VOL. 18, NO. 14, JULY 2023 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

# HYBRID DEEP LEARNING-BASED INTRUSION DETECTION SYSTEM USING MODIFIED CHICKEN SWARM OPTIMIZATION ALGORITHM

Mohammed Ishaque[1], Md Gapar Md Johar[2], Ali Khatibi[1] and Muhammed Yamin[3]
[1]School of Graduate Studies, Management and Science University, Shah Alam, Malaysia
[2]Information Technology and Innovation Centre, Management and Science University, Shah Alam, Malaysia
[3]School of Computing, ANU Colleage of Engineering, Computing and Cybernetics, The Australian National University, Canberra, Australia
E-Mail: ishaq.research@gmail.com

**ABSTRACT**

Web Systems which are the backbone of information resources, communications, and personal information management, attackers might take advantage of their vulnerability and beguile them to get access to sensitive data or the web servers and apps in full. Wider usage of the Internet and its features have come a long ago, with both advantages and disadvantages. The security and its side effects have been discussed by researchers in various aspects of the network. HTTP, one of the most widely used network protocols, has paid a huge price due to various intrusions like SQL injections, code injections, and cross-site scripting (XSS). To handle these intrusions, various intrusion detection algorithms have been proposed and addressed in the literature. The accuracy and timeliness of these detections has been an issue due to false positives and the amount of information that ought to be processed and delivered within a short span of time. In recent times the classic classifiers have become outdated and detecting abnormal traffic in a web system has become a hassle. In this research, we propose handling intrusion detection over a Web System using hybrid Deep Learning (HDL) based classification and robust feature extraction using a modified chicken swarm optimization algorithm (MCSO). This approach also incorporates the idea of deep learning, which makes it possible to have a peer-to-peer learning system for aberrant patterns with a fewer number of characteristics, hence reducing the amount of time needed to complete the work. In order to distinguish or classify data that the web system has to deal with, a hybrid computational intelligence-based classifier algorithm is used. The combination of the hybrid classifier is fuzzy neural network with Long Short Term Memory (LSTM) which is basically used to classify the attacks and distinguish between normal and anomalous data. The use of helps in understanding the nature of intrusions over time, which is constant and predictable, On the other hand, with the assistance of deep learning and a method called feature extraction, which pulls important information from noisy data, we can do this. Lastly, the findings of the experiments show that this technique has an excellent detection performance, with an accuracy rate that is more than 98.7%.

**Keywords:** intrusion detection system, hybrid learning, modified chicken swarm optimization algorithm (MCSO), fuzzy neural network, long short-term memory (LSTM).

Manuscript Received 19 August 2023; Revised 22 September 2023; Published September 30, 2023

## 1. INTRODUCTION

During the course of its existence, the web application has progressed from a straightforward, unchanging, and read-only system to a sophisticated, dynamic, and user-friendly platform that offers users access to both information and services. Since they can be accessed from any computer with Internet connectivity and are readily available without charge, web apps have quickly become an essential component of everyday life [1]. They often deal with private information and are used in the performance of essential activities including as banking, networking, internet buying, and online tax filing. Since, because of their popularity, ubiquity, simplicity of use, and expanding user base, web apps have emerged as a top target for attackers. According to the annual Global Security Report 2018, every application contains at least one vulnerability, which examined billions of security incidents worldwide [2]. There are 11 failures per application. This study demonstrates how targeted, frequent, and sophisticated cyberattacks are on the rise. Several rape cases reveal evidence of significant preparation and forethought on the part of online perpetrators who extensively research their victims.

The application layer has become a frequent concern to web security since it may enable a lot of assaults in web-based applications and has grown significantly. The goal of hacking and crippling a website is to employ methods such as susceptible harmful programmers [3]. They have been used in low-level attacks and high-level data breaches that exposed web application foundations. This severe security issue affects most systems, especially high-availability businesses like healthcare, banking, and e-commerce [4]. With 40% of these web attacks being Cross-site Scripting (XSS) attacks, SQL injection (SQLi) attacks come in second with 24% of attacks, cross-section attacks come in third with 7% of attacks, local file inclusion (LFI) attacks come in fourth with 4% of attacks, and distributed denial of service (DDoS) attacks come in third with 3% of attacks. An XSS attack occurs when malicious code, sometimes in the form of a script, is delivered or executed through a web application from the victim's computer's browser [5]. As a result, it gives attackers the chance to steal sensitive

information or possibly gain control of specific machines. You might steal user cookies or filter personal information with this execution. This category (XSS), as well as other categories like "insecure deserialization" have seen an increase in web application vulnerabilities this year.

When a malevolent user manipulates a web application's input to create and submit a query that is not what the creator intended, this is known as a "SQL injection attack," and it may be done both manually and automatically [6]. SQL Injection Attacks (SQLIAs) are a major database security risk. The integrity and confidentiality of this information are therefore not sufficiently guaranteed. SQLIA code injection attacks exploit user input validation flaws. Malicious actors may pile their illicit input into the final query string, which databases employ [7]. As a result of this vulnerability, attackers may threaten the authority, integrity, and confidentiality of financial web applications and secret information systems. Cross-site attacks, in which malicious code is inserted on the client side of a website, are the most frequent network assaults on the internet. Attackers try to take advantage of a flaw discovered in a website with inadequate coding. It uses the victim's browser as a vector to spread malicious script from a vulnerable website [8]. The attacker, the website, and the victims may all utilize JavaScript, Flash, VBScript, and ActiveX to carry out this attack (XSS). In order to close this issue, developers used various protective coding techniques, yet these measures are insufficient.

Network traffic, syslog records, and operating system calls are constantly analysed by an intrusion detection system (IDS) to determine whether particular actions are legitimate or attacks [9]. There are two common ones for these systems: host-based intrusion detection systems (HIDS) and network-based intrusion detection systems (NIDS). A HIDS, on the other hand, is installed on a specific host and monitors data relevant to the system, a NIDS works using feature vectors that include summarised information about network traffic over a certain period of time. There are two popular approaches to intrusion detection: anomaly-based and signature-based [10]. By comparing predetermined patterns, the latter, also known as abuse detection, aims to identify and categorise assaults. This method only works for well-known assaults, keeping tolerable levels of false alarm rates.

To determine the web parameters with the greatest detection rate and increase detection rates, some researchers use the ant colony method [11]. Fuzzy entropy-based ant colony optimization (ACO) provides the best and cheapest network traffic characteristics for real-time intrusion detection data sets. It then recommends real-time intrusion detection network traffic attributes. Unfortunately, the method is time-consuming, quite complex, and may not always provide the best result in practice. In recent times the classic classifiers have become outdated and detecting abnormal traffic in a web system has become a hassle. In this research, we propose handling intrusion detection over a Web System using hybrid Deep Learning (HDL) based classification and robust feature extraction using a modified chicken swarm optimization algorithm (MCSO).

Section 2 analyses some of the most current methods for identifying various online threats using machine learning models, and this is how the remaining research is structured. The suggested method is shown in the process diagram in section 3. Section 4 gives the conclusions and observations. Section 5 discusses the result and further work.

## 2. LITERATURE REVIEW

Review some of the most current methods for applying machine learning to identify online attacks in this section.

Jemal *et al* [12] presented a taxonomy of the recently suggested detection and protection methods, along with a SQL injection attack. We categorize the various assault sources, objectives, and kinds. In addition, analyse and categorize the most recent and significant proposed solutions for mitigating the effects of this attack, paying particular attention to those that are founded on ontology and machine learning. Nadeem *et al* [13] established a model for the dynamic analyzer's access to the data. Moreover, the suggested approach has certain benefits, including broad application, fast response times, coverage of a variety of SQL Injections (SQLI) methodologies, and resource efficiency. Sheykhkanloo *et al* [14] suggested a model for detecting and classifying SQLi threats based on neural network (NN). Three components made up the suggested model: 1) a Uniform Resource Locator (URL) generator, 2) a URL classifier, and 3) a NN model. The suggested model was effective in 1) classifying each produced URL as malicious or benign, and 2) determining the kind of SQLi attack used by each malicious URL. The proposal's effectiveness was demonstrated by the published results. In employing two situations and contentious data sets, the author of this research reevaluates how well the plan performed. To show the model's accuracy, true-positive rate, and false-positive rate, the experiment data are presented.

Kuang *et al* [15] presented a working prototype of DeepWAF, it utilizes deep learning to identify online assaults. We cover how to effectively employ CNN, LSTM, and their combinational models. LSTM will broadcast CNN and LSTM content. The four detection models have a detection rate of 95% and a false alarm rate of 2% on the HTTP DATASET CSIC 2010 dataset. In order to better understand the reasons for false positives and false negatives, we also conducted case studies. Tekerek *et al* [16] using deep learning techniques, An anomaly-based Web attack detection programme was suggested. The architecture includes data preprocessing and CNN phases. The proposed CNN architecture was tested using CSIC2010v2 datasets. Using an anomaly-based detection type, the suggested architecture carried out Web assault detection. In accordance with the study's experimental findings, the suggested CNN deep learning architecture produced favorable results. Tian *et al* [17] a technique for detecting online attacks that makes use of URL analysis has been suggested. The technology is

installed on edge devices and made to detect online assaults. In the notion of the Edge of Things, the cloud addresses the aforementioned problems. Several concurrent deep models increase stability and updating convenience. We tested the system with two deep models concurrently using various datasets. The testing findings show that the system is competitive in identifying online assaults with accuracy scores of 99.411 percent, true positive rate (TPR) of 98.91 percent, and a detection rate of normal requests (DRN) of 99.55 percent.

Rawat *et al* [18] for the categorization and prediction of SQL-Injection attacks, proposed an SVM (Support Vector Machine). The best accuracy of any currently available SQL-Injection Detection Methods is achieved by the approach we present, with a detection rate of 96.47% for SQL-Injection attacks. Zhang et al [19] described a way for employing a custom-created CNN to identify Online threats using deep learning. The approach is based on analyzing the HTTP request packets, for which only a small amount of preprocessing is required, while the CNN itself is responsible for the laborious task of extracting features. The CNN operates effectively and detects online threats with a high detection rate and low false alarm rate for HTTP DATASET CSIC 2010. Pan *et al* [20] An unsupervised/semi-supervised web attack detection method is based on the Robust Software Modeling Tool (RSMT), which autonomously monitors and characterizes online application runtime behaviour. Second, how does RSMT instruct a stacked denoising autoencoder to encode and rebuild the call graph to allow end-to-end deep learning? This learning method estimates request data reconstruction errors using a low-dimensional representation of raw features and unlabeled request data. In the last step of this process, we conduct an analysis of the outcomes of RSMT's experimental testing on real applications that have been the time, which is constant and predictable, whereas with the use of helps in understanding the nature of intrusions over and distinguish between normal and anomalous data. The

## 3. METHODOLOGY

To determine if events and actions taking place in a web application or network are authentic, security programmers are used. The intrusion detection system examines packets entering and leaving the network and keeps track of those that are susceptible or participating in an attack in accordance with established criteria. A system to detect and precisely identify assault is urgently needed to safeguard online applications against SQL injection attacks, code injections, and Cross-Site Scripting (XSS). SQL injection, XSS, and command injection attacks on the Web can be detected using a combination of features in the Web Injection Attacks Detection system. By building and altering the parameters of Web requests, web injection attacks accomplish their goal. As a result, we analyse the URL and Body content of HTTP requests to create a detection model. We propose to handle intrusion detection over a Web System using hybrid Deep Learning (HDL) based classification and robust feature extraction using chicken swarm optimization algorithm (CSO). This approach also incorporates the idea of deep learning, which opens up the prospect of an aberrant pattern peer-to-peer learning system with fewer features, thus reducing time. In order to distinguish or classify data that the web system has to deal with, a hybrid computational intelligence-based classifier algorithm is used. The combination of the hybrid classifier is a fuzzy neural network with LSTM which is basically used to classify the attacks and distinguish between normal and anomalous data. The use of helps in understanding the nature of intrusions over time, which is constant and predictable, whereas with the help of deep learning and feature extraction process which extract key features from the intrusive data.

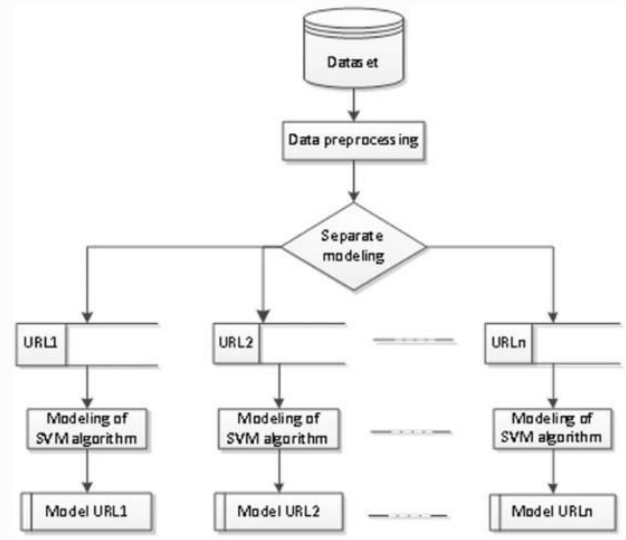


**Figure-1.** The process of the proposed web attack detection model using an intrusion detection system.

### 3.1 Dataset description

In this work, two types of datasets are used for web attack detection based on intrusion detection system. The proposed detection model is evaluated in NSL-KDD and CICIS-2017 datasets. The section below provides a comprehensive explanation of the dataset.

#### 3.1.1 NSL-KDD dataset

In this study, the KDD Cup 1999 Data (KDD99) NSL-KDD dataset from the Canadian Institute for Cyber Security is used. The KDDTrain+ dataset was divided into a 75% to 25% ratio for this study's training and testing purposes. Denial of Service (DoS), Probe, User to Root (U2R), and Remote to Local (R2L) assaults are among the four distinct groups of attacks in the dataset. The following is a succinct description of each attack:

- Attempts to stop traffic from and to the target system are made in a DoS attack. The system can't handle the requests because the IDS is being inundated with an extraordinary volume of traffic, so it shuts down to protect itself. As a result, a network cannot receive

regular traffic. This is the data set's most prevalent attack.

- A probe attack or surveillance attack is one that tries to acquire data from a network. This attack is being carried out with the intention of posing as another perpetrator to steal critical client data like financial or personal information.
- U2R attacks start try to access a system or network with a super-user account using an ordinary user account (root). By taking advantage of vulnerabilities, The attacker wants root privileges or system access.
- Getting local access to a remote computer is possible using R2L. The network is being attempted to be "hacked" by an attacker that lacks local access to the system. Table 1 shows the dataset's attack type counts. The dataset includes 21 attacks in Table-2 categories.

**Table-1.** Number of incidents for each assault type.

| | |
|---|---|
| **Normal** | 67343 |
| **DoS** | 45927 |
| **Probe** | 11656 |
| **R2L** | 995 |
| **U2R** | 52 |

**Table-2.** Various assaults in each attack type.

| | |
|---|---|
| **DoS** | Back, Land, Neptune, Pod, Smurf, Teardrop, Apache2, Udpstorm, Processtable, Worm |
| **Probe** | Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint |
| **R2L** | Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named |
| **U2R** | Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps |

### 3.1.2 CICIDS2017 dataset

The CICIDS2017 dataset utilises CICFlowmeter-V3.0 to extract 78 characteristics and 79 labels that closely resemble real-world network data (PCAPs). According to the HTTP, HTTPS, FTP, SSH, and email protocols, this dataset contains the abstract characteristic attitudes of 25 users, as indicated in Table 3. During the course of many time periods, data are collected. According to the McAfee Report from 2016, the attacks contained in this dataset can be broken down into the following categories: brute force FTP attacks, brute force SSH attacks, DoS attacks, heartbleed attacks, web attacks, infiltration attacks, botnet attacks, and DDoS attacks. None of these types of attacks were discovered in any of the datasets that were discussed earlier. CICIDS2017 examines the abstract characteristics of human interactions by simulating various multi-stage attack scenarios utilising the B-Profile system and the Alpha profile. Due to its realistic and trustworthy benchmarking, this dataset's key characteristic stands out from those of others. To verify the validity of the assessment, the benchmarking uses 11 criteria, including accessible protocols and entire traffic.

**Table-3.** Dataset details of CICIDS 2017.

| Name of Files | Class Found |
|---|---|
| Monday-Hours.pcap_ISCX.csv | Benign (Normal human activities) |
| Tuesday-Hours.pcap_ISCX.csv | Benign, FTP-Patator,SSH Patator |
| Wednesday-.pcap_ISCX.csv | Benign, DoS GoldenEye, DoSHulk, DoS lowhttptest, DoS slow loris, Heartbleed |
| Thursday-WebAttacks.pcap_ ISCX.csv | Benign, Brute Force, SQL Injection, XSS. |
| Thursday-Infilteration.pcap_.csv | Benign, Infiltration |
| Friday-pcap_ISCX.csv | Benign, Bot |
| Friday-PortScan.pcap_ISCX.csv | Benign, PortScan |
| Friday- DDos.pcap_ISCX. csv | Benign, DDoS |

SQL injection attacks may take several forms, including wide-byte, error-based, Union, and Boolean. Data is simple to acquire for popular SQL injection attacks such union injection assaults, and the model's detection findings are reliable. Wide-byte injection data capture may be tricky. It lacks training data for certain subdivision kinds, it results in insufficient deep model training for this sort of data and a high incidence of false negatives when identifying this type. The normal approach in this case is to repeatedly sample this restricted set of sample types or, alternatively, to expand samples using a sampling algorithm. Fresh samples with overlap or low quality typically arise from this procedure. Sub-training often suffers from severe over-fitting problems, which leaves the learner with poor generalisation skills.

### 3.2 Preprocessing

Machine Learning is used in this investigation. CICIDS-2017 CSV data from the ISCX Consortium. Machine Learning. Eight (8) traffic monitoring sessions are included in CSV, which is a file format for comma-separated values. This file includes anomalous traffic known as "Attacks" traffic as well as typical traffic, which is designated as "BENIGN" traffic. The second column of

Table 3 gives a more thorough description of the attack traffics. In this dataset, there are 14 more sorts of assaults in addition to regular traffic and benign traffic. The numericalization phase converts null or infinity symbols to zeros or mean values. The normalization step is then required since, according to their histograms, several CICIDS2017 characteristics have extremely big values and non-distributed data. We use the formulae in (1-4) to perform the normalisation technique inside the [-3, 3] region in order to scale all attribute values to the same scale:

$$X = -3 < \text{mininmum}\left(z_{max}, \text{maximum}(z, z_{min})\right) < 3 \quad (1)$$

$$z_i = \frac{x-\mu}{\sigma} \quad (2)$$

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i) \quad (3)$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \quad (4)$$

A set of 38 characteristics with a range of [3, 3] values are generated by the preprocessing. To enhance the data's distribution and produce better training results, this range was chosen.

### 3.3 Feature Extraction

Expertly retrieved characteristics are able to compensate for the aforementioned circumstance. Experts understand SQL injection attack classification and its numerous types. On the basis of their own subject expertise and understanding, experts may often suggest superior solutions. To cover certain subdivision types, representative characteristics are employed. Experts manually extract discrete features increasing detection rates and results accuracy. Deep learning algorithms can also extract discrete features at a high level, but manual experts are typically able to do so with higher detection results.

Figure-2 illustrates the framework of the detection model. Two components make up the detection model. Sequence feature extraction is one component. Using the pre-processed Web assault sample data, the CSO model automatically extracts sequence characteristics; the human extraction of discrete features is the other step. A priori knowledge is used by experts to define particular characteristics, and then, discrete characteristics are retrieved and validated from input data. After being retrieved, the discrete and sequence characteristics are combined and fed into a hybrid deep learning algorithm for classification. Softmax activating functional layer output normalises categorization findings.

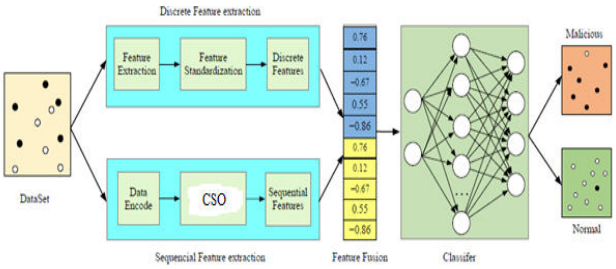


**Figure-2.** The architecture of model.

#### 3.3.1 Discrete features extraction and standardization

Discrete feature extraction is covered there; you will need to extract the following discrete features in order to evaluate feature fusion techniques.

- Keyword characteristics: It's critical to use keywords to recognise Web injection attacks. For instance, despite the fact that the word "script" is seldom seen in URLs, it often appears in the payload of XSS attacks; In a manner similar to this, it is important to be on the lookout for SQL injection attacks whenever the term "select" or any other SQL statement keywords occur in the URL;

- Length feature: Statistics show that harmful URLs are significantly longer than benign URLs because web assaults often require to create certain attack payloads;

- Other features: Malicious requests vary from benign requests primarily by their number percentage, peculiar character percentage, and the measured value contains IP.

#### 3.3.2 Modified Chicken Swarm Optimization algorithm (MCSO)

Standardizing the extracted discrete features comes next after extracting artificial discrete features. Due to the fact that each feature's quantitative scale is not always the same, while the model is being trained, the features with the bigger scales will be more important than those with the smaller scales. To reduce the effects of scale variations and feature units during the feature standardisation phase and accelerate the rate of convergence of model training, manage distinct elements of varying sizes under one standard scale. This makes standardising distinctive traits an important first step. The mean is 0 and the standard deviation is 1 by feature standardisation using the original discrete features as the starting point. Equation (5) is the definition of the standardisation formula, where the discrete feature mean value is assumed to be m and the standard deviation is assumed to be s.

$$\bar{z} = (\bar{x} - \mu)/\sigma \quad (5)$$

Chicken Swarm Optimization (CSO), a novel bio-inspired optimization method. Imitate the chicken swarm's hierarchical structure and behaviour. Each chicken swarm cluster has a rooster, hens, and chicks. Various types of chickens adhere to various rules of motion [23]. There are hierarchical chicken tournaments. Chickens' social life are affected by it. A flock's dominant chickens will rule the weak. The group has dominant and submissive hens and roosters. Once a hierarchical structure is formed, adding or removing hens from a flock would upset the social order. When they locate food, the roosters may order their group to eat first., but the dominant individuals get first access to it. When raising their young, hens display the same kind manners. Individuals from other groups, however, do not share this trait. When chickens from another flock intrude on their area, roosters would make a loud call. Generally, a chicken's behaviour varies depending on its gender. The group's dominant rooster would aggressively pursue food and engage in conflict with any intruding chickens. Foraging for food by the head roosters would almost always be done by the dominant chickens. when it was time for the submissive ones to go hunting for food, they would unwillingly stand outside the group. Many chicken breeds compete against one another. The chicks look for food near their mother as they get older.

We can mathematically develop CSO based on the aforementioned descriptions. For ease of understanding, we idealized the hens' actions using the following guidelines.

a) Many groups may be seen in the chicken swarm. The dominating rooster, a few hens, and chicks are present in each group.

b) The chickens' fitness levels determine how to organise and identify them (roosters, hens, and chicks). The fittest chickens would be a group's top rooster. Chicks are hens with low fitness ratings across categories. Hens would be the other animals. The group the hens dwell in is determined at random. Moreover, a mother-child attachment is haphazardly created between the hens and the chicks.

c) Mother-child, dominance, and hierarchical relationships are constant in groups. Only sometimes (G) do their statuses alter.

d) Chickens may prevent roosters from consuming their food while following them to get food. Chickens could take other people's food. Around their mother (hen), the chicks search for nourishment. Food competition favours powerful people.

The letters RN, HN, CN, and MN stand for the number of roosters, hens, chicks, and mother hens. Roosters are the greatest RN chickens, whereas chicks are the worst CN chickens. The others are treated like chickens. The locations of all N virtual chickens are shown $x_{i,j}^{t}(i \in [1, \ldots \ldots, N], j \in [1, \ldots ., D])$ discover food at time step t in a D-dimensional space. The optimization issues in this study are the simplest. The hens with the lowest RN minimum fitness values are thus the best RN chickens.

**a) Movement of the chickens**

The roosters who have higher levels of physical fitness are given more access to food than the roosters that have lower levels of physical fitness. This problem is simplified by simulating that fitter roosters can obtain food in more areas [24]. The following is one possible formulation for this:

$$x_{i,j}^{t+1} = x_{i,j}^{t} * (1 + Randn(0, \sigma^2)) \quad (6)$$

$$\sigma^2 = \begin{cases} 1, if\ f_i \leq f_k \\ \exp\left(\frac{(f_k - f_i)}{|f_i| + \varepsilon}\right), otherwise, \end{cases} k \in [1, N], k \neq i \quad (7)$$

Where $Randn\ (0, \sigma^2)$ has a mean of 0 and a standard deviation, and is Gaussian $\sigma^2$. $\varepsilon$, It prevents zero-division errors as a computer's lowest constant. A rooster's index, k, and fitness value, f, are selected at random.

The hens may go on a food hunt with the roosters in their group. They also stole other chicken excellent food while being restricted. Dominant hens would outcompete meek ones for food. Following are some mathematical formulations for these occurrences.

$$x_{i,j}^{t+1} = x_{i,j}^{t} + S1 * Rand * \left(x_{r1,j}^{t} - x_{i,j}^{t}\right) + S2 * Rand * \left(x_{r2,j}^{t} - x_{i,j}^{t}\right) \quad (8)$$

$$S1 = \exp((f_i - f_{r1})/(abs(f_i) + \varepsilon)) \quad (9)$$

$$S2 = \exp((f_{r2} - f_i) \quad (10)$$

A uniform random number called Rand is used [0, 1]. $r_1 \in [1, \ldots ., N]$ is a group member of the ith hen and an index of the rooster, while $r_2 \in [1, \ldots ., N]$ is a number that corresponds to the chicken (whether it be a rooster or a hen). The ith hen would look for food in the same sequence as the previous hens, assuming that S1 is equal to zero. S2 will be lower and the differential in position between the two chickens will be higher the more the two chickens' fitness levels diverge. The hens would thus be less inclined to eat the food that other chickens had found. There are group contests, which is why the formula form of S1 is different from that of S2. For simplicity of use, the contests between the chickens in a group are used to represent how fit the chickens are in comparison to how fit the rooster is. If S2=0, the ith hen would search for food in its own zone. The group's rooster fitness is unique. Hence, the closer S1 is near 1, the closer the ith hen is to its groupmate the rooster. In light of this, the dominant hens would be more inclined to consume the food than the submissive hens.

To find nourishment, the chicks move around their mother. This is expressed as follows.

$$x_{i,j}^{t+1} = x_{i,j}^{t} + FL * (x_{m,j}^{t} - x_{i,j}^{t}) \quad (11)$$

A random choice between 0 and 2 would be made by each chick's FL, taking into account their unique variations.

As compared to alternative methods, the traditional CSO algorithm offers several benefits. Attack detection-based feature extraction, however, is a challenging non-linear task. As a result, suggest an MCSO method to enhance CSO's performance and make it more appropriate for the attack detection model in online applications. The adaptive weight factor is introducing to the chicks for modifying the step size in real-time. The detail of the updation is explained in the below section.

**b) Improved update method of the hens with Adaptive weighting factor (AWF)**

Comparatively speaking to the roosters, the hens are somewhat removed from the ideal situation. As a result, the chicks are given the weighting factor AWF for modifying the step size on the fly. AWF looks like this:

$$Adaptive\ weight\ factor\ (AWF) = \exp(R/K)^{P} \quad (12)$$

where R is the probability of coming across chicks that have the same fitness score. Parameter tweaking tests show that K = 5 improves performance, where P is the step index. As a consequence, the number of optimisation iterations in which the same fitness values emerge is correlated with the introduced AWF for the updating of chicks. The optimal answer may be within the range if the hens with the same fitness score have few iterations. As a consequence, AWF can be rather little. If not, w has a high value. Integrating AWF describes the chicken solution updating technique:

$$x_{i,j}^{t+1} = AWF \times x_{i,j}^{t} + exp\,\frac{f_i - f_{ri}}{|f_i| + \varepsilon} \times Rand \times \left(x_{r1,j}^{t} - x_{i,j}^{t}\right) + \exp(fr_2 - f_{r1}) \times Rand \times \left(x_{r2,j}^{t} - x_{i,j}^{t}\right) \quad (13)$$

Chicks updating methods are exclusively related to their mothers' in the classic CSO algorithm, which made the solution the chicks represented easily deviate from the ideal position.

Input: preprocessed data
Output: sequential features

Initialize a population of $N$ chickens and define the related parameters;
Evaluate the $N$ chickens' fitness values, $t$=0;
While ($t$ < Max_Generation)
If ($t$ % $G$== 0)
Rank the chickens' fitness values and establish a hierarchal order in the swarm;
Divide the swarm into different groups, and determine the relationship between the chicks and mother hens in a group;
End if
For $i$ = 1 : $N$
If $i$ = = rooster Update its solution/location using equation (6);
End if
If $i$ = = hen Update its solution/location using equation (8);
End if
If $i$ = = chick Update its solution/location using equation (11);
End if
If AWF > N
Evaluate the new solution using equation (13);
If the new solution is better than its previous one, update it;
End for
End while

**Algorithm 1**. Algorithm for modified chicken swarm optimization.

### 3.4 Hybrid Deep Learning based Classifier

We input the classifier with the fusion features created by combining discrete and series sequence information. To achieve classification and normalisation, the classifier uses a softmax activation function and a fully connected neural network. Figure-3 depicts the classifier's structural layout.

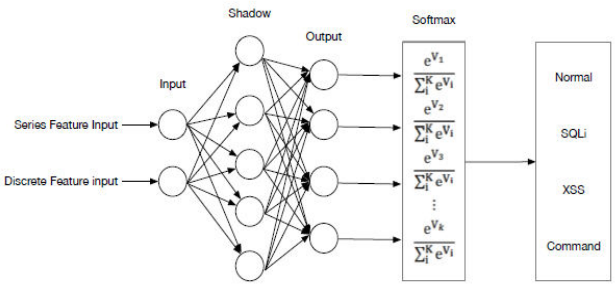


**Figure-3.** The architecture of classifier model.

In the existing related works, for detecting Web injection threats, many applications of both machine learning and deep learning. In order to efficiently analyse and recognise data with certain patterns or features, On the basis of the development of multiple databases, deep learning models are often created. While deep learning models often work well, one of their drawbacks is the insufficient representation of assault kinds for a small sample of data. In order to detect attacks, this research work introduced a hybrid deep learning model.

#### 3.4.1 Fuzzy neural network (FNN)

Figure-4 depicts the design of the fuzzy neural network. The buried layer is fully coupled to the input layer, as shown. Fuzzy membership functions (FSH) determine each hidden neuron's output [25]. As the FSHs formed while clustered for that class are only coupled to

the class node, Figure-4 shows a partly coupled output-hidden layer connection. Fuzzy union outputs class node.

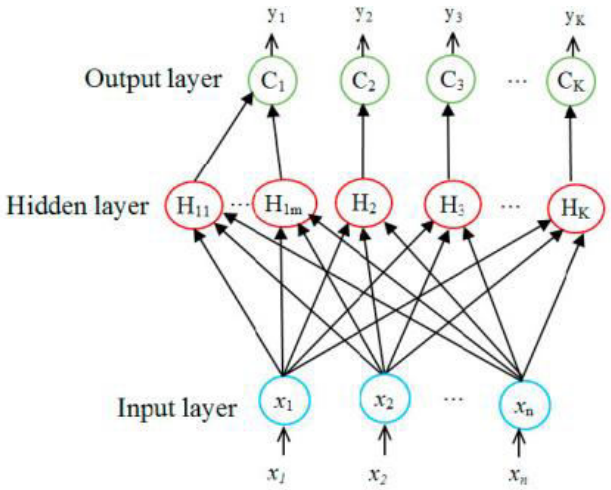


**Figure-4.** Fuzzy neural network.

Class 1 contains m clusters, according to Figure-4, while other classes only have one cluster, which may change depending on the application. Two steps train proposed architecture: a) Fuzzy clustering with Maximum count is used to generate FSH in the hidden layer of the FNN. As the centroid, choose the fuzzy membership function-clustering pattern. Reduce single pattern clusters by pruning after clustering. Fuzzy membership function is given by

$$f(l, r_j) = \begin{cases} l & l \le r_j \\ \frac{r_j}{l} & \text{otherwise} \end{cases} \quad (14)$$

### 3.4.2 Long Short Term Memory (LSTM)

Four gates and a memory cell make up an LSTM network. LSTM networks have four gates: forget gate (f), input gate, control gate (c), and output gate (o). RNN algorithm data dependence may be addressed by extracting and remembering the underlying data pattern [26]. The LSTM architecture may be seen in Figure-6. The architecture accepts three inputs: ht1, xt, and b. Present input vector, bias, and preceding cell state are Xt, b, and ht-1, respectively. The architecture's ct output shows memory content. ht, representing cell status, is another architectural output. The memory cell's data is influenced by these four gates. Forget gate yields 0-1 values. The preceding memory cell's value is disregarded by this value. A number near to 0 from the forget gate indicates so much of the prior timestamp's storage won't be utilised at the present time stamp, while a value close to 1 indicates the opposite. The following equations serve as representations for the LSTM gates:

The LSTM's forget gate is represented by Equation 9 as,

$$f_t = \alpha_g(w_f x_t + u_f h_{t-1} + b_f) \quad (15)$$

Equation 10 represents the input gate of LSTM as

$$i_t = \sigma_g(w_i x_t + u_i h_{t-1} + b_i) \quad (16)$$

Equation 11 represents the control gate of LSTM as,

$$c_t = f_t \times c_{t-1} + \sigma_h(w_c x_t + u_c h_{t-1} + b_c) \quad (17)$$

Equation 12 and 13 represents the control gate of LSTM as

$$o_t = \sigma_g(w_o x_t + u_o h_{t-1} + b_o) \quad (18)$$

$$h_t = o_{t\times}\sigma_h c_t \quad (19)$$

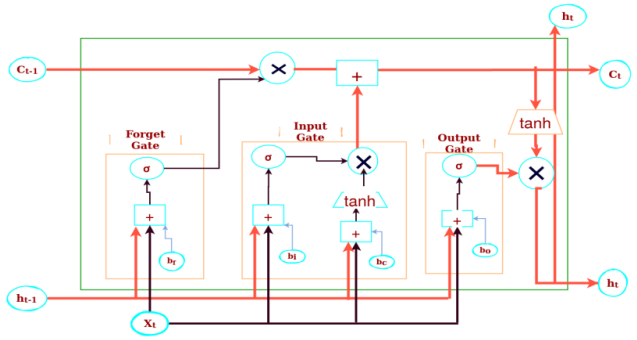


**Figure-5.** Architecture of LSTM.

Whereas the hyperbolic tangent function is, sigmoid function is represented by in this example σh. Weights are represented by the letters w and u. Typically; these weights prevent the gradient problem from occurring. Each layer in this case had 50 LSTM units. Each layer input has an attention value. Attention rating helps forecast by indicating information significance. The thick layer determines diabetes using an attention vector. Figure-5 demonstrates that none of the network's gates are related to the information in the prior memory. If the output gate is locked, this leads to an abnormal circumstance. Due to this, prediction and classification tasks perform less effectively. This work's main objective is to use adaptive linear function to classify assaults in order to get over LSTM's drawbacks.

- Adaptive linear function (ALF)

Introduce some notation and terminology used throughout the study effort before going into the specifics of the regularised model. Let $D_t = \{(x_1^t, y_1^t), \ldots, (x_n^t, y_n^t)\}$ be a bundle of practise data containing $y_i^t \in \{1, \ldots, C\}$ indicating the class label $x_i^t$, and let $D_a = \{(x_1^a, v_1^a), \ldots, (x_m^a, v_m^a)\}$ be a collection of additional data. Learn a prediction function provided by in particular $Pr(y|x)$, i.e., the likelihood that the input data matches a class label, given the input data.

Consider using a Softmax function to simulate the probability $Pr(v|x)$if we use h(x) to represent the high level properties of x.

$$Pr(v|x) = \frac{\exp(u_v^T h(x))}{\sum_{v'}^{K} \exp(u_v^T h(x))} \quad (20)$$

where $\{u_v\}$ the models class weights should be indicated.

$$Pr(y = c|v, x) = \frac{\exp(w_{v,c}^T h(x))}{\sum_{c=1'}^{C} \exp(w_{v,c}^T h(x))} \quad (21)$$

where $\{w_{v.c}\}$ is the weight.

Factor-specific weights $w_{v.c}$ should capture equivalent high-level information. Characteristics to the associated weight values uv to provide motivation for the non-trivial regularisation. In order to do this, add the following adaptive linear function between "$w_{v,c}$" and "uv.",

$$ALF(\{w_{v,c}\}, \{u_v\}) = \frac{\beta}{2} \sum_{v=1}^{K} \sum_{c=1}^{C} \|w_{v,c} - u_v\|_2^2 \quad (22)$$

Since there is no method to deduce the viewpoint class from the original data, a per-viewpoint category classifier called $w_{v.c}$ cannot be learned to interpret the regularisation. The per-viewpoint category classifier receives input from the regularisation, which moderates intra-class variance in the task. Nevertheless, train viewpoint classifiers Only on class augmented data. By applying a normal prior to $w_{v.c}$, the aforementioned regularisation may also be understood.

$$Pr(w_{v,c}|u_v) \propto exp\left(-\frac{\beta}{2}\|w_{v,c} - u_v\|_2^2\right) \quad (23)$$

The regularisation in (18) also shares weights across Fine-grained detection and factor-type hyper-class classification models. In order to demonstrate this, we introduce the formula $w0_{v,\ c} = w_{v.c}$ uv and express the regularizer as (18).

$$ALF(\{w'_{v,c}\}) = \frac{\beta}{2} \sum_{v=1}^{K} \sum_{c=1}^{C} \|w'_{v,c}\|_2^2 \quad (24)$$

And $Pr(y = c|x)$ is computed by

$$Pr(y = c|x) = \sum_{v=1}^{K} \frac{\exp(w'^T_{v,c} + u_v)^T h(x))}{\sum_{c=1'}^{C} \exp(w'^T_{v,c} + u_v)^T h(x))} \frac{\exp(u_v^T h(x))}{\sum_{v'=1}^{K} \exp(u_{v'}^T, h(x))} \quad (25)$$

Thus, the proposed adaptive linear function effectively updates the regularization factor while extracting the data.

## 4. RESULTS AND DISCUSSIONS

This portion assesses the model for detecting injection attacks and the sample generation approach. To improve its classification results, the model constantly modifies its parameters using the training data. During model multiple training, to determine the optimum model hyperparameters, it uses the training dataset. Finally, the classification model is evaluated using the test set. Command insertion assaults are detected via deep learning and analyse how sample production affects model detection. The model's performance is assessed using some metrics. Command injection samples are "positive," benign samples are "negative," and both are "neutral".

The ratio of data for which all hypotheses are confirmed (both favourable and unfavourable) to the entire sample size is known as the accuracy rate.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (26)$$

The model's prediction accuracy is the proportion of samples properly categorised as positive classes:

$$Precision = \frac{TP}{TP+FP} \quad (27)$$

For the initial sample, the recall rate is the proportion of positive class samples in all test data that were properly classified:

$$Recall = \frac{TP}{TP+FN} \quad (28)$$

A balanced assessment index:

$$F1 - value = \frac{2*precision*recall}{precision+recall} \quad (29)$$

False negatives and false positives are defined as the proportion of false negatives to genuine clones and positives to identified clones, respectively.

$$False\ negative\ (\%) = \frac{[N]}{[A]} * 100 \quad (30)$$

$$False\ positive\ (\%) = \frac{[P]}{[A]} * 100 \quad (31)$$

where P stands for a false positive, a stands for genuine clones, N stands for false negatives in percentage, and D stands for discovered clones.

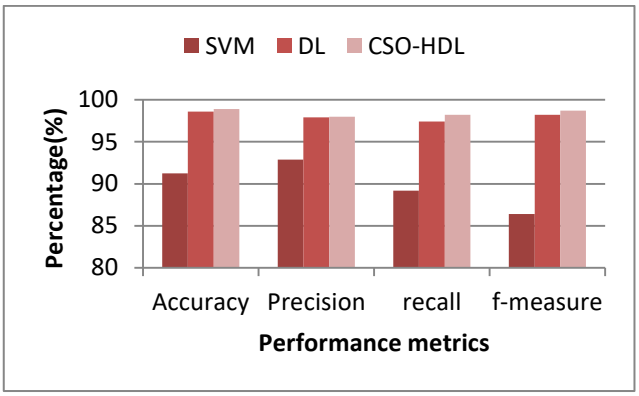


**Figure-6.** The results of IDS based web attack detection model.

Figure-6 illustrates this. At 98.9%, 98%, 98.2%, and 98.7%, respectively, the accuracy, precision, recall, and f-measure rates are the greatest, and all three indicators were higher than 97%. The signs no longer

improve but instead significantly decrease when the ratio is raised to 50%. Experimental findings suggest that the sample generation strategy presented may reduce overfitting caused by imbalanced sample distribution. It allows the classifier to increase detection accuracy and generalisation capabilities. Feature fusion identifies Web injection threats. We use a multi-classification approach to identify SQL injection, XSS, and command injection threats in web traffic. Thus, compared to the two-class issue, the multi-class model's evaluation index is calculated in a different way. For each kind of assault, first determine the outcomes for accuracy, precision, recall, and F1-value. The detection model's output will then be determined by calculating the arithmetic mean of each evaluation measure.

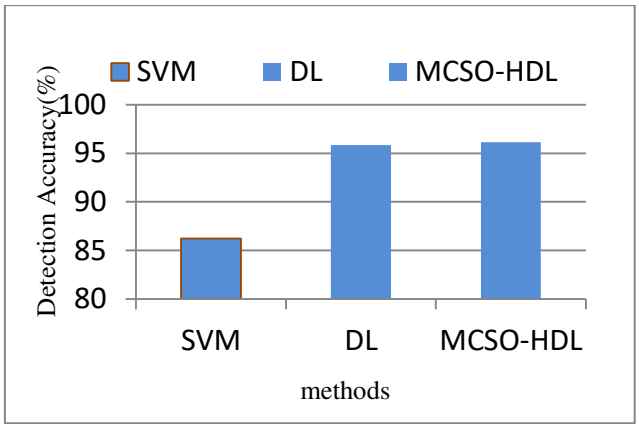


**Figure-7.** The suggested IDS-based web attack detection model's detection accuracy was compared to that of current techniques.

The suggested MCSO-detection HDL's accuracy is superior than the techniques currently in use, as shown in Figure-7. In comparison to the DL method metric of 95.84% and the SVM technique metric of 86.24%, the suggested MCSO-HDL achieves superior detection accuracy of 96.14%. As compared to current security solutions, the suggested method achieves improved detection accuracy.

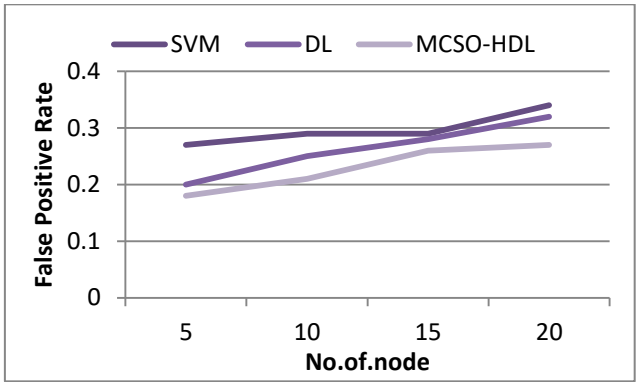


**Figure-8.** Comparison of the proposed MCSO-HDL with current techniques False Positive Rates (FPR).

False Positive Rate (FPR) in Figure-8 illustrates how the suggested MCSO-HDL performs better than the current approaches. The proposed MCSO-HDL produce lesser False Positive Rate. From the result it is identified that the proposed Hybrid deep learning based classification provides the best prediction rate than the existing models. It finds that the suggested security approach has a lower FPR than current methods.

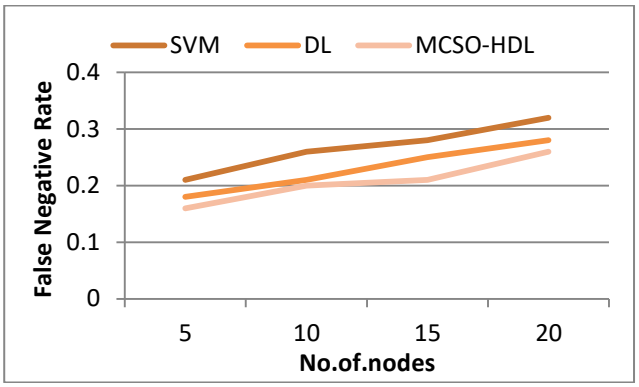


**Figure-9.** Comparison of the proposed MCSO-False HDL Negative Rate (FNR) with the current approaches.

In the Figure-9. FNR, MCSO-HDL outperforms prior approaches. The proposed MCSO-HDL produce lesser False Negative Rate. Assault forecasting characteristics are better selected using enhanced Chicken swarm optimization. It finds that the suggested security technique has a lower FPR than current methods.

- **Proposed web attack detection Model Discussion**

Enhance the detection of injection assaults with more precision via studies, and lower the false positive rate by using a different sample creation technique. Testing the impact of this approach on detection accuracy is the main goal of this proposed IDS-based web assault detection model. Choose to first confirm that this strategy is efficient in enhancing the accuracy of model identification accuracy and performance. Research demonstrates that the effectiveness and accuracy of training are impacted by varied inputs of data length. Hence, to verify the model's effectiveness ultimately achieve superior detection accuracy and runtime performance, we chose the right fixed length input via experiments. The aforementioned findings demonstrate that the feature-based detection approach has good generalisation and can successfully identify web injection attempts in communication traffic.

## CONCLUSIONS

In the last several decades, information delivery has become more dependent on web attack detection. In this research work, discussed an intrusion detection model using hybrid Deep Learning (HDL) based classification and robust feature extraction using modified chicken swarm optimization algorithm (MCSO). In order to distinguish or classify data which the web system has to deal with, a hybrid computational intelligence-based

classifier algorithm is used. Fuzzy neural networks and LSTM are combined to create the hybrid classifier, which categorizes attacks and distinguishes normal and abnormal data. The limits of huge network data flow, high dimension, and data feature extraction are solved by this study, which gathers the characteristics of typical Web assaults, examines the HTTP protocol, and picks important data features. Separate models are built for each sort of access request due to the significant variances in the parameter features of each. The suggested intrusion detection system provides a greater detection effect, according to simulation and experimental data. Consequently, the suggested classification model is not very accurate, but optimal training with a huge dataset may enhance it, which is the future focus of the study.

## REFERENCES

[1] G. Deepa and P. S. Thilagam. 2016. Securing web applications from injection and logic vulnerabilities: Approaches and challenges. Information and Software Technology. 74: 160-180.

[2] G. E. Rodríguez, J. G. Torres, P. Flores, and D. E. Benavides. 2020. Cross-site scripting (XSS) attacks and mitigation: A survey. Computer Networks. 166: 106960.

[3] F. Q. Kareem, S. Y. Ameen, A. A. Salih, D. M. Ahmed, S. F. Kak, H. M. Yasin, et al. 2021. SQL injection attacks prevention system technology. Asian Journal of Research in Computer Science. 6(15): 13-32.

[4] T. Scholte, W. Robertson, D. Balzarotti and E. Kirda. 2012. An empirical analysis of input validation mechanisms in web applications and languages. in Proceedings of the 27th Annual ACM Symposium on Applied Computing. pp. 1419-1426.

[5] M. K. Gupta, M. C. Govil, and G. Singh. 2014. Static analysis approaches to detect SQL injection and cross-site scripting vulnerabilities in web applications: A survey. in International conference on recent advances and innovations in engineering (ICRAIE-2014). pp. 1-5.

[6] R. Johari and P. Sharma. 2012. A survey on web application vulnerabilities (SQLIA, XSS) exploitation and security engine for SQL injection. in 2012 international conference on communication systems and network technologies. pp. 453-458.

[7] D. Gollmann. 2008. Securing web applications. Information Security Technical Report. 13(1): 1-9.

[8] D. Mitropoulos, P. Louridas, M. Polychronakis, and A. D. Keromytis. 2017. Defending against web application attacks: approaches, challenges and implications. IEEE Transactions on Dependable and Secure Computing. 16(2): 188-203.

[9] V. Prokhorenko, K. K. R. Choo and H. Ashman. 2016. Web application protection techniques: A taxonomy. Journal of Network and Computer Applications. 60: 95-112.

[10] N. Agarwal and S. Z. Hussain. 2013. A closer look at intrusion detection system for web applications. Security and Communication Networks.

[11] J. Fonseca, N. Seixas, M. Vieira, and H. Madeira. 2013. Analysis of field data on web security vulnerabilities. IEEE Transactions on Dependable and Secure Computing. 11(2): 89-100.

[12] I. Jemal, O. Cheikhrouhou, H. Hamam, and A. Mahfoudhi. 2020. SQL injection attack detection and prevention techniques using machine learning. International Journal of Applied Engineering Research. 15(6): 569-580.

[13] R. M. Nadeem, R. M. Saleem, R. Bashir and S. Habib. 2017. Detection and prevention of SQL injection attack by dynamic analyzer and testing model. International Journal of Advanced Computer Science and Applications. 8(8): 209-214.

[14] N. M. Sheykhkanloo. 2017. A learning-based neural network model for the detection and classification of SQL injection attacks. International Journal of Cyber Warfare and Terrorism (IJCWT). 7(2): 16-41.

[15] X. Kuang, M. Zhang, H. Li, G. Zhao, H. Cao, Z. Wu and X. Wang. 2019. DeepWAF: detecting web attacks based on CNN and LSTM models. in Cyberspace Safety and Security: 11th International Symposium, CSS 2019, Guangzhou, China, December 1-3, 2019, Proceedings, Part II. pp. 121-136.

[16] A. Tekerek. 2021. A novel architecture for web-based attack detection using convolutional neural network. Computers & Security. Vol. 100.

[17] Z. Tian, C. Luo, J. Qiu, X. Du and M. Guizani. 2019. A distributed deep learning system for web attack detection on edge devices. IEEE Transactions on Industrial Informatics. 16(3): 1963-1971.

[18] R. Rawat and S. K. Shrivastav. 2012. SQL injection attack Detection using SVM. International Journal of Computer Applications. 42(13): 1-4.

[19] M. Zhang, B. Xu, S. Bai, S. Lu, and Z. Lin. 2017. A deep learning method to detect web attacks using a specially designed CNN. in Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14–18, 2017, Proceedings, Part V, pp. 828-836.

[20] Y. Pan, F. Sun, Z. Teng, J. White, D. C. Schmidt, J. Staples, and L. Krause. 2019. Detecting web attacks with end-to-end deep learning. Journal of Internet Services and Applications. 10(1): 1-22.

[21] C. Liu, J. Yang and J. Wu. 2020. Web intrusion detection system combined with feature analysis and SVM optimization. EURASIP Journal on Wireless Communications and Networking.

[22] C. Zhao, S. Si, T. Tu, Y. Shi, and S. Qin. 2022. Deep-Learning Based Injection Attacks Detection Method for HTTP. Mathematics. 10(16): 2914.

[23] X. Meng, Y. Liu, X. Gao, and H. Zhang. 2014. A new bio-inspired algorithm: chicken swarm optimization. in Advances in Swarm Intelligence: 5th International Conference, ICSI 2014, Hefei, China, October 17-20, 2014, Proceedings, Part I, pp. 86-94.

[24] D. Wu, S. Xu, and F. Kong; 2016. Convergence analysis and improvement of the chicken swarm optimization algorithm. IEEE Access. 4: 9400-9412.

[25] J. L. Castro, C. J. Mantas, and J. M. Benítez. 2002. Interpretation of artificial neural networks by means of fuzzy rules. IEEE Transactions on Neural Networks. 13(1): 101-116.

[26] F. Karim, S. Majumdar, H. Darabi, and S. Chen. 2017. LSTM fully convolutional networks for time series classification. IEEE access. 6: 1662-1669.