INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Hyper-Heuristic Firefly Algorithm Based Convolutional Neural Networks for Big Data Cyber Security

Rajan Aswanandini[1,2], **Chandran Deepa**[3]*

1 Assistant Professor, Department of Computer Science, KG College of Arts and Science, Coimbatore, 641037
2 Ph.D Scholar, Sri Ramakrishna College of Arts and Science, Coimbatore, 641006
3 Associate Professor, Department of Information Technology, Coimbatore, 641006

OPEN ACCESS

Received: 02.08.2021
Accepted: 17.10.2021
Published: 18.11.2021

Citation: Aswanandini R, Deepa C (2021) Hyper-Heuristic Firefly Algorithm Based Convolutional Neural Networks for Big Data Cyber Security. Indian Journal of Science and Technology 14(38): 2934-2945. https://doi.org/10.17485/IJST/v14i38.1401

* **Corresponding author**.

deepapkd@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: A highly accurate Intrusion detection model is developed that classifies both the network-based and host-based intrusions without any complexity issues. Method: An optimized Deep Learning (DL) algorithm of IDS model is presented in the form of a Hyper-Heuristic Firefly Algorithm based Convolutional Neural Networks (HHFA-CNN). This proposed HHFA-CNN reduces false values and improves accuracy without increasing the complexities. Findings: The proposed HHFA-CNN system is performed on two network traffic datasets: NSL-KDD and ISCX-IDS. The outcomes demonstrated that the proposed HHFA-CNN model gives predominant execution than the other existing models. Novelty: The proposed model has employed a novel Hyper-Heuristic Firefly Algorithm for optimizing the hyper-parameters of the CNN. This model maintains the standard guidelines of the firefly algorithm and applies the high-level technique for controlling the exploration and determination of low-level heuristics.

Keywords: Big data; Cyber security; Intrusion detection system; Hyper-Heuristic Firefly Algorithm; Convolutional Neural Networks

1 Introduction

With the introduction of advanced technologies in the recent years, the big data analytics have attained significant interest in various domain applications such as medicine, healthcare, education, smart cities, environment analytics, business analytics, data processing and cyber security [(1)]. As most businesses are operated over the internet in the modern big data age, the cybercrimes are not only limited to hacking the business information but has also intruded towards the common man. The increasingly complex as well as intelligent cyber threats have resulted in massive destructions [(2)]. Cyber security has become a vital part of any domain and has attracted big investments to protect the data and also the organization systems [(3)]. The conventional cyber security systems cannot detect all advanced threats such as the denial of service, spoofing, brute force, and SQL injection, etc. Also, these systems face difficulties in handling both the system operations and threat analysis together as a huge amount of security

data increases complexity. These limitations triggered the search for advanced cyber security systems. As the huge amount of security data can be handled effectively by big data technologies, a cyber-security analytics system has been formed by combining big data analytics and cyber security [4]. The new technology is termed as Big Data Cyber security Analytics (BDCA) which is a system for collecting, storing and analyzing large volumes of security data to protect the network against the malicious threats [5]. It usually includes the security systems such as intrusion detection system (IDS), botnet detector, phishing detector and malware detectors. The BDCA has gained the trust of many companies who have sought the usage of it to protect the business data from the hackers and malicious intruders.

The most common duty of BDCA is to monitor the network and internet traffic to analyze the intrusions. The intrusion detection is considered as a fundamental security solution as the intrusions pave the way for other malicious events. The malicious cyber-attacks lead to serious security degradation and hence the research community has insisted on the requirements of a novel, adaptive and reliable IDS. Depending upon the detected intrusion behaviors, the IDS are classified as network-based IDS (NIDS) [6] and host-based IDS (HIDS) [7]. For detecting the intrusions, the NIDS analyses the packet data in the network traffic flows while the HIDS analyses the log information such as sensor logs, system logs, file systems, disk information, user account files, etc. Although the NIDS is widely used in many fields, many organizations prefer using both NIDS and HIDS together. Likewise, the network traffic flow data can be analyzed using misuse detection, anomaly detection and stateful protocol analysis [8]. These intrusion detection strategies have been reliant on different aspects of the network traffic flows. The misuse detection employs the signatures predetermined by the users stored in the signature databases to analyze and filter the attacks. The anomaly detection employs the heuristic strategies to detect the unknown attack activities [9]. Stateful protocol analyses are the powerful analysis strategy than the misuse and anomaly detection as it works on the network, applications and transport layers to detect the malicious activates. Most studies have employed machine learning-based anomaly detection methods [10] as they detect maximum attack types including the unknown possibilities from both the NIDS and HIDS. However, it has a high false-positive rate than misuse detection. Hence the recent studies have suggested the use of hybrid models incorporating the misuse and anomaly detection to reduce the false-positive values. It is also found that the deep learning approaches have provided a high detection rate with fewer errors due to their abilities to learn deep features at the network level as well as host-levels [11]. But these approaches also suffer from limitations of non-generalization for publicly available datasets. Most approaches are suitable only for single dataset while provides poor performance on another dataset. Based on these suggestions and inferences, it is summarized that the high false-positive rate in machine learning and non-generalization in deep learning has negatively impacted the NIDS and HIDS.

This paper has suggested the use of optimized deep learning algorithm for accurately identifying the attacks in the network flow data with less false positive rate and less complexity. Previously, Hyper-Heuristic Improved Particle Swarm Optimization based Support Vector Machines (HHIPSO-SVM) [12] has been developed to tackle big data cyber security problems. However, it has limitations in handling big data complexity when NIDS and HIDS are collaboratively combined. Hence, in this paper, the Convolutional Neural Networks (CNN) has been selected as the deep learning classifier to reduce the complexity when applied to NIDS and HIDS. CNN configuration is optimally determined by defining its hyper-parameters through the novel Hyper-Heuristic Firefly Algorithm (HHFA).

The contributions of this paper are summarized as follows:

- The Natural Language Processing (NLP) Text Representation methods are used to process the log files to determine the host-level events. As NLP based text representation methods identify the contextual and semantic similarity from a large amount of unstructured and fragmented texts, it enhances the detection accuracy of the IDS model.
- A scalable IDS framework has been developed using an effective deep earning approach of HHFA-CNN to handle the deep characteristics of network-level and host-level events. The collaborative combination of NIDS and HIDS increases the complexity and hence the proposed deep learning HHFA-CNN is introduced in this paper.
- The proposed HHFA-CNN based IDS model is applied on benchmark datasets of NIDS and HIDS for conducting the experimental comparisons.

Recent studies have employed different types of deep learning algorithms and ensemble approaches for big data analytics-based intrusion detection. To compete with such IDS approaches, machine learning algorithms were predominantly employed using optimization algorithms. Sabar et al. [13] presented bi-objective hyper-heuristic support vector machines (HH-SVM) in which the SVM configuration was optimally selected. They have tested NSL-KDD and BIG 2015 datasets and achieved 85.68% accuracy. However, the modelling of SVM configuration as a bi-objective problem considers only accuracy and model complexity factors. This reduces the performance efficiency of the approach. In another similar approach, Safaldin et al. [14] also developed IDS approach using SVM optimized by improved binary grey wolf optimizer (SVM-IBGWO). It was applied on NSL-KDD and achieved 96% detection rate and 96% accuracy but took 69.6 hours. It might be because of the use of standard,

benchmark feature selection algorithms. Extreme Learning Machine has been used as an advanced machine learning algorithm for classification problems. Lv et al.[15] have developed advanced IDS model using optimal hybrid kernel ELM (HKELM). Initially, the hybrid kernel is designed and the hybrid optimization of gravitational search algorithm (GSA) and differential evolution (DE) algorithm were used to optimize HKELM. Additionally, the kernel principal component analysis (KPCA) was used for dimensionality reduction. It was tested on three datasets with the accuracy of 96.69% on KDD99, 89% on UNSW-NB15 dataset and 95.82% on TE intrusion dataset. However, this approach results in higher computational complexity which has also increased the processing time.

Due to the limitations in machine learning approaches including the ELM, the researchers have started employing deep learning algorithms for the big data cyber security models. Lopez-Martin et al.[16] proposed the use of four different versions of deep reinforcement learning (DRL) for IDS tested on NSL-KDD and AWID datasets. This supervised machine learning algorithms of DRL have improved the accuracy of detection while reducing the false positives. Double Deep Q-Network (DDQN) has achieved 89.78% and 95.7% accuracy over NSL-KDD and AWID, respectively. However, the Q-function and policy functions of DRL has limitations that cannot adapt to the multi-agent concept effectively.

Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) neural networks have achieved maximum exposure and increased classification accuracy in IDS models. Xiao et al.[17] presented an IDS model based on CNN and feature reduction technique. In this model, the redundant features are removed using dimensionality reduction and then the CNN classifies the normal data and attacks. Applied on KDD99 dataset, this IDS model achieved 96%. But the detection rates of U2R and R2L attack classes are very low at 20.61% and 18.96% which is the biggest drawback of this model. Riyaz & Ganapathy[18] developed a deep learning-based IDS approach using CNN in which the features are selected using a conditional random field and linear correlation coefficient. This approach increases the detection accuracy to 98.8% and reduced false alarm rate below 1% when applied KDD99 dataset. Although efficient, the data communication speed in this approach is very less so that the time complexity for making effective decisions. Li et al.[19] presented a robust IDS model using multi-CNN fusion algorithm. In this model, four CNNs are used for classifying the featured dataset based one-dimensional features and the best of the CNN result is obtained. This multi-CNN achieved 86.95% training and 76.67% testing accuracies for binary classification and 81.33% training and 64.81% testing accuracies for multiclass classification of NSL-KDD dataset with less complexity. However, this model has a higher time complexity than the single CNN due to the large feature set processing. Nguyen & Kim[20] developed an optimized CNN using Genetic algorithm (GA-CNN). The deep features are selected by the CNN whose structure is selected optimally using the GA. Evaluation of NSL-KDD showed that the GA-CNN has achieved 96.2% detection accuracy. However, the exhaustive search process of GA increases the time consumption. Also, this approach has limitations in handling the imbalanced datasets.

Almiani et al.[21] developed an IDS model using deep recurrent neural network (DRNN) on NSL-KDD dataset. It detected maximum attack types with 92.18% accuracy and above 84% classification coefficients. However, it has a high false-positive rate of 9.8% compared to other existing models. Kasongo & Sun[22] designed a deep long short-term memory (DLSTM) based IDS approach for wireless networks. DLSTM provided accuracy of 99.51% and 86.99% over the validation and test data, respectively of NSL-KDD. Kasongo & Sun[23] also introduced another deep learning method-based IDS using feed-forward Deep Neural Network (FFDNN) with wrapper-based feature extraction. This method was tested over UNSW-NB15 and achieved accuracies of 87.10% and 77.16% for the binary and multiclass classifications. Similarly, this method was also applied on AWID dataset with accuracies of 99.66% and 99.77% for the binary and the multiclass classifications. Although efficient, both these methods did not effectively investigate the individual attack class detection rates.

Irrespective of their advantages, CNN and LSTM have limitations in learning the spatial and temporal features. Hence some studies have combined them for increasing their effectiveness. Khan et al.[24] proposed a scalable and hybrid IDS approach using convolutional-LSTM (Conv-LSTM). This approach integrated the benefits of CNN and LSTM and increased the accuracy of classification. Evaluated on ISCX-UNB dataset showed that this Conv-LSTM approach achieved 97.29% intrusion detection accuracy and also reduced computational complexity. However, the evaluation of only a single dataset is considered a drawback. Hsu et al.[25] developed a robust IDS approach using LSTM based CNN (CNN-LSTM) for classifying all the network traffic data of NSL-KDD. This CNN-LSTM based IDS approach increased the accuracies for both training and testing datasets with 98.64% and 96.47% achieved for two categories. However, this approach took longer computational time than the other deep learning algorithms. Zhang et al.[26] presented the IDS model by integrating the spatial-temporal features using Multi-scale CNN with Long Short-Term Memory (MSCNN-LSTM). The MSCNN learns the spatial features while the LSTM extracts the temporal features. This method was evaluated on UNSW-NB15 dataset and achieved 89.8% accuracy. However, this method cannot handle the data imbalance problems.

Although the combination of CNN and LSTM has provided better classification accuracy, their main drawback is the computational complexity. In some studies, the class imbalance problem is also cited as a limitation. From the literature, it

has been found that the optimized CNN has provided significantly better accuracy and also has less complexity. Hence this study focuses on exploring the optimized CNN and suggests the use of advanced optimization algorithms to overcome the limitations in the GA search process.

2 HHFA-CNN BASED IDS METHODOLOGY

The proposed HHFA-CNN methodology includes the hyper-heuristic modelling of the firefly algorithm for elevating the hyper-parameters of CNN to attain the best structural design of CNN. Figure 1 shows the HHFA-CNN structure diagram. The structure diagram consists of Convolution Layer (CL), Pooling Layer (PL), and Fully Connected Layer (FCL). Along with these three parameters, the hidden layer units and the type of pooling are optimally selected by the HHFA whose components are also shown in Figure 1. The proposed HHFA is the integration of hyper-heuristics with the multi-objective firefly algorithm.

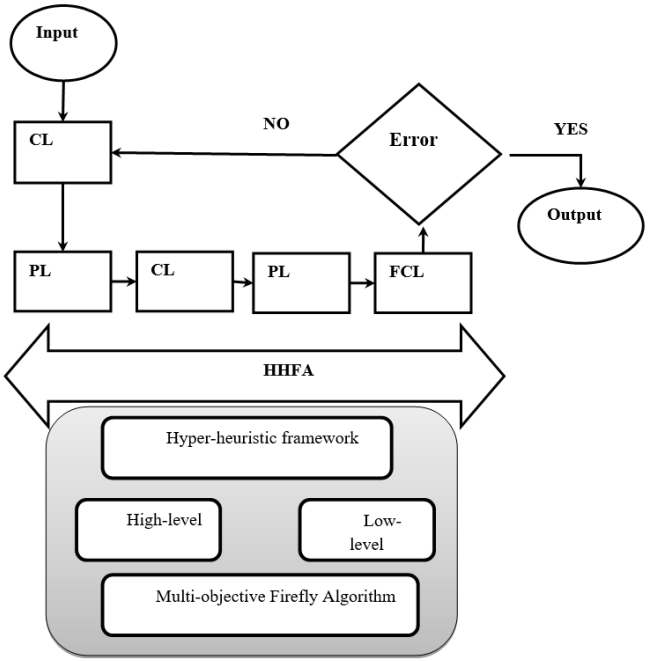

Fig 1. HHFA-CNN structure diagram

2.1 CNN

CNN comprises four main operators namely Convolution layer, 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 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 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 1 which will be optimally selected using HHFA.

Here N_c denotes the maximum number of convolutional layers.

Table 1. Hyper-parameter value ranges in CNN

Hyper-parameter	Range	Difference
Number of CL	1-4	1
Number of PL	1-N_c	1
Number of FCL	1-5	1
Hidden units/layer	256-1024	256
Pooling type	1-4	1
Kernel size	1-8	1

2.2 HHFA

The HHFA is developed by fusing the hyper-heuristics to the multi-objective firefly algorithm. The hyper-heuristic framework consists of two strategies namely high-level and low-level strategies for enhancing the optimization function of the firefly algorithm. The low-level strategy explores the problem and forms the rules to select the solutions. Then one or more solutions is considered, combined or modified to form a new set of solutions to increase better options. The high-level strategy initiates the heuristic search process to select the solutions from the set of possible solutions based on the rules generated by the low-level strategy.

The low-level heuristics contains the set of problem-related rules generated to provide solutions to each selected problem instances. It forms a new set of solutions by considering one or more solutions and transforming or combining them using different search processes. In this study, the FA based search process is used as one of the search processes to generate new solutions. Once the new solutions are formed, the high-level strategy imitates the selection process. The high-level strategy automatically performs the heuristic selection by choosing the heuristics one-by-one and applying it to the solutions. From the existing set of heuristics formed by the rules generated by low-level strategy, the heuristics are selected through an online heuristic selection mechanism. The empirical reward and the confidence level variables are the main metrics for measuring the efficiency of the heuristics. The rewards obtained in past performance are called empirical reward while the frequency of utilization of the heuristic denotes the confidence level. Using these two variables, the heuristics is deemed fit or unfit for the current state of operation. Thus selected heuristics are applied to the solutions through the firefly foraging process[27].

The heuristics are initialized as the population of fireflies $x_i \quad (i = 1, 2, \ldots, z)$. When the brightest firefly at location x denotes the best solution, it can be formed as the maximization problem $I(x) \sim f(x)$, where $I(x)$ is the light intensity and $f(x)$ is the fitness function. In this study, the error rate is to be used as the fitness and hence the problem is converted into minimization problem as

$$I(x) = \begin{cases} \frac{1}{f(x)}, \text{ if } f(x) > 0 \\ 1 + |f(x)|, \text{ otherwise} \end{cases} \quad (1)$$

The light intensity $I(x)$ and the attractiveness function (β) are the basis of this algorithm. When the light intensity and attractiveness increases, the distance (r) between the source firefly and destination firefly decreases.

$$I(r) = \frac{I_0}{1+\gamma r^2} \tag{2}$$

Here I_0 is the source light intensity and γ is the absorption coefficient. This can be approximated by the Gaussian form.

$$I(r) = I_0 exp(-\gamma r^2) \tag{3}$$

The attractiveness β is proportionate to the light intensity seen by the adjacent fireflies. Hence the attractiveness β is given as proportional to the light intensity of the solutions

$$\beta(r) = \beta_0 exp(-\gamma r^m) \tag{4}$$

Where β_0 is the attractiveness at r=0 and m is the number of iterations.

In CNN optimization, the computational complexity must be reduced which means the resource utilization must be less. So, the attractiveness expression is modified for the practical application as given below

$$\beta(r) = \frac{\beta_0}{1+\gamma r^2} \tag{5}$$

The distance amongst any two fireflies (nodes) i and j positioned at x_i and x_j is denoted as $r_{i,j}$, which is computed as the Cartesian distance measure.

$$r_{i,j} = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{6}$$

Where $x_{i,k}$ and $x_{j,k}$ are the Cartesian points of x_i and x_j and d is the number of dimensions.

The firefly moves towards the best firefly and this location is updated after each iteration using the following equation

$$x_i = x_i + \beta_0 e^{-\gamma r^2}(x_j - x_i) + \alpha(rand - \frac{1}{2}) \tag{7}$$

Here *rand* is the pseudo-random number in the range [0, 1] and α is the step-size controlling parameter. For best outcomes, the values of α and β_0 are adjusted to $\alpha \in [0,1]$ and $\beta_0 = 0.2$.

The heuristic is applied to each of the solutions obtained by the firefly based on the light intensity and the attractiveness of the firefly algorithm. The firefly which is returned as the global best solution contains the solution to be applied. The heuristic is applied with the selected solution to form a new set of solutions. In this stage, the serial scheduling and double justification are used. Serial scheduling is used to select the solutions without interleaving the feasible solutions. Likewise, the double justification is a simple local search technique which searches the solutions with exacting shifting to control the search quality. The new solutions are compared, and then they are analysed by their properties. This analysis in terms of configuration determines whitener to include them in the existing set of solutions or terminate them to accommodate newer solution from next iterations.

After the formation of new solutions by low-level heuristics and the selection by the high-level strategy, they are saved in the non-dominated set of solutions in the archive. The non-dominated sorting procedure is used to classify the archive to create several levels for saving the newer solutions. The first level is given to the solution with high priority and the next level will be given to the second-best priority and vice versa. The HHFA selects the solutions from this archive based on the Pareto-front and returns the best configuration as the final solution. Algorithm 1 summarizes the steps involved in HHFA.

Algorithm 1: HHFA

```
Begin
Initialize population of fireflies x_i, position vector of each firefly
Assign heuristics as fireflies
The light intensity I at x_i is found via I(r)
Set light absorption coefficient γ
Evaluate the fireflies to determine the fitness f(x)
While (m < Max_Generation)
    For i=1:n all n fireflies
        For j=1:i all n fireflies
        Call the j-th low-level heuristics of the firefly search space
        Apply serial scheduling and double justification
        If (I_j > I_i),
            Move firefly i towards j in d-dimension;
        End if
        Estimate new solutions and update light intensity
        Update the location of fireflies
        End for j
    End for i
    Check the stopping criteria
    Update the firefly ranking list to determine current best
End while
Return best firefly
End process
```

2.3 HHFA-CNN

The CNN architecture is represented as $h = \{h_{cl},\ h_{pl}, h_{fcl}\}$, where h is the set of structural parameters which consists of the parameter annotations of convolutional (h_{cl}), pooling (h_{pl}) and fully-connected layers (h_{fcl}). The set of convolutional parameters h_{cl} is defined in the problem as $h_{cl} = \{C_1, C_2, \ldots, C_{a-1}\}$ where a denotes the number of convolutional layers and $C_i = (C_{count}, C_{size})$ represent the configuration tuple of the i-th layer in CL. C_{count} is the number of kernels per i-th layer and C_{size} is the kernel size of the i-th layer in CL. Likewise, $h_{pl} = \{p_1, p_2, \ldots, p_{b-1}\}$ denotes the set of PL and $h_{fcl} = \{s_1, s_2, \ldots, s_{n-1}\}$ denotes the set of FCL, with b and n representing the number of layers in PL and FCL, respectively. The size of each i-th layer in PL is denoted as p_i and that of FCL is denoted as s_i. Let H represent the set of possible CNN configurations and the objective will be to find the configuration $h \in H$ which provides minimum classification error. As this problem is an NP-hard optimization, the search space of HHFA is reduced by the upper and lower boundaries as given in Table 1.

For the optimal selection of the CNN hyper-parameters, each solution is made up of the problem parameters subject to optimization by the firefly search process of exploitation (intensification) and exploration (diversification). The exploitation in HHFA is controlled by the values assigned to β and γ. There are two cases of values for these parameters that have been implemented in previous studies. i) When the value of $\beta = \beta_0$ and $\gamma = 0$, the solutions move towards others with the largest step size since there are the maximal exploitation and minimal exploration. ii) When the value of $\beta = 0$ and $\gamma = \infty$, the solutions move in random steps since the exploration is maximum while the exploitation is absent altogether. This trade-off between the values of β and γ can be determined by adjusting their values based on practical experiments. The value of γ is said to be optimal when it is between 0.01 and 100. Likewise, $\beta = 0.2$ and $\alpha \in [0, 1]$ are considered for optimal performance.

The hyper-parameters problem is encoded as $h = \{h_{cl},\ h_{pl}, h_{fcl}\}$ using the HHFA with each solution representing a possible CNN configuration. The HHFA population is denoted as P at every t time and consists of n individual fireflies (solutions) such that $P^t = \{h_0,\ h_1, \ldots h_{n-1}\}$. At the initialization phase, the population of uniform distributed solutions is generated as

$$x_{i,j} = lb_{i,j} + \psi.(ub_j - lb_j) \tag{8}$$

Here $x_{i,j}$ denotes the j-th parameter of the i-th solution in the population, ψ is a pseudo-random integer between [0, 1], and lb_j and ub_j represent the lower and upper bounds of the j-th parameter, respectively. To reduce the computation time and search

space reduction, the values for the number of kernels is set as $lb_j = 1$ and $ub_j = 128$.

In this study, the hyper-parameters like dropout rate, the learning rate, etc. are not optimized as they mostly have real values. The hyper-parameters that provide the integer values are only optimized using HHFA. As the upper and lower bounds for each parameter are set high i.e. greater than 1, the equations (1) to (7) depicted in HHFA can be adaptively used for the CNN optimization problem. The classification error rate is used as the fitness function. The objective is to minimize the error rate while calculating the fitness for the i-th solution which can be expressed as

$$x_i = \frac{1}{1 + |error_i|} \tag{9}$$

Here $error_i$ denotes the classification error of the test dataset for the i-th solution. For testing purpose, the proposed HHFA-CNN is pled on the NSL-KDD dataset. The model is trained with k epochs for CNN to obtain different error rates. While many configurations for the CNN are found by the tuning process, HHFA selects the configuration with minimum error rate. The top four configurations for CNN obtained using HHFA are shown in Table 2.

Table 2. Best hyper-parameter configurations of CNN obtained using HHFA

Layer type	Configuration	Kernel size	Error rate
CNN configuration 1		1*1	16.3
CL	3 layers	2*2	16.7
PL	2 layers; max pooling	3*3	16.6
FCL	3 layers; 512 units		
CNN configuration 2		1*1	16.7
CL	3 layers	2*2	17.3
PL	2 layers; max pooling	3*3	17.2
FCL	3 layers; 256 units		
CNN configuration 3		1*1	16.8
CL	2 layers	2*2	17.1
PL	2 layers; max pooling	3*3	16.9
FCL	3 layers; 512 units		
CNN configuration 4		1*1	17.1
CL	3 layers	2*2	17.8
PL	3 layer; max pooling	3*3	17.5
FCL	2 layers; 1024 units		

The configurations are obtained such that the CL, PL, FCL are determined and the kernel size is varied to obtain the three different error rates. This CNN 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. Considering the configurations from the above table, the CNN configuration with less classification error is chosen by the HHFA. The best performance was obtained only after 13 to 18 iterations in all conducted HHFA runs. In this case, CNN configuration 1 has less classification error of 16.3 when the 1*1 kernel is used and hence it will become the optimal CNN architecture. This optimal CNN increases the classification of the intrusion datasets.

3 Results and Discussion

The assessment of the suggested HHFA-CNN prototype is achieved using two benchmark cases of cyber security problems, NSL-KDD and ISCX-IDS datasets. The tests are performed in MATLAB R2016b on a Windows 64 bit machine of processor Intel core i5 3470 3.2 GHz, RAM 4GB DDR3 and Storage of 500GB Intel SSD. The two benchmark instances are collected from https://www.unb.ca/cic/datasets/index.html.

The NSL-KDD consists of training, testing, 20% training and 20% testing data files. It also contains a subset file with difficulty levels. The NSL-KDD is an improved version of the popular KDDCUP99 dataset. NSL-KDD problem instance consists of 311,027 training samples and 77,289 testing samples which are classified as either normal or malicious. ISCX-IDS was created by monitoring the network activity for 7 days from Friday 11/6/2010 to Thursday, 17/6/2010. It consists of records of normal, HTTP Denial of Service attacks, Brute Force attacks and infiltration activities. Around 208,667 training samples and 78,400 testing samples that are classified as either normal or attack activities are used for this evaluation.

3.1 Performance evaluation

The proposed HHFA-CNN is implemented along with the existing HHIPSO-SVM[12] and HH-SVM[13] for performance comparison. The comparison metrics are accuracy, precision, recall, f-measure, and time. Table 3 shows the performance evaluation of HHFA-CNN compared to the current HHIPSO-SVM and HH-SVM on NSL-KDD for 25 independent runs.

Table 3. Performance comparison on NSL-KDD

Algorithm / Metrics	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	Time (seconds)
HH-SVM[13]	89.76	67.10	62.81	62.22	4.65
HHIPSO-SVM[12]	93.33	73.99	64.29	68.37	2.55
HHFA-CNN	96.6667	93.9394	74	82.7860	1.38
DT	80.14	72.33	61.25	85.12	5.62
FC	82.98	74	60.28	61.35	6.58
GNBT	80	69	70.23	76.52	5.35

It can be seen that the performance values of HHFA-CNN are higher than the HHIPSO-SVM and HH-SVM. HHFA-CNN has 96.6667% accuracy which is 3.3% and 6.9% higher than HHIPSO-SVM and HH-SVM. Likewise, HHFA-CNN has outperformed both HHIPSO-SVM and HH-SVM in terms of precision, recall and f-measure. HHFA-CNN has 20% and 26.9% high precision, 9.7% and 11.2% higher recall, 14.5% and 20.5% higher f-measure than the HHIPSO-SVM and HH-SVM models, respectively. The execution time taken by HHFA-CNN is also less than the HHIPSO-SVM and HH-SVM. Figure 2 shows the graphical plot of these results.

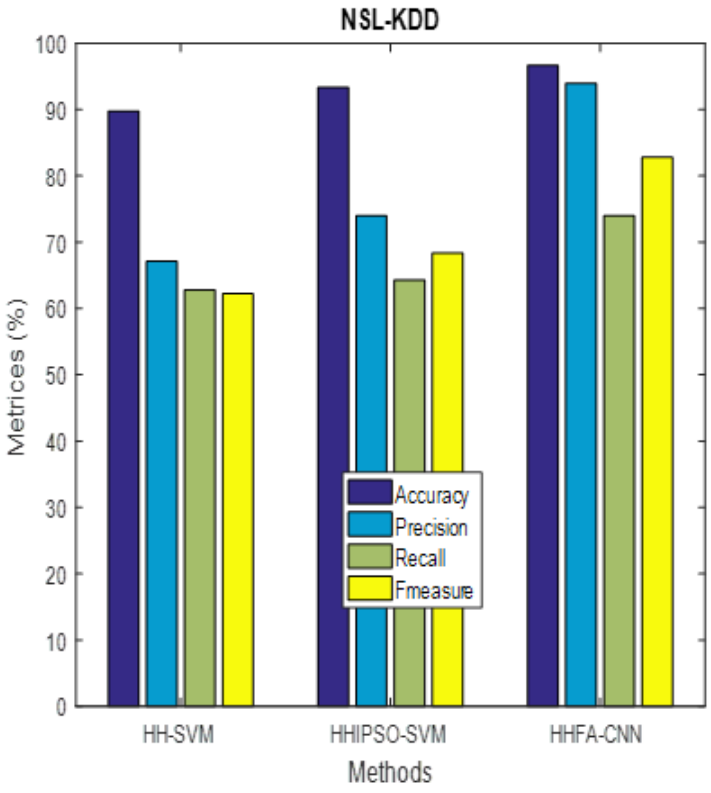

Fig 2. Performance comparison on NSL-KDD

Table 4 shows the performance comparison of HHFA-CNN against the existing HHIPSO-SVM and HH-SVM on ISCX-IDS testing dataset for 25 independent runs.

Table 4. Performance comparison on ISCX-IDS

Algorithm / Metrics	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)	Time (seconds)
HH-SVM [13]	86.6	63.3	60.0	56.19	126
HHIPSO-SVM [12]	92.4	69.65	61.1	59.82	49.5
HHFA-CNN	93.33	99.7	93.33	96.55	48.2

Similar to NSL-KDD, the performance obtained on ISCX-IDS shows that HHFA-CNN has outperformed the HHIPSO-SVM and HH-SVM models. HHFA-CNN has 0.97% and 6.7% higher accuracy, 30% and 36.3% higher precision, 32.2% and 33.3% higher recall, 36.7% and 40.4% higher f-measure than the HHIPSO-SVM and HH-SVM models,- respectively. HHFA-CNN also consumes 1.3 seconds and 77.8 seconds less than HHIPSO-SVM and HH-SVM models, respectively for executing the ISCX-IDS data. Figure 3 shows the graphical plot of these results.

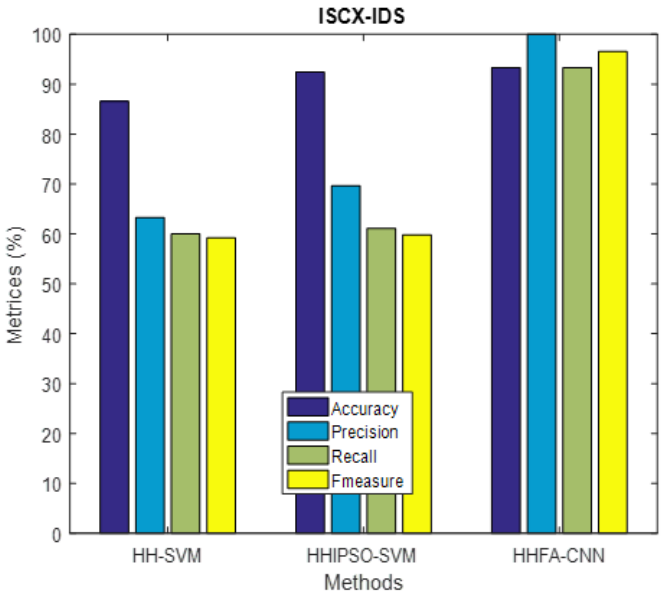

Fig 3. Performance comparison on ISCX-IDS

The performance of the proposed HHFA-CNN is also compared with other popular algorithms from the literature that were tested on NSL-KDD dataset. The accuracy values of the algorithms namely HH-SVM [13], SVM-IBGWO [14], DRL [16], Multi-CNN [19], GA-CNN [20], DRNN [21], DLSTM [22], CNN-LSTM [25], and HHIPSO-SVM [12] are compared in Table 5.

Table 5. Accuracy comparison of HHFA-CNN against models in the literature

Algorithm	Accuracy (%)
HH-SVM [13]	89.76
SVM-IBGWO [14]	96
DRL [16]	89.78
Multi-CNN [19]	86.95
GA-CNN [20]	98.2
DRNN [21]	92.18
DLSTM [22]	86.99
CNN-LSTM [25]	96.47
HHIPSO-SVM [12]	93.33
HHFA-CNN	**96.6667**

From Table 5, it can be inferred that the proposed HHFA-CNN framework has better performance than the other algorithms discussed in the literature. The main reason for this improvement is the use of better design strategy and efficient solution for different problem instances by selecting the optimal configuration of the CNN. The spatial features of the problem instances are deep learned by the optimized CNN and it has also positively impacted the performance.

4 Conclusion

In this study, a hyper-heuristic firefly optimization is intended for the improvement of the CNN design to determine the big data intrusion problems. In the first part, the CNN design issue is displayed as a multi-objective optimization issue dependent on the hyper-parameters. This problem is addressed by adopting the proposed HHFA structure which uses the high-level methodology and low-level heuristics of hyper-heuristic methodology on the standard firefly optimization. The proposed HHFA-CNN system was assessed on two network traffic datasets: NSL-KDD and ISCX-IDS. The outcomes demonstrated that the proposed HHFA-CNN model gives predominant execution than the other existing models. In the future, the proposed hyper-heuristic system can be used for multi-class attack detection. Also, other cyber security instances such as UNSW-NB15 will be tested. Moreover, the impact of feature dimension reduction techniques will also be investigated.

References

1) Hu H, Wen Y, Chua TS, Li X. Toward Scalable Systems for Big Data Analytics: A Technology Tutorial. *IEEE Access*. 2014;2:652–687. Available from: https://dx.doi.org/10.1109/access.2014.2332453.
2) Mahmood T, Afzal U. Security Analytics: Big Data Analytics for cybersecurity: A review of trends, techniques and tools. *2013 2nd National Conference on Information Assurance (NCIA)*. 2013;p. 129–134. doi:10.1109/NCIA.2013.6725337.
3) von Solms R, van Niekerk J. From information security to cyber security. *Computers & Security*. 2013;38:97–102. Available from: https://dx.doi.org/10.1016/j.cose.2013.04.004.
4) Petrenko SA, Makoveichuk KA. Big data technologies for cybersecurity. *CEUR Workshop*. 2017;p. 107–111.
5) Ullah F, Babar MA. Architectural Tactics for Big Data Cybersecurity Analytics Systems: A Review. *Journal of Systems and Software*. 2019;151:81–118. Available from: https://dx.doi.org/10.1016/j.jss.2019.01.051.
6) Ahmed M, Pal R, Hossain MM, Bikas MAN, Hasan MK. NIDS: A Network Based Approach to Intrusion Detection and Prevention. *2009 International Association of Computer Science and Information Technology - Spring Conference*. 2009;p. 141–144.
7) Deshpande P, Sharma SC, Peddoju SK, Junaid S. HIDS: A host based intrusion detection system for cloud computing environment. *International Journal of System Assurance Engineering and Management*. 2018;9(3):567–576. Available from: https://doi.org/10.1007/s13198-014-0277-7.
8) Yang Y, Mclaughlin K, Sezer S, Yuan YB, Huang W. Stateful intrusion detection for IEC 60870-5-104 SCADA security. *2014 IEEE PES General Meeting | Conference & Exposition*. 2014;p. 1–5. doi:10.1109/PESGM.2014.6939218.
9) Xu X. Sequential anomaly detection based on temporal-difference learning: Principles, models and case studies. *Applied Soft Computing*. 2010;10(3):859–867. Available from: https://doi.org/10.1016/j.asoc.2009.10.003.
10) da Costa KAP, Papa JP, Lisboa CO, Munoz R, de Albuquerque VHC. Internet of Things: A survey on machine learning-based intrusion detection approaches. *Computer Networks*. 2019;151:147–157. Available from: https://dx.doi.org/10.1016/j.comnet.2019.01.023.
11) Aleesa AM, Zaidan BB, Zaidan AA, Sahar NM. Review of intrusion detection systems based on deep learning techniques: coherent taxonomy, challenges, motivations, recommendations, substantial analysis and future directions. *Neural Computing and Applications*. 2020;32(14):9827–9858. Available from: https://dx.doi.org/10.1007/s00521-019-04557-3.
12) Aswanandini R, N M. Multi-Objective Hyper-Heuristic Improved Particle Swarm Optimization Based Configuration of Support Vector Machines for Big Data Cyber Security. *International Journal of Innovative Technology and Exploring Engineering*. 2019;8(12):3892–3897.
13) Sabar NR, Yi X, Song A. A Bi-objective Hyper-Heuristic Support Vector Machines for Big Data Cyber-Security. *IEEE Access*. 2018;6:10421–10431. Available from: https://dx.doi.org/10.1109/access.2018.2801792.
14) Safaldin M, Otair M, Abualigah L. Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks. *Journal of Ambient Intelligence and Humanized Computing*. 2021;12(2):1559–1576. Available from: https://dx.doi.org/10.1007/s12652-020-02228-z.
15) Lv L, Wang W, Zhang Z, Liu X. A novel intrusion detection system based on an optimal hybrid kernel extreme learning machine. *Knowledge-Based Systems*. 2020;195:105648. Available from: https://dx.doi.org/10.1016/j.knosys.2020.105648.
16) Lopez-Martin M, Carro B, Sanchez-Esguevillas A. Application of deep reinforcement learning to intrusion detection for supervised problems. *Expert Systems with Applications*. 2020;141:112963. Available from: https://dx.doi.org/10.1016/j.eswa.2019.112963.
17) Xiao Y, Xing C, Zhang T, Zhao Z. An Intrusion Detection Model Based on Feature Reduction and Convolutional Neural Networks. *IEEE Access*. 2019;7:42210–42219. Available from: https://dx.doi.org/10.1109/access.2019.2904620.
18) Riyaz B, Ganapathy S. A deep learning approach for effective intrusion detection in wireless networks using CNN. *Soft Computing*. 2020;24(22):17265–17278. Available from: https://dx.doi.org/10.1007/s00500-020-05017-0.
19) Li Y, Xu Y, Liu Z, Hou H, Zheng Y, Xin Y, et al. Robust detection for network intrusion of industrial IoT based on multi-CNN fusion. *Measurement*. 2020;154:107450. Available from: https://dx.doi.org/10.1016/j.measurement.2019.107450.
20) Nguyen MT, Kim K. Genetic convolutional neural network for intrusion detection systems. *Future Generation Computer Systems*. 2020;113:418–427. Available from: https://dx.doi.org/10.1016/j.future.2020.07.042.
21) Almiani M, AbuGhazleh A, Al-Rahayfeh A, Atiewi S, Razaque A. Deep recurrent neural network for IoT intrusion detection system. *Simulation Modelling Practice and Theory*. 2020;101:102031. Available from: https://dx.doi.org/10.1016/j.simpat.2019.102031.
22) Kasongo SM, Sun Y. A Deep Long Short-Term Memory based classifier for Wireless Intrusion Detection System. *ICT Express*. 2020;6(2):98–103. Available from: https://dx.doi.org/10.1016/j.icte.2019.08.004.

23) Kasongo SM, Sun Y. A deep learning method with wrapper based feature extraction for wireless intrusion detection system. *Computers & Security*. 2020;92:101752. Available from: https://dx.doi.org/10.1016/j.cose.2020.101752.
24) Khan M, Karim M, Kim Y. A Scalable and Hybrid Intrusion Detection System Based on the Convolutional-LSTM Network. *Symmetry*. 2019;11(4):583. Available from: https://dx.doi.org/10.3390/sym11040583.
25) Hsu CM, Azhari MZ, Hsieh HY, Prakosa SW, Leu JS. Robust Network Intrusion Detection Scheme Using Long-Short Term Memory Based Convolutional Neural Networks. *Mobile Networks and Applications*. 2021;26(3):1137–1144. Available from: https://dx.doi.org/10.1007/s11036-020-01623-2.
26) Zhang J, Ling Y, Fu X, Yang X, Xiong G, Zhang R. Model of the intrusion detection system based on the integration of spatial-temporal features. *Computers & Security*. 2020;89:101681. Available from: https://dx.doi.org/10.1016/j.cose.2019.101681.
27) Yang XS, He X. Firefly algorithm: recent advances and applications. *International Journal of Swarm Intelligence*. 2013;1(1):36. Available from: https://dx.doi.org/10.1504/ijsi.2013.055801.