday	255 054	3381	137 338	1822

Table 3
Data distribution in UNSWNB-15.

Class	Training set	Testing set	Testing set A
Normal	56 000	37 000	2485
Reconnaissance	10 491	3496	457
Backdoor	1746	583	233
Worms	130	44	44
Analysis	2000	677	301
Shellcode	1133	378	255
Generic	40 000	18 871	457
Fuzzers	18 184	6062	457
Dos	12 264	4089	457
Exploits	33 393	11 132	457
Total	175 341	82 332	5576

UNSWNB-15 dataset: This dataset contains a mixture of real recent legitimate activities and malicious attack behaviours. It includes nine attack categories namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. There are 44 aspects deprived of a class label for each traffic records. It includes basic, content, Time, additional features and 1 attack class. Table 3 shows dataset numbers for training, testing and testing after sampling.

3.2. Pre-processing

Pre-processing comprises data tuning and normalization. It anticipates enumerating and regularizing the data. In the original dataset, particular point values are symbolic, continuous and many other types. These values must be converted to numerical type for the processing of data. Likewise, some of the values may not in the specified range due to different representation. For example, an attribute might be represented with three decimal values while other attributes by 2 decimal values. These must be normalized to support the linear handling of the data.

3.3. Spatial feature processing using OCNN

CNN is engaged in this work to study the spatial features by mapping inputs as level flow images. The proposed OCNN model is developed by optimizing the parameters of CNN using LSO. Fig. 2 illustrates the organization of CNN that will be optimized. CNN comprises four main operators namely Convolution, pooling layer, and fully connected layer and non-linear activation function.

Convolution layer (CL): It forms the major core of CNN that analyses and extracts the desired features. This convolution task

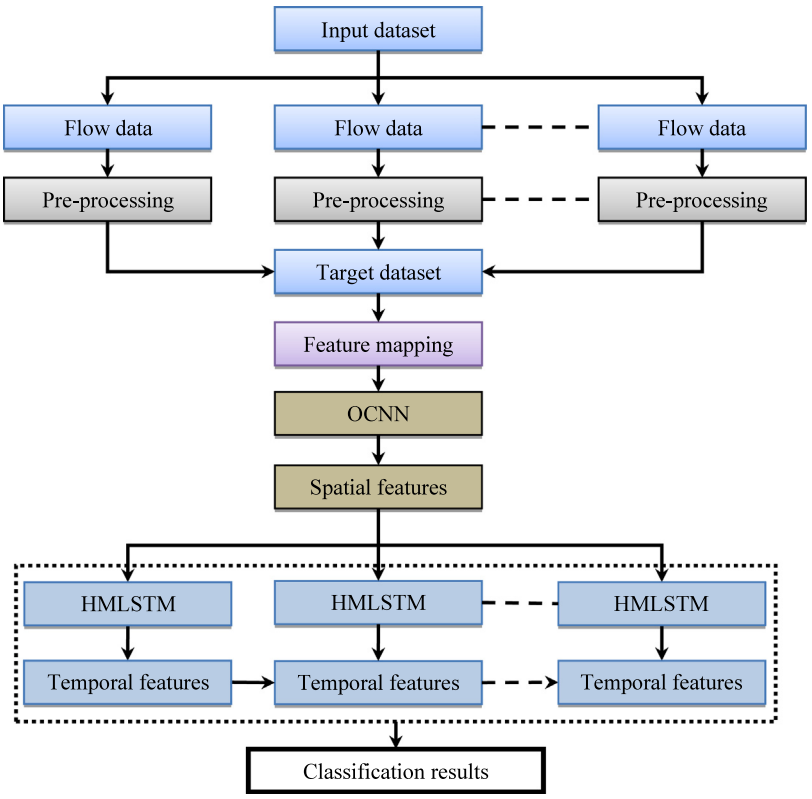

Fig. 1. Spatial–temporal features integration in OCNN–HMLSTM intrusion-detection model.

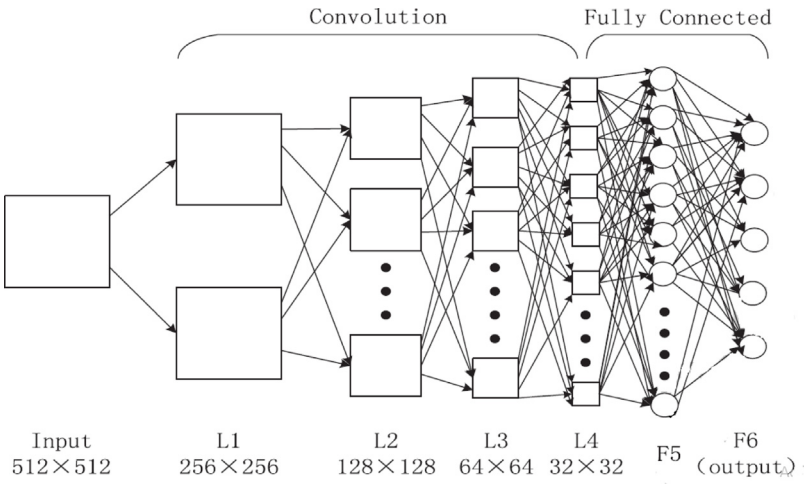

Fig. 2. Structure of CNN.

conserves the spatial connection amongst the input data by acquiring the aspects by the kernel function. The outcome of the CL will be the convolved aspect plot. The kernel points are updated automatically based on the optimal structure configuration. The magnitude of the aspect plot is reliant on the depth of the layers.

Non-linear activation (NLA): After the convolution operation, the additional nonlinear function is used before the creation of feature maps. The NLA can either be tanh, sigmoid or Rectified Linear Unit (ReLU). This NLA acts as the element-wise task to compromise the negative points of the aspects. In most cases, the sigmoid or ReLU provided better performance.

Pooling layer (PL): Spatial pooling is the sub-sampling or down-sampling process in CNN, performed to reduce the dimensionality of the feature maps. It is similar to the feature reduction process that removes the less important data while retaining vital information. Kinds of pooling are average, max, stochastic and sum pooling denoted by the pooling numbers 1–4. In most cases, max-pooling provides the most important features.

Fully-connected Layer (FCL): It is a conventional multi-level neural layer employing a softmax initiation utility in the outcome

Table 4
Hyper-parameter value ranges in CNN.

Hyper-parameter	Range	Difference
CL	1–4	1
PL	1-N_c	1
FCL	2–5	1
Hidden units/layer	256–1024	256
Pooling type	1–4	1

layer. The FCL has the preceding layer nodes interlinked with the succeeding layer nodes. The complex aspects yielded from CL and PL are used by this FCL for labelling the data into classes using the past learning knowledge.

Combining all these operators, CNN is formed. The hyper-parameters used in CNN are listed in Table 4. LSO is used to optimize these parameters. Here N_c denotes the maximum number of convolutional layers.

LSO [38] is based on the intelligent characteristics of three inhabitants including shielding lion, hunting lioness and dependent cubs.LSO converges fast and is not easily trapped into a locally optimal solution. For performing the LSO, initially, a random configuration is set for OCNN. A proportion factor of adult lions as β is arbitrarily assigned. When β is small, the proportion of lion cubs is large, thus improving the detective ability and increases the diversity of the population. For quick convergence, the β value is set below 0.5. Similarly, a disturbance factor α is determined to improve the exploration ability (divergence). α is calculated as

$$\alpha = step.\exp\left(-\frac{30t}{T}\right)^{10} \quad (1)$$

Here t is the current iteration, T denotes maximum iterations and *step* denote the largest step of the lion in motion. When $\alpha > 1$, the speed of movements increase over time and the lions hardly adjust their direction to reach the optimum solution and the swarm diverges. When $\alpha = 0$, the speed movement drops towards zero and the lions move aimlessly without knowledge of the previous motion. When $\alpha \ll 1$, less momentum is attained from the previous stage and direction changes quickly to adjust the reset process. The increase of disturbance factor balances the local search ability and global search ability and accelerates the convergence speed by avoiding the premature problem. In this manner, the LSO analyses and determines the optimal configuration. Algorithm 1 shows the LSO process.

Algorithm 1: Lion Swarm Optimization steps
Input: Population size, Maximum number of CL, PL and FCL
Begin
Initialize lion population, the position of each lion, iteration =0
Set proportion factor for adult lion and moving range for the lioness and cubs
For each lion category
    Calculate the number of each lion category
    For each lion
    Calculate fitness (Here learning error rate)
    Update positions of king lion, lionesses, and cubs
    End for
Select the lion king (pbest) based on the constraint violations of positions
Rearrange the order of lions using fitness
Mark highest order lion as Gbest
Increment iterations by 1
If maximum iterations reached
    Terminate algorithm
    Return the Gbest
End if
End for
End

Based on the LSO, the CNN architecture is modified. The weight initialization is performed using the LSO before the fitness evaluation. Then the individual will be decoded and trained with k epochs for the CNN in the training process to obtain different error rates. While many configurations for the OCNN are found by the tuning process, LSO selects the configuration with a minimum error rate. The top four configurations for OCNN obtained using LSO are shown in Table 5.

Table 5
Top configurations of OCNN obtained using LSO.

Layer type	Configuration
OCNN configuration 1	
CL	2 layers
PL	2 layers; max pooling
FCL	3 layers; 512 units
Error rate	18.3
OCNN configuration 2	
CL	3 layers
PL	3 layers; max pooling
FCL	3 layers; 512 units
Error rate	17.5
OCNN configuration 3	
CL	3 layers
PL	1 layer;max pooling
FCL	2 layers; 256 units
Error rate	18.8
OCNN configuration 4	
CL	4 layers
PL	2 layers; max pooling
FCL	3 layers; 1024 units
Error rate	17.2

Considering the configurations from the above table, the OCNN configuration with less error rate is chosen. In this case, OCNN configuration 4 has less error rate of 17.2 and hence it will become the optimal CNN architecture. This model will perform similar to the model designed manually by human experts. This OCNN can extract the spatial features by setting many kernels of varying sizes. The most common kernels are the convoluted 1*1, 2*2, and 3*3 kernels among which the 2*2, and 3*3 kernels learn the features accurately while 1*1 kernel helps in increasing the learning rate.

3.4. Temporal feature learning and classification using HMLSTM

LSTM neural network is often used to extract the time domain aspects in time-series data. In this work, a modified LSTM named HMLSTM is utilized for this purpose. HMLSTM has many variations. In this work, the HMLSTM modelled by Zhao et al. [51] has been tweaked for the IDS model requirements. The general LSTM uses three gates to get inputs and processes them with the sigmoid activation function. In this HMLSTM, a parameterized boundary detector is introduced which obtains binary output value in each layer for learning the termination conditions to get the temporal features. Additionally, the dense connections are introduced to allow the layer ℓ to absorb the feature maps of all previous layers as input and produce a concatenation of feature maps. This process improves the spatial feature learning property of the LSTM model to increase the classifier efficiency for intrusion detection. When the boundary detector is set as 1 at a time step of layer ℓ, the HMLSTM model considers it as the end segment and feeds the summarized representation into the upper layer $(\ell + 1)$ i.e. the narrower dense layers for spatial learning. Based on the boundary states, the layers select any one operation from (Update, Copy or Flush). First, the standard LSTM equations

are formulated.

$$\text{Gates and candidate: } \begin{bmatrix} i_t \\ f_t \\ u_t \\ o_t \end{bmatrix} = Wx_t + Uh_{t-1} + b \tag{2}$$

$$\text{Cell state: } c_t = c_{t-1} \odot \sigma(f_t) + \tanh(u_t) \odot \sigma(i_t) \tag{3}$$

$$\text{Hidden state: } h_t = \sigma(o_t) \odot \tanh(c_t) \tag{4}$$

Here x_t denotes the current input to LSTM, h_{t-1} denotes previous hidden state, and c_{t-1} denotes the previous cell state. These three parameters are the input to the LSTM. i_t, f_t, u_t and o_t represent the input, forget, candidate activation and output gates, respectively. W and U are the weight matrix and activation function matrix, respectively, while b denotes the bias.

Adding the boundary detector variable (z_t) to these standard functions,

$$\begin{bmatrix} i_t \\ f_t \\ u_t \\ o_t \\ z_t \end{bmatrix} = Wx_t + Uh^1_{t-1} + z_{t-1}Vh^2_{t-1} + b \tag{5}$$

Here z_{t-1} is the previous boundary variable with $\ell = 1$; x_t becomes the input at the current time-step for bottom-up connection, (h^1_{t-1}, c^1_{t-1}) denoting the hidden and cell states in recurrent connection with $\ell = 1$, h^2_{t-1} denoting hidden state in top-down connection with $\ell = 2$ and V denoting the activation boundary matrix.

The proposed HMLSTM model having L layers ($\ell = 1, 2, \ldots, L$) and at each layer, the update process is performed at time t

$$h^\ell_t, c^\ell_t, z^\ell_t = f^\ell_{HMLSTM}(c^\ell_{t-1}, h^\ell_{t-1}, h^{\ell-1}_t, h^{\ell+1}_{t-1}, z^\ell_{t-1}, z^{\ell-1}_t) \tag{6}$$

The function f^ℓ_{HMLSTM} denotes the forget gate of HMLSTM which is determined by the two boundary states $z^\ell_{t-1}, z^{\ell-1}_t$. The cell states can be updated as

$$c^\ell_t = \begin{cases} f^\ell_t \odot c^\ell_{t-1} + i^\ell_t \odot g^\ell_t & \text{if } z^\ell_{t-1} = 0 \text{ and } z^{\ell-1}_t = 1 \ (Update) \\ c^\ell_{t-1} & \text{if } z^\ell_{t-1} = 0 \text{ and } z^{\ell-1}_t = 0 \ (Copy) \\ i^\ell_t \odot g^\ell_t & \text{if } z^\ell_{t-1} = 1 \ (Flush) \end{cases} \tag{7}$$

Here g denotes the cell proposal vector. Unlike the standard LSTM, the forget, input and output gates are needed to be computed at every time step along with g.

The Update operation is executed to update the summary illustration of the layer ℓ if the boundary z^ℓ_{t-1} is found at the bottom layer but $z^{\ell-1}_t$ not present in the previous time step. This case occurs very rarely and hence Update operation is sparsely used. The copy operation simply performs $(h^1_t, c^1_t) \leftarrow (h^1_{t-1}, c^1_{t-1})$ which means the top layer stays unchanged until the bottom layer summarized input is received. The Flush operation has two sub-tasks: Eject to bypass the current state and reach the top layer while RESET operation to reinitialize the states at every new segment. This means Reset allows the top layer to absorb the bottom layer summary but deletes it if Eject is not performed.

The gate values $(f^\ell_t, i^\ell_t, o^\ell_t)$, cell proposal g^ℓ_t and the pre-activation of the boundary detector $\tilde{z}^\ell_t = hardsigm(Uh^\ell_t)$ can be determined at each operation by the slice function

$$\begin{bmatrix} f^\ell_t \\ i^\ell_t \\ o^\ell_t \\ g^\ell_t \\ \tilde{z}^\ell_t \end{bmatrix} = \begin{bmatrix} sigm \\ sigm \\ sigm \\ tanh \\ hardsigm \end{bmatrix} f_{slice}(s^{recurrent(\ell)}_t + s^{top-down(\ell)}_t + s^{bottom-up(\ell)}_t + b) \tag{8}$$

Here

$$s^{recurrent(\ell)}_t = U^\ell_\ell h^\ell_{t-1} \tag{9}$$

$$s^{top-down(\ell)}_t = z^\ell_{t-1} U^\ell_{\ell+1} h^{\ell+1}_{t-1} \tag{10}$$

$$s^{bottom-up(\ell)}_t = z^{\ell-1}_t W^\ell_{\ell-1} h^{\ell-1}_t \tag{11}$$

As the HMLSTM uses a top-down connection from layer $\ell + 1$ to ℓ, the activation is possible only when the boundary is detected at the previous time step. This ensures that the layer is initialized with more long-term dependencies. The gating operations of the HMLSTM are given in Fig. 3. The output module for the HMLSTM learning temporal features with three layers can be obtained as shown in Fig. 4.

The binary boundary state z^ℓ_t is determined as

$$z^\ell_t = f_{bound}(\tilde{z}^\ell_t) \tag{12}$$

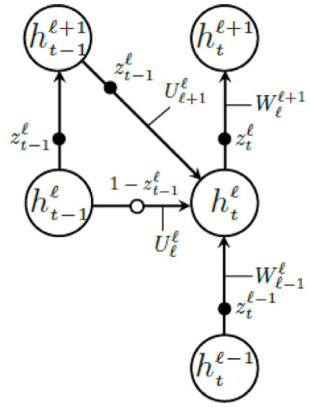

Fig. 3. Gating operation of HMLSTM.

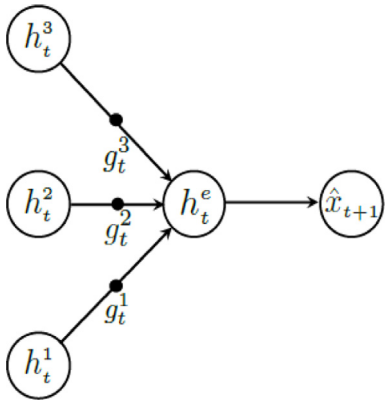

Fig. 4. Output module of HMLSTM.

It can be modelled using the deterministic step function

$$z_t^{\ell} = \begin{cases} 1 & if\ \tilde{z}_t^{\ell} > 0.5 \\ 0 & otherwise \end{cases} \tag{13}$$

Thus the HMLSTM model can be formulated and used to learn the long term dependencies of the temporal features and the spatial features with the short term memory.

3.5. Model uncertainty of OCNN-HMLSTM

The model uncertainty is the standard deviation of the results that provide spatial information at which point the model is uncertain. It occurs when the training data is insufficient or not proper. Model uncertainty is estimated from the network trained with a Monte Carlo dropout rate of 50% in each hidden layer [52]. It is estimated by applying to global uncertainty scores namely Monte Carlo dropout uncertainty and Dice uncertainty. Monte Carlo dropout uncertainty provides the score of how certain or uncertain the model based on the given input data. High Dice uncertainty ensures that the model is less certain of the obtained results and is highly variable. It is done by using the Monte Carlo dropout U-Net architecture which is trained using the Dice loss function and the Monte Carlo dropout is turned on. For testing the uncertainty, the three intrusion datasets are used. For testing, the Monte Carlo dropout uncertainty inference is performed 1000 times to obtain 1000 results. Then the values of Monte Carlo dropout uncertainty and Dice uncertainty are calculated. Then linear regression is performed to determine the relationship between the Monte Carlo dropout uncertainty and Dice uncertainty. It is found that the Monte Carlo dropout uncertainty and Dice uncertainty agreement have achieved the values of 0.01, 0.005 and 0.0033 for NSL-KDD, ISCX-IDS 2012 and UNSW-NB15, respectively which means the correlation is strong. It implies that the proposed model has very less uncertainty that has a negligible impact on the performance without uncertainty. This condition will be evaluated further when noisy data are used in future researches.

4. Results and discussion

The evaluation of the proposed OCNN–HMLSTM is performed using MATLAB R2016b under a controlled environment. Experiments are performed to compare the efficiency of OCNN–HMLSTM on the three datasets individually. The comparisons are made against the existing SVM [10], NN [50], ELM [17], CNN [24], LSTM [26], Conv-LSTM [29], DNN [30] and MSCNN [31] based IDS models from literature. Besides, the individual performance of OCNN and HMLSTM for the intrusion detection problem is also evaluated separately and compared with the proposed unified model of OCNN–HMLSTM.

4.1. Evaluation metrics

The proposed OCNN–HMLSTM is evaluated using accuracy, precision, recall, F-measure, false Positive rate (FPR), false-negative rate (FNR), MCC and Kappa coefficient metrics.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{14}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{15}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{16}$$

$$F - measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{17}$$

Table 6
Performance of OCNN–HMLSTM on NSL-KDD.

Class	Accuracy (%)	FPR (%)	FNR (%)
Normal	93.61 ± 0.07	3.87 ± 0.01	5.23 ± 0.03
DoS	90.20 ± 0.03	8.61 ± 0.03	4.45 ± 0.02
Probe	89.10 ± 0.05	7.34 ± 0.02	6.21 ± 0.06
R2L	90.66 ± 0.08	9.15 ± 0.05	6.66 ± 0.03
U2R	91.11 ± 0.07	9.92 ± 0.02	9.87 ± 0.04

$$FPR = \frac{FP}{(FP + TN)} \tag{18}$$

$$FNR = \frac{FN}{(FN + TP)} \tag{19}$$

Mathew Correlation Coefficient (MCC) varies between −1 and 1, where the best binary classifier obtains positive 1 and worst classifier obtains negative 1.It is computed as

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{20}$$

Kappa coefficient is used to check whether the classifier can process imbalanced data classes successfully. It is calculated as

$$K = \frac{Obsolute - Expect}{1 - Expect} \tag{21}$$

Here $Obsolute = Accuracy$ and $Expect = \frac{A+B}{(TP+TN+FP+FN)}$. The values of A and B can be obtained as $A = \frac{(TP+FN)(TP+FP)}{(TP+TN+FP+FN)}$ and $B = \frac{(FP+TN)(FN+TN)}{(TP+TN+FP+FN)}$.

Kappa coefficient values ≤ 0 denote worst classifier, (0–0.2) denote slight promise, (0.21–0.4) as reasonable, (0.41–0.60) as modest, (0.61–0.80) as significant and (0.81–1) as nearly perfect classifier.

4.2. NSL-KDD evaluation

The evaluation of the OCNN–HMLSTM on NSL-KDD has provided better results. The most prominent measures are accuracy, FPR and FNR. Table 6 displays the attack recognition outcomes obtained on the test dataset—KDDTest^{+}.

From Table 6, it can be seen that the OCNN–HMLSTM effectively classifies the intrusions with higher accuracy. The outcomes demonstrate that the accuracy of detection is improved for almost all the attacks records and normal records. Likewise, the FPR and FNR values are also significantly ranged between good numbers. This result has been obtained when considering 100% of the testing dataset. Partitioning the test dataset into different parts such as 10%, 20% ...90% of results are also obtained and shown in Fig. 5.

Fig. 5(a) shows the classification metrics. 5(b) shows the coefficient values and 5(c) shows the FNR and FPR values. It is apparent that for all the parameters, the suggested OCNN–HMLSTM has achieved significantly good results.

Table 7 shows the evaluation results of the proposed OCNN–HMLSTM evaluated over the NSL-KDD compared against the existing SVM [10], NN [50], ELM [17], CNN [24], LSTM [26], Conv-LSTM [29], DNN [30] and MSCNN [31] based IDS models and the OCNN only and HMLSTM only IDS models.

From the above table, it is evident that the proposed OCNN–HMLSTM has comparatively better performance than the existing models. The proposed model has high values of accuracy, recall and f-measure and less value of FNR and FPR. Although the SVM based IDS model has achieved 100% precision and less detection time of 20.99 s, it has low values for other parameters. Particularly, the trade-off between FPR and FNR is huge (47.9% and 1.04%), thus making the model less efficient. LSTM achieved

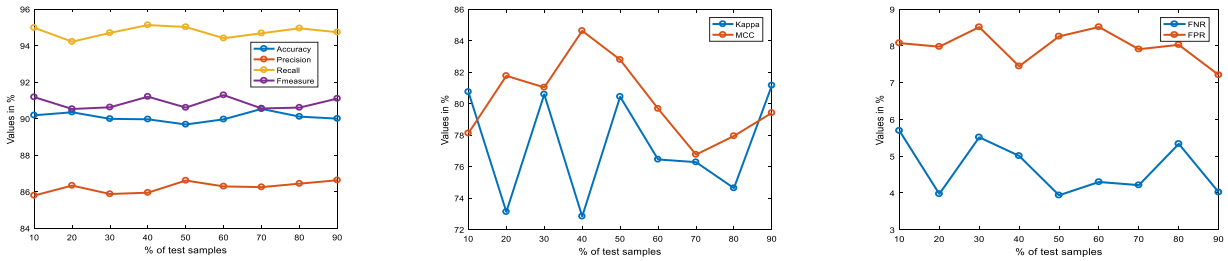

Fig. 5. Performance of OCNN–HMLSTM on NSL-KDD.

Table 7
Performance of OCNN–HMLSTM vs. other methods in literature on NSL-KDD dataset.

Methods/Metrics	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	FPR (%)	FNR (%)	MCC	Kappa	Training time (s)	Detection time (s)
SVM [10]	87.05	**100**	88.502	87.33	47.9	1.04	0.659	**0.9193**	4155	**20.99**
NN [50]	83.67	88.406	86.211	83.28	23.47	33.42	0.3151	0.7844	6790	106.23
ELM [17]	81.33	81.94	84.91	81.95	26.02	56.28	0.6979	0.7747	9865	25.349
CNN [24]	79.67	68.182	83.35	72.508	18.62	19.83	0.6821	0.6282	7756	93.015
LSTM [26]	83	86.77	83.39	87.57	15.03	17.51	0.2174	0.7842	7865	75.46
Conv-LSTM [29]	89.94	80.501	88.87	88.77	11.8	9.95	0.789	0.8334	8123	82.55
DNN [30]	89.33	86.67	92.56	90.67	12.67	11.2	0.8101	0.7567	7655	46.6
MSCNN [31]	88.45	79.89	90.11	88.76	9.94	6.67	0.8080	0.8337	6545	33.34
OCNN only	88.67	84.34	90.12	89.78	11.89	7.89	0.7678	0.8112	5868	32.35
HMLSTM only	87.11	78.89	93.67	88.4	12.2	6.66	0.8110	0.80	4245	30.67
OCNN–HMLSTM	**90.67**	86.71	**95.19**	**91.46**	**8.86**	**5.78**	**0.8222**	0.8633	5118	32.97

Table 8
Performance of OCNN–HMLSTM on ISCX-IDS 2012.

Class	Accuracy (%)	FPR (%)	FNR (%)
Friday	96.46 ± 0.06	9.22 ± 0.03	5.50 ± 0.02
Saturday	95.33 ± 0.08	7.88 ± 0.01	4.67 ± 0.03
Sunday	95.43 ± 0.04	7.76 ± 0.03	4.45 ± 0.03
Monday	96.86 ± 0.06	9.89 ± 0.03	5.67 ± 0.02
Tuesday	91.10 ± 0.05	6.87 ± 0.02	4.88 ± 0.06
Wednesday	93.87 ± 0.07	8.16 ± 0.06	4.16 ± 0.07
Thursday	96.55 ± 0.03	7.67 ± 0.02	8.75 ± 0.04

86.77% precision which is fractionally higher than the proposed model but does not have better results for the false positives and detection time. Comparing the individual models of OCNN and HMLSTM, HMLSTM has performed better; however, on overall analysis, the proposed unified model outperforms the individual models. Thus analysing the compared IDS models, it can be stated that the proposed model has significantly better performance and has advantages than the other models for the NSL-KDD data.

4.3. ISCX-IDS 2012 evaluation

Table 8 shows the attack detection results obtained on the test datasets of ISCX-IDS.

From Table 8, it can be seen that the OCNN–HMLSTM effectively classifies the intrusions with higher accuracy and low false values. The outcomes explain that the accuracy of recognition is improved for almost all the attacks records and normal records. Likewise, the FPR and FNR values are also significantly ranged between good numbers. Partitioning the test dataset into different parts such as 10%, 20%,…90% of results are also obtained and shown in Fig. 6.

Fig. 6(a) shows the classification metrics. 66(b) shows the coefficient values and 6(c) shows the FNR and FPR values. For all the parameters, the suggested OCNN–HMLSTM has achieved significantly good results. For comparison, Table 9 shows the evaluation results of the proposed OCNN–HMLSTM evaluated over the ISCX-IDS 2012 compared against the existing SVM [10], NN [50], ELM [17], CNN [24], LSTM [26], Conv-LSTM [29], DNN [30] and MSCNN [31] based IDS models and the OCNN only and HMLSTM only IDS models.

From the above table, it is evident that the proposed OCNN–HMLSTM has comparatively better performance than the existing models. All the compared models have the highest precision values. Apart from the FNR and Kappa coefficient, the proposed model has better results for all other parameters. It has the highest accuracy, precision, recall, f-measure and MCC values while also achieving low values of FPR. Also, the trade-off between FPR

Table 9
Performance of OCNN–HMLSTM vs. other methods in literature on ISCX-IDS 2012 dataset.

Methods/Metrics	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	FPR (%)	FNR (%)	MCC	Kappa	Training time (s)	Detection time (s)
SVM [10]	94.834	100	89.75	97.35	34.7	5.17	0.9188	0.8780	14 400	2100
NN [50]	66	100	78.72	79.52	35.4	34.0	0.6277	0.5676	62 240	2760
ELM [17]	66.33	100	76.23	79.76	45.1	33.67	0.6286	0.6436	86 580	2400
CNN [24]	93.55	100	87.67	92.84	14.5	18.7	0.9083	**0.9718**	98 768	2456
LSTM [26]	93.33	100	85.45	97.6	12.54	4.67	0.9228	0.9179	112 367	2110
Conv-LSTM [29]	95.29	100	91.91	93.17	11.3	6.78	0.9110	0.9056	88 970	2433
DNN [30]	95.05	100	92.22	95.66	19.1	4.7	0.9	0.911	79 895	1867
MSCNN [31]	94.9	100	91.67	90.88	9.9	4.6	0.897	0.9034	75 550	1985
OCNN only	93.56	100	93.34	97.2	8.89	4.95	0.9004	0.889	68 455	2080
HMLSTM only	94.99	100	90.75	95.38	8.75	**4.58**	0.912	0.9182	64 780	**1756**
OCNN–HMLSTM	**95.333**	100	**94.77**	**97.611**	**7.84**	4.67	**0.928**	0.9121	54 480	1765

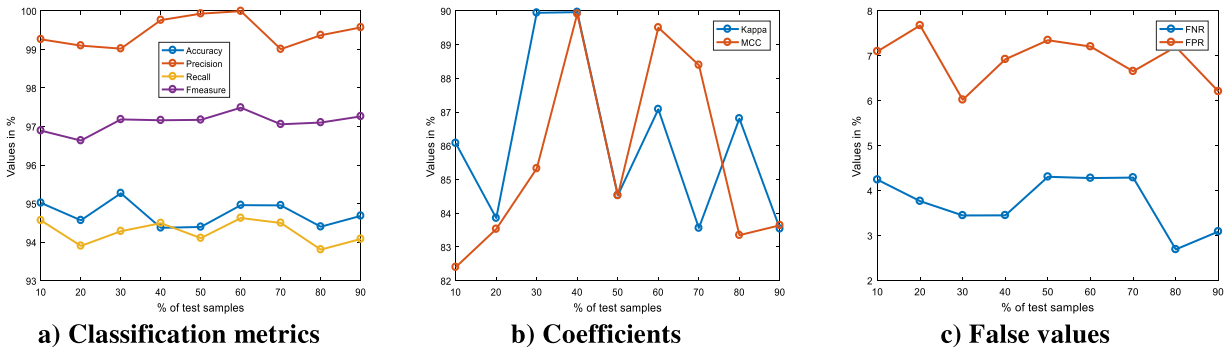

Fig. 6. Performance of OCNN–HMLSTM on ISCX-IDS 2012.

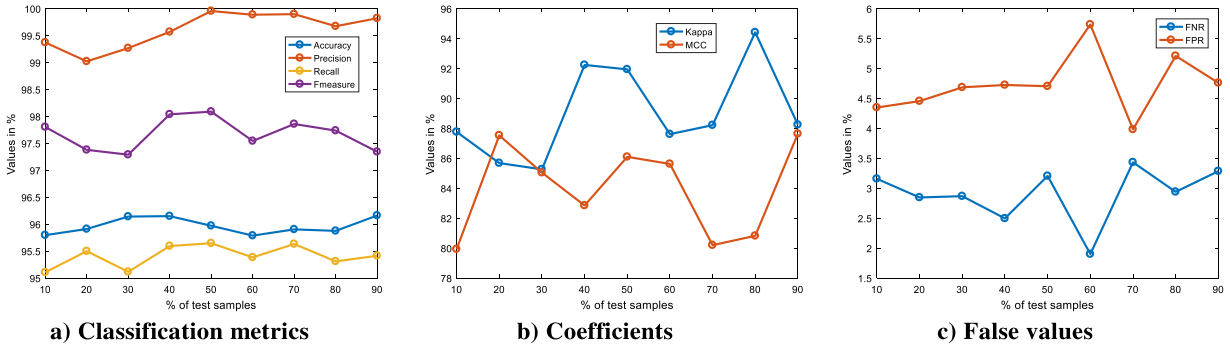

Fig. 7. Performance of OCNN–HMLSTM on UNSWNB-15.

Table 10
Performance of OCNN–HMLSTM on UNSWNB-15.

Class	Accuracy (%)	FPR (%)	FNR (%)
Normal	96.33 ± 0.04	5.87 ± 0.04	5.67 ± 0.02
Reconnaissance	96.43 ± 0.06	8.76 ± 0.05	3.67 ± 0.01
Backdoor	98.86 ± 0.09	7.66 ± 0.03	4.45 ± 0.04
Worms	94.10 ± 0.1	7.76 ± 0.05	3.48 ± 0.05
Analysis	97.87 ± 0.03	5.54 ± 0.02	3.33 ± 0.04
Shellcode	98.55 ± 0.05	4.67 ± 0.03	6.67 ± 0.04
Generic	89.76 ± 0.06	9.76 ± 0.05	5.52 ± 0.04
Fuzzers	93.67 ± 0.07	8.20 ± 0.06	4.87 ± 0.03
Dos	96.54 ± 0.04	6.76 ± 0.02	3.45 ± 0.02
Exploits	92.25 ± 0.07	5.45 ± 0.04	6.23 ± 0.03

and FNR is aptly balanced in the proposed unified model. Although the individual HMLSTM achieved low detection time, the proposed unified model is only fractionally higher. CNN achieved a high Kappa coefficient value indicating its superior classification ability, but still underperformed in comparison with the OCNN–HMLSTM model. Thus, it can be stated that the proposed model has significantly better performance and has advantages than the other models for the ISCX-IDS 2012 data.

4.4. UNSWNB-15 evaluations

Table 10 indicates the attack detection outcomes obtained on the test data of UNSWNB-15. These results are obtained on the unbalanced test dataset.

From Table 10, it can be seen that the OCNN–HMLSTM effectively classifies the intrusions with higher accuracy and low false values. The outcomes exemplify that the accuracy of detection is superior for almost all the attacks records and normal records. Likewise, the FPR and FNR values are also significantly ranged between good numbers. Partitioning the test dataset into different parts such as 10%, 20%,…90% of results are also obtained and shown in Fig. 7.

Fig. 7(a) shows the classification metrics. 7(b) shows the coefficient values and 7(c) shows the FNR and FPR values. The outcomes are noticeable that for all the metrics the proposed OCNN–HMLSTM has achieved significantly good results. For comparison, Table 11 shows the evaluation results of the proposed OCNN–HMLSTM evaluated over the UNSWNB-15 dataset compared against the existing SVM [10], NN [50], ELM [17], CNN [24], LSTM [26], Conv-LSTM [29], DNN [30] and MSCNN [31] based IDS models and the OCNN only and HMLSTM only IDS models.

From the above table, it is apparent that the proposed OCNN–HMLSTM has comparatively better performance than the existing models. All the compared models have the highest precision values of 100%. Apart from recall, FPR, detection time and Kappa coefficient, the proposed model has better results for all other parameters. It has the highest accuracy, precision, f-measure and MCC values while also achieving low values of FNR. Although the individual CNN achieved low FPR and HMLSTM achieved high recall, the proposed unified model has better advantages than them. CNN achieved a high Kappa coefficient value while SVM consumed less detection time. This indicates that these models are well suited for the UNSWNB-15 dataset, but still comparing the other properties of the OCNN–HMLSTM model, it shows the proposed model would outperform those models in terms of accuracy, precision, f-measure and FNR. Even the detection time is considerably low considering the complexities of the proposed unified model and the complex dataset. Thus, it can be stated that the proposed model is efficient than the other models for the UNSWNB-15 data.

4.5. Effect of parameter variation in OCNN– HMLSTM

The individual attack detection performance of the OCNN–HMLSTM is significantly good and they outperform the existing models in terms of most of the assessment metrics. With the variation of parameters of OCNN–HMLSTM, the performance varies

Table 11
Performance of OCNN–HMLSTM vs. other methods in literature on UNSWNB-15 dataset.

Methods/Metrics	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	FPR (%)	FNR (%)	MCC	Kappa	Training time (s)	Detection time (s)
SVM [10]	89.926	100	89.978	83.231	35.2	30.07	0.7939	0.711	12 577	**366.6**
NN [50]	66.33	100	77.48	79.76	33.5	33.67	0.5742	0.610	36 869	724.8
ELM [17]	66.33	100	76.84	79.759	47.12	33.67	0.6563	0.5725	24 300	456.92
CNN [24]	92.85	100	90.11	94.25	17.8	17.45	0.938	**0.9484**	39 650	622.7
LSTM [26]	93	100	91.24	96.373	11.55	7.0	0.8707	0.9269	44 500	633.34
Conv-LSTM [29]	94.75	100	93.5	94.78	9.8	4.45	0.8976	0.897	48 600	542
DNN [30]	95.08	100	94.8	95.67	11.7	6.67	0.9190	0.9	34 550	480.75
MSCNN [31]	95.6	100	92.1	96.7	7.56	5.22	0.933	0.9389	36 750	512.5
OCNN only	94.67	100	95	97.51	**5.03**	4.12	0.9005	0.91	32 180	444.67
HMLSTM only	96.22	100	**96.2**	98.05	6.75	3.69	0.9378	0.8711	31 900	423.44
OCNN–HMLSTM	**96.334**	100	95.87	**98.132**	5.87	**3.67**	**0.9489**	0.8917	30 665	475.88

Table 12
Effect of parameter variation in OCNN–HMLSTM.

Parameter variation/Metrics	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	FPR (%)	FNR (%)	Training time (s)
Hidden nodes = 20, learning rate = 0.01	90.1	86.23	94.97	91.23	8.9	6.1	4115
Hidden nodes = 20, learning rate = 0.1	90.3	85.99	95.1	91.23	8.91	5.98	4553
Hidden nodes = 20, learning rate = 0.5	90.28	86.34	95.08	91.23	8.95	6.05	3900
Hidden nodes = 60, learning rate = 0.01	90.13	86.31	95.17	91.36	8.97	6.01	4946
Hidden nodes = 60, learning rate = 0.1	90.26	86.67	95.14	91.4	8.97	5.89	5356
Hidden nodes = 60, learning rate = 0.5	90.44	86.45	95.18	91.1	8.94	5.93	4871
Hidden nodes = 80, learning rate = 0.01	90.41	86.21	95.09	91.33	8.91	5.8	4557
Hidden nodes = 80, learning rate = 0.1	**90.67**	**86.71**	**95.19**	**91.46**	**8.86**	**5.78**	5118
Hidden nodes = 80, learning rate = 0.5	9.55	86.7	95.09	91.45	8.96	5.81	4189
Hidden nodes = 120, learning rate = 0.01	90.39	86.71	95.03	91.4	8.97	5.93	5045
Hidden nodes = 120, learning rate = 0.1	90.61	86.26	95.07	91.38	8.99	5.95	5456
Hidden nodes = 120, learning rate = 0.5	90.64	86.43	95.11	91.44	8.98	5.82	5732

significantly. The effect of varying each parameter is demonstrated briefly in this section. Two parameters namely hidden nodes and learning rate is varied to obtain differing results for the proposed unified model. The MCC and Kappa coefficient parameters are not considered in this section since the variation of these parameters is very minimal. The NSL-KDD dataset is used for this evaluation and the obtained results are shown in Table 12. The hidden nodes are varied from 20 to 120 while the learning rate is set from 0.01 to 0.5.

From Table 12, it is evident that the performance of OCNN–HMLSTM has been comparatively good when the number of hidden nodes = 80 and learning rate = 0.1. Although the training time is not low, it is even not too high. The performance results are also varied only by a small margin, thus the comparative performance of the proposed OCNN–HMLSTM is stable. The accuracy values have been consistently been around 90%, thus setting the threshold for a good/almost perfect classifier. These results illustrate that the proposed OCNN–HMLSTM, through the significantly advanced learning of both spatial and time features of network traffic data, ensures superior intrusion detection than most existing models. Also, the evaluation of three datasets depicts that this model is supportive of different network traffic data.

5. Conclusion

An efficient intrusion detection model has been developed in this paper using the proposed unified DL approach of OCNN–HMLSTM. This model does not need separate feature engineering techniques as CNN and LSTM in the unified approach performs the feature extraction process. The proposed OCNN extracts the spatial features while the HMLSTM extracts the temporal features and classifies the network data. The performance of this model has been evaluated over NSL-KDD, ISCX-IDS and UNSWNB15. The outcomes justified that the OCNN–HMLSTM model achieved improved intrusion detection in all three datasets with effective detection of multiple attacks. It increased the accuracy and reduces the FPR and FNR values significantly. The classification is also effective with higher positive values of MCC and Kappa coefficients. It is concluded that the OCNN–HMLSTM based IDS model is effective for intrusion detection through the automatic learning of spatial–temporal features to outperform the existing IDS models.

The uncertainty evaluation of the proposed unified model has shown it has less uncertainty that has an almost negligible impact on the performance. However, it may not stand the same when considering the other datasets where the training data is not proper. The proposed approach will be further evaluated in the future to adapt to other public intrusion detection datasets with uncertainty. There are some works in literature studies that have achieved higher detection accuracy than the OCNN–HMLSTM model for NSL-KDD data through optimal feature selection. This direction will also be investigated. Moreover, in the real world, the intrusions are very less compared to the normal records. This difference in malicious traffic records will be examined to improve the IDS model.

CRediT authorship contribution statement

P Rajesh Kanna: Conceptualization, Data curation, Formal analysis, Methodology, Investigation, Writing - original draft, Writing - review & editing, Carrying out additional analyses. **P Santhi:** Formal analysis, Methodology, Investigation, Supervision, Validating the ideas, Carrying out additional analyses, Reviewing the paper.

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.

References

[1] A.S. Ashoor, S. Gore, Importance of intrusion detection system (IDS), Int. J. Sci. Eng. Res. 2 (1) (2011) 1–4.

[2] T.S. Sobh, Wired and wireless intrusion detection system: Classifications, good characteristics and state-of-the-art, Comput. Stand. Interf. 28 (6) (2006) 670–694.
[3] N. Hubballi, V. Suryanarayanan, False alarm minimization techniques in signature-based intrusion detection systems: A survey, Comput. Commun. 49 (2014) 1–17.
[4] M. Gyanchandani, J. L.Rana, R.N. Yadav, Taxonomy of anomaly-based intrusion detection system: a review, Int. J. Sci. Res. Publ. 2 (12) (2012) 1–13.
[5] Y. Yang, K. McLaughlin, S. Sezer, Y.B. Yuan, W. Huang, Stateful intrusion detection for IEC 60870-5-104 SCADA security, in: 2014 IEEE PES General Meeting| Conference and Exposition, 2014, pp. 1–5.
[6] Z. Li, A. Das, J. Zhou, Usaid: Unifying signature-based and anomaly-based intrusion detection, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, Berlin, Heidelberg, 2005, pp. 702–712.
[7] P. Owezarski, On the impact of DoS attacks on Internet traffic characteristics and QoS, in: Proceedings 14th International Conference on Computer Communications and Networks, 2005. ICCCN 2005. (2005), pp. 269-274.
[8] M.C. Belavagi, B. Muniyal, Performance evaluation of supervised machine learning algorithms for intrusion detection, Procedia Comput. Sci. 89 (2016) (2016) 117–123.
[9] E.M. Kakihata, H.M. Sapia, R.T. Oiakawa, D.R. Pereira, J.P. Papa, V.H.C. De Albuquerque, F.A. Da Silva, Intrusion detection system based on flows using machine learning algorithms, IEEE Latin Amer. Trans. 15 (10) (2017) 1988–1993.
[10] H. Wang, J. Gu, S. Wang, An effective intrusion detection framework based on SVM with feature augmentation, Knowl.-Based Syst. 136 (2017) 130–139.
[11] M. Usha, P. Kavitha, Anomaly-based intrusion detection for 802.11 networks with optimal features using an SVM classifier, Wirel. Netw. 23 (8) (2017) 2431–2446.
[12] W. Meng, W. Li, L.F. Kwok, Design of intelligent KNN-based alarm filter using knowledge-based alert verification in intrusion detection, Secur. Commun. Netw. 8 (18) (2015) 3883–3895.
[13] S. Mukherjee, N. Sharma, Intrusion detection using naive Bayes classifier with feature reduction, Proc. Technol. 4 (2012) 119–128.
[14] N. Farnaaz, M.A. Jabbar, Random forest modelling for network intrusion detection system, Procedia Comput. Sci. 89 (1) (2016) 213–217.
[15] P. Mishra, V. Varadharajan, U. Tupakula, E.S. Pilli, A detailed investigation and analysis of using machine learning techniques for intrusion detection, IEEE Commun. Surv. Tutor. 21 (1) (2018) 686–728.
[16] A. Shenfield, D. Day, A. Ayesh, Intelligent intrusion detection systems using artificial neural networks, ICT Express 4 (2) (2018) 95–99.
[17] B.G. Atli, Y. Miche, A. Kalliola, I. Oliver, S. Holtmanns, A. Lendasse, Anomaly-based intrusion detection using extreme learning machine and aggregation of network traffic statistics in probability space, Cogn. Comput. 10 (5) (2018) 848–863.
[18] R. Singh, H. Kumar, R.K. Singla, An intrusion detection system using network traffic profiling and online sequential extreme learning machine, Expert Syst. Appl. 42 (22) (2015) 8609–8624.
[19] J. Gao, S. Chai, B. Zhang, Y. Xia, Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis, Energies 12 (7) (2019) 1223.
[20] Z.X. Yang, X.L. Qin, W.R. Li, Y.J. Yang, A ddos detection approach based on CNN in cloud computing, in: Applied Mechanics and Materials, Vol. 513, Trans Tech Publications Ltd, 2014, pp. 579–584.
[21] L.O. Anyanwu, J. Keengwe, G.A. Arome, Scalable intrusion detection with recurrent neural networks, in: 2010 Seventh International Conference on Information Technology: New Generations, 2010, pp. 919–923.
[22] R.C. Staudemeyer, Applying long short-term memory recurrent neural networks to intrusion detection, S. Afr. Comput. J. 56 (1) (2015) 136–154.
[23] C. Yin, Y. Zhu, J. Fei, X. He, A deep learning approach for intrusion detection using recurrent neural networks, Ieee Access 5 (2017) 21954–21961.
[24] J. Kim, J. Kim, H. Kim, M. Shim, E. Choi, CNN-based network intrusion detection against denial-of-service attacks, Electronics 9 (6) (2020) 916.
[25] M.T. Nguyen, K. Kim, Genetic convolutional neural network for intrusion detection systems, Future Gener. Comput. Syst. 113 (2020) 418–427.
[26] S.A. Althubiti, E.M. Jones, K. Roy, LSTM for anomaly-based network intrusion detection, in: 2018 28th International Telecommunication Networks and Applications Conference (ITNAC, IEEE, 2018, pp. pp. 1–3.
[27] A. Chawla, P. Jacob, B. Lee, S. Fallon, Bidirectional LSTM autoencoder for sequence-based anomaly detection in cyber security, Int. J. Simul. Syst. Sci. Technol. 20 (5) (2019) 7.1–7.6.
[28] M. Amar, B.E. Ouahidi, Weighted LSTM for intrusion detection and data mining to prevent attacks, Int. J. Data Mining, Modell. Manage. 12 (3) (2020) 308–329.
[29] M.A. Khan, M. Karim, Y. Kim, A scalable and hybrid intrusion detection system based on the convolutional-LSTM network, Symmetry 11 (4) (2019) 583.
[30] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, M. Zhu, HAST-IDS: Learning hierarchical spatial–temporal features using deep neural networks to improve intrusion detection, IEEE Access 6 (2017) 1792–1806.
[31] J. Zhang, Y. Ling, X. Fu, X. Yang, G. Xiong, R. Zhang, Model of the intrusion detection system based on the integration of spatial–temporal features, Comput. Secur. 89 (2020) 101681.
[32] J. Feng, Y. Liu, L. Wu, Bag of visual words model with deep spatial features for geographical scene classification, Comput. Intell. Neurosci. (2017) (2017).
[33] M.A. Hogo, Temporal analysis of intrusion detection, in: 2014 International Carnahan Conference on Security Technology (ICCST), 2014, pp. 1–6.
[34] M. Jenckel, S. S.Bukhari, A. Dengel, Training LSTM-RNN with imperfect transcription: limitations and outcomes, in: Proceedings of the 4th International Workshop on Historical Document Imaging and Processing, (2017), pp. 48-53.
[35] R. Chai, A. Tsourdos, A. Savvaris, Y. Xia, S. Chai, Real-time re-entry trajectory planning of hypersonic vehicles: a two-step strategy incorporating fuzzy multi-objective transcription and deep neural network, IEEE Trans. Ind. Electron. 67 (8) (2019) 6904–6915.
[36] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, C.P. Chen, Six-DOF spacecraft optimal trajectory planning and real-time attitude control: a deep neural network-based approach, IEEE Trans. Neural Netw. Learn. Syst. 31 (11) (2019) 5005–5013.
[37] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, C.P. Chen, Design and implementation of deep neural network-based control for automatic parking maneuver process, IEEE Trans. Neural Netw. Learn. Syst. (2020).
[38] S.J. Liu, Y. Yang, Y.Q. Zhou, A swarm intelligence algorithm-lion swarm optimization, Pattern Recogn. Artif. Intell. 31 (5) (2018) 431–441.
[39] G. Serpen, E. Aghaei, Host-based misuse intrusion detection using PCA feature extraction and KNN classification algorithms, Intell. Data Anal. 22 (5) (2018) 1101–1114.
[40] M. Wazid, A.K. Das, An efficient hybrid anomaly detection scheme using K-means clustering for wireless sensor networks, Wirel. Pers. Commun. 90 (4) (2016) 1971–2000.
[41] Y.Y. Aunga, M.M. Min, An analysis of k-means algorithm-based network intrusion detection system, Adv. Sci. Technol. Eng. Syst. J. 3 (1) (2018) 496–501.
[42] K. Peng, V. Leung, L. Zheng, S. Wang, C. Huang, T. Lin, Intrusion detection system based on decision tree over big data in fog environment, Wirel. Commun. Mobile Comput. (2018) (2018).
[43] E. Besharati, M. Naderan, E. Namjoo, LR-HIDS: logistic regression host-based intrusion detection system for cloud environments, J. Ambient Intell. Human. Comput. 10 (9) (2019) 3669–3692.
[44] S. Teng, N. Wu, H. Zhu, L. Teng, W. Zhang, SVM-DT-based adaptive and collaborative intrusion detection, IEEE/CAA J. AutomaticaSinica 5 (1) (2017) 108–118.
[45] P. Tao, Z. Sun, Z. Sun, An improved intrusion detection algorithm based on GA and SVM, Ieee Access 6 (2018) 13624–13631.
[46] B. Zhang, Z. Liu, Y. Jia, J. Ren, X. Zhao, Network intrusion detection method based on PCA and Bayes algorithm, Secur. Commun. Netw. (2018) (2018).
[47] M. Mazini, B. Shirazi, I. Mahdavi, Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and adaboost algorithms, J. King Saud Univ. Comput. Inform. Sci. 31 (4) (2019) 541–553.
[48] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, A. Alazab, A novel ensemble of hybrid intrusion detection system for detecting internet of things attacks, Electronics 8 (11) (2019) 1210.
[49] M.M. Baig, M.M. Awais, E.S.M. El-Alfy, A multiclass cascade of artificial neural network for network intrusion detection, J. Intell. Fuzzy Systems 32 (4) (2017) 2875–2883.
[50] I. Sumaiya Thaseen, J. SairaBanu, K. Lavanya, M. RukunuddinGhalib, K. Abhishek, An integrated intrusion detection system using correlation-based attribute selection and artificial neural network, Trans. Emerg. Telecommun. Technol. (e4014) (2020).
[51] Y. Zhao, Y. Shen, J. Yao, Recurrent neural network for text classification with hierarchical multi-scale dense connections, in: IJCAI, 2019, pp. 5450–5456.
[52] Y. Gal, Z.Z. Ghahramani, Dropout as a Bayesian approximation: Representing model uncertainty in deep learning, in: International Conference on Machine Learning, PMLR, 2016, pp. 1050–1059.