Materials Today: Proceedings 62 (2022) 4806–4813

Contents lists available at ScienceDirect

Materials Today: Proceedings

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

Web attack detection using deep learning models

J.I. Christy Eunaicy [a,*], S. Suguna [b]

[a] Thiagarajar College, Madurai 625016, India
[b] Sri Meenakshi Govt. Arts College for Women, Madurai 625016, India

ARTICLE INFO

Article history:
Available online 30 March 2022

Keywords:
Web attack detection
Machine learning
Web applications
ANN
CNN
RNN

ABSTRACT

Due to the network access and security vulnerabilities of web applications, web applications are often targets of cyber-attacks. Attacks against web applications can be extremely dangerous. A lot of damage has been done because of the vulnerability of the application, which lets them access the Web Application database. Monitoring web attacks and generating alarms when a challenge to an attack is detected. This work uses deep learning models (ANN, CNN & RNN) to detect web attacks automatically. To identify the time when the attack on the payload occurred, the work first analyses the web log information provided by the user. To make an attack prediction, the log information is pre-processed. Web-log information is pre-processed to remove duplicate values and missing values and to get the payload information. To encode the fields and normalize (Min-Max) that converts into unique format while predicting and the encoding value also applied. To construct the prediction model for the detection of web attacks, the pre-processed dataset is incorporated into the deep learning classifiers. In the performance evaluation, RNN provided 94% accuracy and 6% error rate, higher than other method.

Selection and peer-review under responsibility of the scientific committee of the International Conference on Innovative Technology for Sustainable Development.

1. Introduction

Web applications are commonly the easy target of cyber-attacks because they are network-accessible and have vulnerabilities. Monitoring web applications for attacks and alerting users when the attack is detected is the job of an intrusion detection system. Web traffic filtering is a highly complex and challenging task due to its dynamic nature. There are numerous attacks are available such as Man in the Middle, DDOS, Malware, Email Phishing attack etc. We mainly focus on the HTTP Based Web attacks and quickly learn that there are many challenges to overcome, including detecting unknown attacks, detecting false positives, identifying uncertain queries, and detecting abnormal behavior. By utilizing machine learning techniques, this work improves the detection capabilities of the HTTP attacks, with a special focus on reducing the number of false positives generated by this work when it is used to protect a web application, without reducing the number of attacks.

The proposed model (Fig. 1) provides a characterization of the issue by identifying different scenarios based on data availability. On the basis of the corresponding scenario, this work serves as a framework for experimenting with different AI algorithms. Throughout the paper, Section 2 summarizes the review of web application security and protection mechanisms. In Section 3, we described the experimental framework used in this study. In Section 4, presented the comparison-based results; in Section 5, we presented the conclusion what we have done in this work.

2. Review of literature

A study by Wang et al. [1] utilized authentication and crawler models to identify SQL injection vulnerabilities through simulated web attacks and response data analysis. Two experiments were also performed for this work. A comparison of the coverage of our tool with the three traditional scanners is the first step. The system we have developed was able to retrieve hidden pages, as well as have a larger coverage area compared to the other scanners. In Park et al.'s [2] Web application intrusion detection system (WAIDS), input validation attacks against web applications were detected using an anomaly intrusion detection model. The approach is based on identical structures and values of parameters in web applications. Web request data is used by WAIDS to generate intrusion detection profiles as part of normal system

* Corresponding author.
E-mail address: eunaicy@gmail.com (J.I. Christy Eunaicy).

https://doi.org/10.1016/j.matpr.2022.03.348
2214-7853/
Selection and peer-review under responsibility of the scientific committee of the International Conference on Innovative Technology for Sustainable Development.

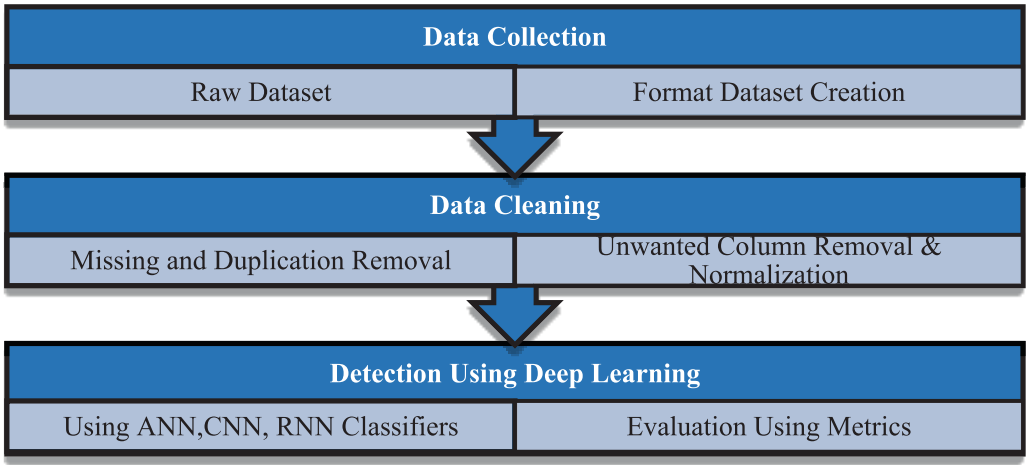

Fig. 1. Proposed model.

operations. A priority queue-based scheme was proposed by Lin et al. [3] as a way to enhance the quality of service for normal users. The method has been tested by implementing it on a server under heavy load. The experiments showed that our method is efficient in detecting malicious flooding attacks, and it can accurately analyze and classify network traffic. The detection method's processing time increases linearly with data size, so our scheme can be applied to servers with heavy loads to improve their performance. Sonewar et al. [4] examined XSS attacks and SQL injection attacks. SQL injection attacks are very common, in which data passing through the web application to the database server is altered so that the database contents are altered or revealed. XSS attacks, on the other hand, target web applications and use tricks to trick users, causing a security breach. The study considers both static and dynamic behavior in three-tier web applications to ensure security.

Tekerek et al. [5] proposed the signature-based and anomaly detection methods to prevent attacks based on HTTP requests. Das and colleagues [6] presented two phases. At first, a web host-based intrusion detection system called Hamming Edit Distance was applied. The matching process was based on the log files sent by the web layer. This phase was tested with a log file from the intranet server of the university. Using SQL injection, unauthorized access was successfully established. Next proposed a 'Query-based projected clustering' method for unsupervised anomaly detection as well as a 'packet arrival factor' method for intrusion detection. Jelodar et al. [7] combining web mining techniques with fuzzy logic to determine the probability of DOS attacks by evaluating two effective factors: hit counts and the time interval between requests to take appropriate measures based on the specific circumstances with Web Crawler-based Phishing Attack Detector (WC-PAD), Natheezhtha et al. [8] defined a three-phased method of detecting attacks. Web traffic, content, and URL that are used to determine whether a website is phishing or not. WCPAD's experimental analysis is based on data collected from real phishing incidents. The proposed WC-PAD was found to detect both phishing attacks and zero-day phishing attacks with an accuracy of 98.9% in experiments. Yang et al. presented [9] a novel web-crawling traces-tolerant approach to building baseline profiles and a mechanism to detect HTTP flooding anomalies (HTTP-sCAN). It has allowed HTTP-sCAN to be immune to the interfering effects of unknown web crawling traces and to detect all HTTP-flooding attacks. Sonewar [10] proposed categorizing web applications in three tiers both statically and dynamically as a means of detecting and preventing these attacks. Such attacks can be detected using a mapping model, in which requests are mapped to queries, and prevention logic can then be applied. A real-time online detection method based on the flow data analysis was proposed by Tial et al. [11] for detecting web attacks in real-time. The study initially constructed a framework for detecting attacks and a cache method based on hashing. Then, an attack detection method based on a multi-pattern matching algorithm is presented. Niu and Li [12] extract statistical features with a good classification effect to enrich the original data. Furthermore, we pre-trained the word embedding matrix using the word2vec model, which was used to obtain the input for the CNN-GRU model. Ito and Iyatomi present an efficient machine learning approach to solve these issues [13]. Using CLCNN with very large global max-pooling, we extract the feature of HTTP requests and identify them as normal or malicious. Under 10-fold cross-validation, the system achieved 98.8% accuracy and an average processing time of 2.35 ms per request. Liu et al. [14] proposed an integrated web intrusion detection system integrated with feature analysis and support vector machine (SVM) optimization. The characteristics of the common Web attacks are analyzed by using expert knowledge. In the results, the proposed system performs well in the detection capability and can detect WAF bypass attacks effectively.

3. Proposed scheme

Fig. 2 illustrates the flow of the proposed work. This figure represents the work flow of the proposed model: Data Collection, Data Cleaning, and Prediction phases. This includes gathering datasets, formatting the dataset, cleaning, and normalizing it, as well as creating a detection model with various deep learning algorithms.

3.1. Dataset

HTTP DATASET CSIC 2010 [11] is widely used for web intrusion detection systems. CSIC 2010 is a database containing thousands of automatically generated web requests, which were developed by the CSIC (Spanish Research National Council). The database is used for testing web attack protection. Data is generated automatically via HTTP traffic by an e-Commerce web application. In this data set, there are 36,000 normal requests and 25,000 anomalous requests. A wide variety of attacks are being conducted, including SQL injections, buffer overflows, information gathering, file disclosures, CRLF injections, XSS, server-side inclusions, parameter tampering's, and more. Table 1 showing the features are used in the dataset.

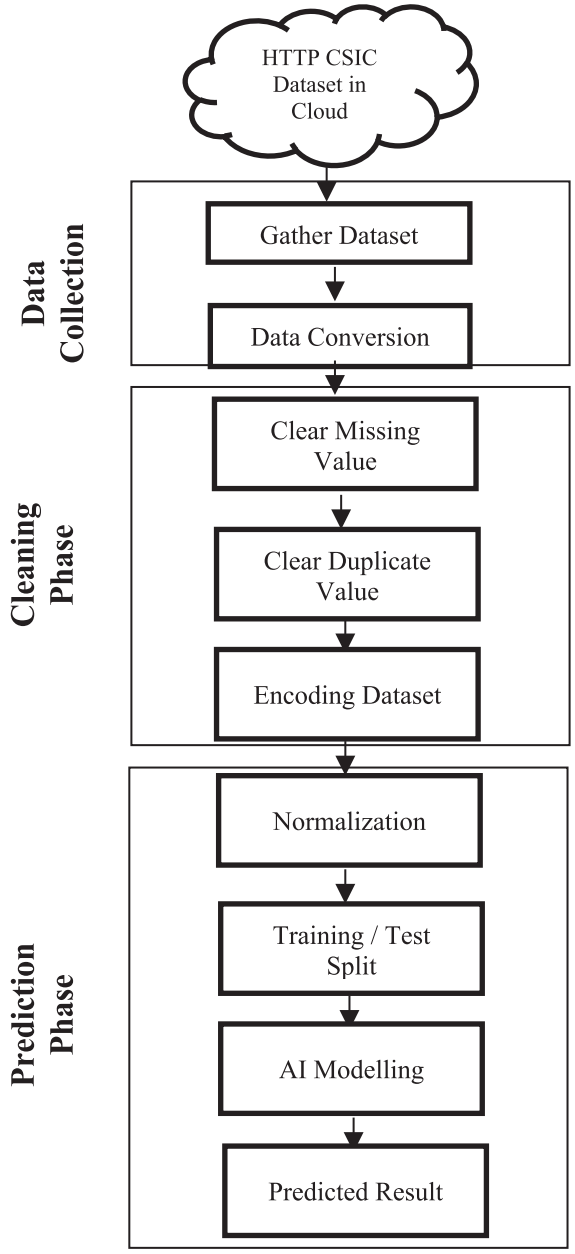

Fig. 2. Flow diagram.

3.2. Pre-processing

This phase involves cleaning and normalizing the raw dataset collected during the previous phase. The raw log data set is obtained and converted into CSV files. During the assessment process, this phase removes missing values, eliminates duplicate values, encrypts, and normalises the data to help detect web attacks.

3.2.1. Data conversion

Splitting raw text data to create CSV files with a Regex pattern and then saving it in CSV format. Fig. 3 depicts the formatted data for the dataset. This figure includes the user's log information and the fields.

3.2.2. Missing value removal

There are many reasons for missing values can occur, including corrupt data, failed loading, or incomplete extraction. Missing values are problematic when using a dataset for machine learning. They will make data visualization and analysis more difficult. Fig. 4 illustrates the visual representation of the missing values in the dataset.

Table 1
Features in the dataset.

No	Features	Description
1	Index	Reference No of the Data
2	Method	It refers to the HTTP methods, such as GET, POST, PUT
3	URL	Unified Resource Locator
4	Protocol	It refers to the set of rules during communication, such as TCP, UDP
5	User Agent	The User-Agent attribute of the HTTP headers is a string that allows network protocol peers to identify the web server's Operating System and browser.
6	Pragma	This Pragma HTTP/1.0 general-header may have various effects along the request-response chain
7	Cache-Control	HTTP header cache-control specifies the browser's caching policy in both client requests and server responses
8	Accept	Accept headers show what content types, expressed as MIME types, the client can understand
9	Accept-Encoding	According to the Accept-Encoding HTTP header, the client can understand the content encoding (usually a compression algorithm).
10	Accept Charset	Request type headers include HTTP Accept-Charset. Character set headers to specify what characters are acceptable for the response from the server.
11	Accept-Language	In the Accept-Language HTTP header, the client specifies what natural language and locale it prefers.
12	Host	The host is the third piece of information that you can use besides the IP address and port number to uniquely identify a Web domain.
13	Connection	The Connection general-header controls whether the network connection stays open after the current transaction finishes.
14	Content-Length	The Content-Length header shows how many bytes the recipient will receive in the message body.
13	Content-Type	The Content-Type representation header is used to indicate the original media type of the resource (before any content encoding is applied for sending).
14	Cookie	A cookie is an HTTP request header, i.e. used in the requests sent by the user to the server.
15	Payload	An HTTP request or response has a payload that includes headers, a URL, body content, and version and status information.
16	Label	Data that is Normal or attack

3.2.3. Duplication removal & column removal

Data cleaning processes often require the detection of duplicate data. In machine learning, non-duplicate values create identical patterns, so duplicates should be removed. There are no duplicates in this dataset. Table 2 shows the number of rows in the original dataset and in the cleaned dataset which is received after missing and duplicate values are removed from some columns (content Length, content Type, accept Language, connection).

3.2.4. Encoding dataset

For a machine learning model to be effective, the data should be clean and ready for use. Ordinal encoding is used when the variables in the data are ordinal, converting each label into an integer value and representing the sequence of labels in the data. The fields of the cleaned dataset and their encoded values are shown in Table 3.

3.3. Normalization

Data are normalized by converting them into non-duplicate patterns. Transformation of data is the process of converting the source data into a format that can be processed effectively.

```
GET
http://localhost:8080/tienda1/publico/anadir.jsp?id=2&nombre=Jam%F3n+Ib%E9rico&precio=85&cantidad=%2
7%3B+DROP+TABLE+usuarios%3B+SELECT+*+FROM+datos+WHERE+nombre+LIKE+%27%25&B1=A%
F1adir+al+carrito HTTP/1.1
User-Agent: Mozilla/5.0 (compatible; Konqueror/3.5; Linux) KHTML/3.5.8 (like Gecko)
Pragma: no-cache
Cache-control: no-cache
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Encoding: x-gzip, x-deflate, gzip, deflate
Accept-Charset: utf-8, utf-8;q=0.5, *;q=0.5
Accept-Language: en
Host: localhost:8080
Cookie: JSESSIONID=B92A8B48B9008CD29F622A994E0F650D
Connection: close
```

Fig. 3. Raw dataset.

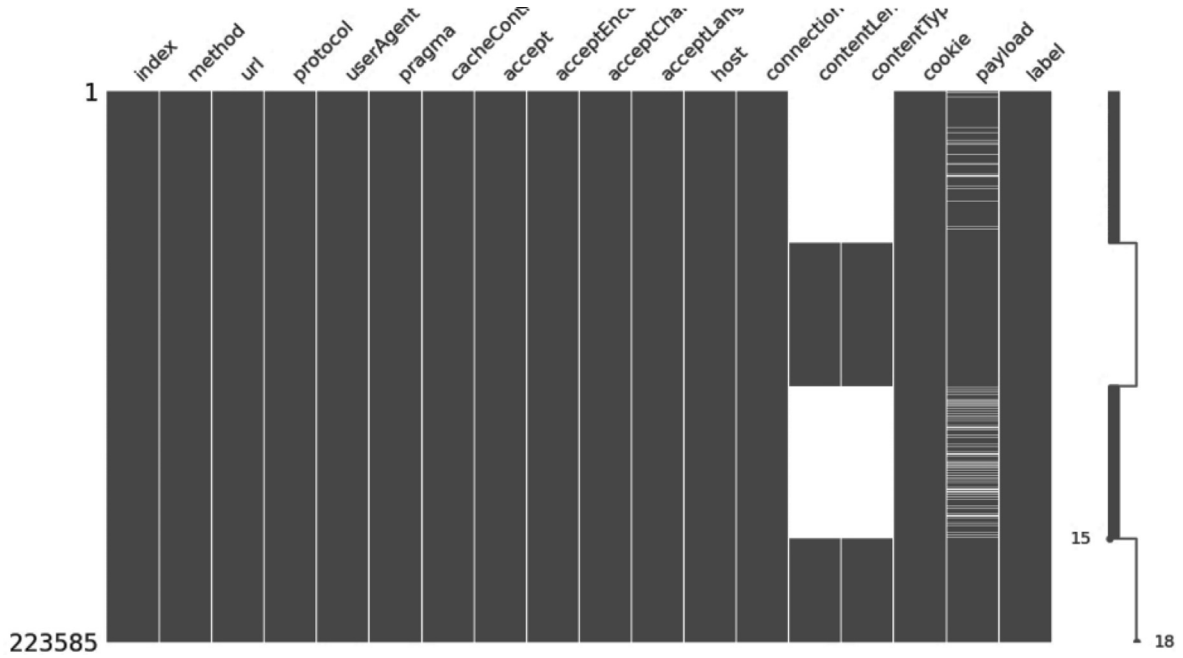

Fig. 4. Missing values.

Table 2
Cleaned set.

Original data (Rows)	Cleaned data (Rows)
223,585	100,135

Table 3
Encoded values.

Features	Value	Encoded Value
METHOD	POST, GET, PUT	1, 2, 3
URL	http://localhost:8080/tiendal1/publico/anadir.jsp https://localhost:8080/	1 to N
HOST	http://localhost:8080/ https://localhost:9090/	1 to N
COOKIE	JSESSIONID = B92A8B48B90CD29F622A994E0F650D JSESSIONID = 51A7470173188BB9939447F2283059E4	1 to N
PAYLOAD	id = 2, nombre = Jam%F3n + Ib%E9rico	1 to N
LABEL	Anomaly and Normal	0 & 1

Normalization is concerned primarily with minimizing or even eliminating duplicate data. Since this is an essential issue, it is becoming increasingly difficult to maintain data in relational databases that store identical data in more than one place. A process called Minimax Normalization is used to achieve this. Data is linearly transformed. The data is scaled from 0 to 1. It is calculated as follows:

$$v' = \frac{v - minF}{maxF - minF}(new\_max_F - new\_min_F) + new\_min_F \quad (1)$$

3.4. Web attack detection using classifiers

In the detection phase, deep learning algorithms are used to develop a prediction model using classifiers to identify the web attacks. A prediction model is constructed using the following deep learning strategies.

3.4.1. Artificial neural network (ANN)

Neurons are a technology that mimics the human brain based on biological principles. A neuronal network is designed by cloning

neurons in the brain and connecting them as neurons, nodes, dendrites, and synapses, which act as the biological nervous system. Each arc has a weight associated with it. After the contributions are applied, an enactment is performed on these arcs to change the loads and get the best arrangement of outputs. Artificial intelligence is primarily an application of neural organization. ANN is suitable for reducing the time it takes to perform perceptual and acknowledgment tasks. To characterize a set of desired data sources, a neural network takes advantage of a problem's nonlinearity. This method of grouping is an important part of AI, and it is used in many fields, such as pattern recognition, data mining, and so on. ANNs are built from three layers:

- **Input Layer** – The input layer of the ANN classifier model contains the selected feature set.
- **Hidden Layer** – The hidden layer obtains the information from the input layer for the process.
- **Output Layer**– The output layer receives the results from the processed information in the hidden layer and provides desired prediction results.

The dataset is the starting point for characterization. Training and testing data sets are separate. While the training set is used to determine the accuracy of the classifier, the test set is used for learning the neural organization. In artificial intelligence, neural organizations are helpful because of their ability to change the design of the organization and adjust the weight, which enhances their learning capabilities.

Algorithm 1: ANN

Algorithm ANN
Input: Cleaned & Normalized Dataset
Output: Predicted Result
BEGIN
1. Get Feature set & Class Label from the Training Set
2. Initialize Input, Output and Hidden Layers
3. Initialize the Weight 'W' Parameter
4. Compute the Bias 'b'
5. Then the kernel is initialized with kernel_initializer (uniform) and RELU activation function (AF)
6. The dense is performed for the hidden layer 1 & 2 with units, kernel and AF (64, uniform,relu)
7. Calculate the Loss function with Accuracy & Binary Cross Entropy
8. Fit the model for Training Set and applied Adam Optimizer
9. Predict the class label with trained model
10. Return predicted result

END

Pre-processed datasets are then fed into a Feature Normalization algorithm, which on application, returns normalized results. To normalize the features, a training set with class labels is produced. There are two training sets with the class in the ratio of 80% and 20%. Table 4 shows the ANN classifier model with training and testing sets. In this model, the ANN classifier model starts with an input layer comprising input, kernel, and "relu" activation. Activation (relu), kernel, and input (11, 100136) comprise the hidden layers. To get the predicted class labels, the 'Uniform' Kernel is used with the 'ADAM" optimizer and the binary_crossentropy() loss function.

Table 4
ANN parameters.

ANN-parameters	Values
Learning rate	0.2
Momentum	0.7
Decay	Learning Rate/Epoch
Optimizer	Adam
Epoch	60
Loss	Binary_Cross Entropy
Metrics	Accuracy
Model	Dense-1(64), Dropout (0.1), Dense-2(64), Activation='Relu', Kernel='Uniform'

3.4.2. Convolution neural network (CNN)

Classification is an important part of data mining, which uses supervised learning to analyse large datasets and predict unknown classes. Alternatively, neural networks can be used to classify data. Neural networks are nonlinear and are capable of modeling complex interactions. Statistical analysis and classification rules are made possible by neural networks, which evaluate posterior probabilities. A modular structure of convolutional neural networks (CNN) was used to construct this classifier level. According to the empirical results, even conventional CNN has more potential than traditional methods. Among the many types of neural networks, CNNs have a unique design that identifies complex features in attack data. A CNN can be used for robotics, image recognition, attack detection, and self-driving cars. CNNs are typically used for ordering content from a range of perspectives. The model needs only to be fed with attack information. Artificial neural networks and CNNs are driven by the brain. Similar to how the human eye recognizes highlights to distinguish objects, CNN can detect attacks using features. The initial convolution and pooling of the weights, training, and evaluating the initial and next convolutional layers must be performed in order to construct a multilayer convolutional network from the model generated so far.

Algorithm 2: CNN

Algorithm CNN
Input: Cleaned & Normalized Dataset
Output: Predicted Result
BEGIN
1. Load the Dataset and Split it to in Train and Test Set
2. Initialize Input, Output and Convolutional Layers
3. Initialize the Max Polling (2) and Convolutional Layer (2) (11 X 100135)
4. Apply Batch Normalization (2) and Initialize Random weight
5. Apply Flatten, Dense and Dropout layers
6. Apply Adam optimizer and soft-max activation with 100 Epochs and 32 Batch Size
7. Fit the model for Training Set
8. Predict the class label with trained model
9. Return predicted result

END

Table 5
CNN parameters.

CNN parameters	Shape	Values
Conv1D	11 × 100135	512
Batch Normalization	–	128
Max Pooling 1D	–	128
Conv1D	11 × 5021	256
Batch Normalization	–	64
Max Pooling 1D	–	64
Flatten	–	256
Dense	–	64 (Relu)
Dropout	–	0.2
Dense	–	16(Relu)
Output Layer	11 × 1	2

Others. Adam **Optimizer**, **Metrics** = Accuracy, **Loss** = Binary Cross Entropy, **Activation** = Soft-max, **Epochs** = 100, **Batch Size** = 32.

A CNN classifier model is constructed from the normalized features set shown in Table 5. With input 512 and with a kernel size of 11 × 100135, CNN began modelling with the input layer. There is a first convolution layer with an input size of 100,135 and batch normalization of 128. Input 128 with batch normalization of 64 is used to construct the max-pooling layer. Next, a second convolution layer is conducted with a kernel size of 11 × 5021 and a maximum of 64 pools. To get the 11 × 1 output, the 'relu' activation is performed densely and with dropout. With the Uniform Kernel Output Layer, the predicted class label, is obtained by optimizing the classification label using the Adam optimizer and 'binary_crossentropy ()' as the loss function.

3.4.3. Recurrent neural network (RNN)

RNN's are design types of artificial neural networks that work well for subjectively grouping data sets. Cyclic associations enable a particular RNN to more likely model arrangement information than a feed-forward network. During training, loops and cycles are crucial because they preserve data. Through cyclic associations and loops, starting with one phase, then moving onto the next, data can be passed through the organization. To predict the yield of a layer, RNN's save the yield of each layer and bring it back to the contribution to the request. The process consists of three phases. First, it pushes the data through a layer called LSTM. The next step is to compare the optimal value with the prediction using the loss function (LF). Loss functions are a measure of how well a model performs. A lower loss function indicates a better model.

Algorithm 3: RNN..

Algorithm RNN
Input: Cleaned & Normalized Dataset
Output: Predicted Result
BEGIN
1. Get Feature set & Class Label from the Training Set
2. Initialize Input, Output and LSTM Layers
3. Initialize the random Weight 'W' Parameter
4. Apply three LSTM and three Drop out layers (0.2) with one dense layer
5. Apply learning rate (0.2), Epochs (100) and Batch Size (32)
6. Fit the model for Training Set and applied Adam Optimizer and soft-max activation
7. Predict the class label with trained model
8. Return predicted result

END

Table 6 shows the RNN Classifier Model. Using the web attack dataset, the RNN classifier model is designed. Training and test sets are split to construct the network layer. The input layer is built with 11 × 100135 input units and a dropout value of 0.2 as the output layer. LSTM back-propagation is used to construct the dense layer. To get the predicted class label, the optimizer (ADAM) is used along with the loss function.

4. Result and discussion

Results and performance metrics are included in this section of the web attack detection model and it shows that the proposed classifier provides better result than other one. The Proposed Framework with the environment of Intel Core-i7, 1 TB Hard disk, 16 GB RAM, Windows-7 64-bit Operating System. In development of python with Anaconda Environment IDE. In Table 7, the accuracy of deep learning strategies is evaluated and the error rate is computed. In the table, it can be seen that the proposed RNN model provides high accuracy rates while reducing error rates.

Table 6
RNN classifier model.

RNN parameters	Shape	Values
Input Layer	11 × 100135	–
LSTM-1	11 × 100135	–
Drop Out-1	–	0.2
LSTM-2	11 × 50021	–
Drop out-2	–	0.2
LSTM-3	11 × 25022	–
Drop out-3	–	0.4
Dense	–	64 (Relu)
Output Layer	11 × 1	2

Others. Adam **Optimizer**, **Metrics** = Accuracy, **Loss** = Binary Cross Entropy, **Activation** = Soft-max, **Epochs** = 100, **Batch Size** = 32, Learning Rate = 0.2, Decay = 1e−6

4.1. Accuracy & error rate

The prediction accuracy and error rate of the proposed model is shown in Fig. 5. It shows RNN gives high accuracy and low error rate while detection than other methods.

4.2. Precision & Recall

Recall is an amount of related attacks retrieved in the model divided by the total amount attacks, while precision is the count of relevant attacks divided by the total count of attacks in the model Fig. 6 shows the precision and recall.

4.3. PPV & NPV

The positive and negative predictive values are the proportions of +ve and −ve results in prediction and tests that are TP & TN results, respectively. Fig. 7 shows the PPV and NPV.

In this work [16] provides machine learning based detection using J48 Decision tree with HTTP CSIC 2010 along 94% accuracy.

Table 7
Metrics.

Method	Accuracy	Error Rate	Precision	Recall	PPV	NPV
ANN	77	23	71	99	71	97
CNN	85	15	86	95	86	84
RNN	94	6	98	94	98	72

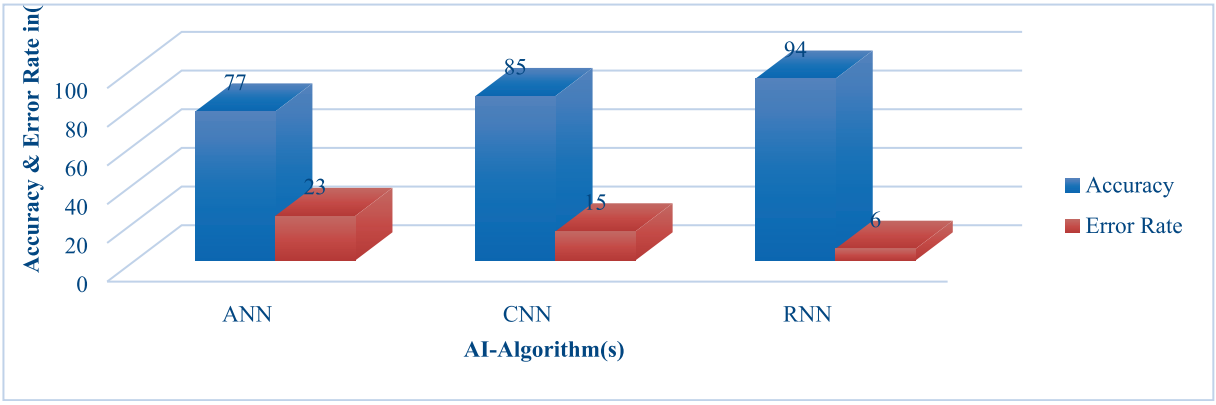

Fig. 5. Accuracy & error rate.

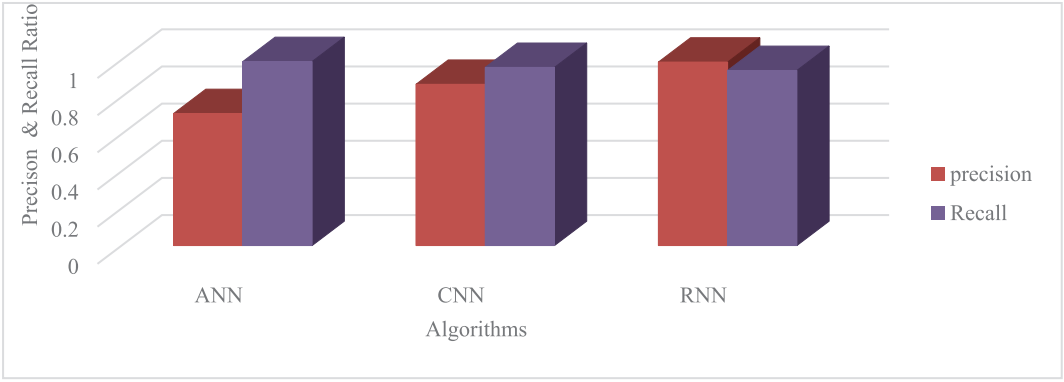

Fig. 6. Precision and recall.

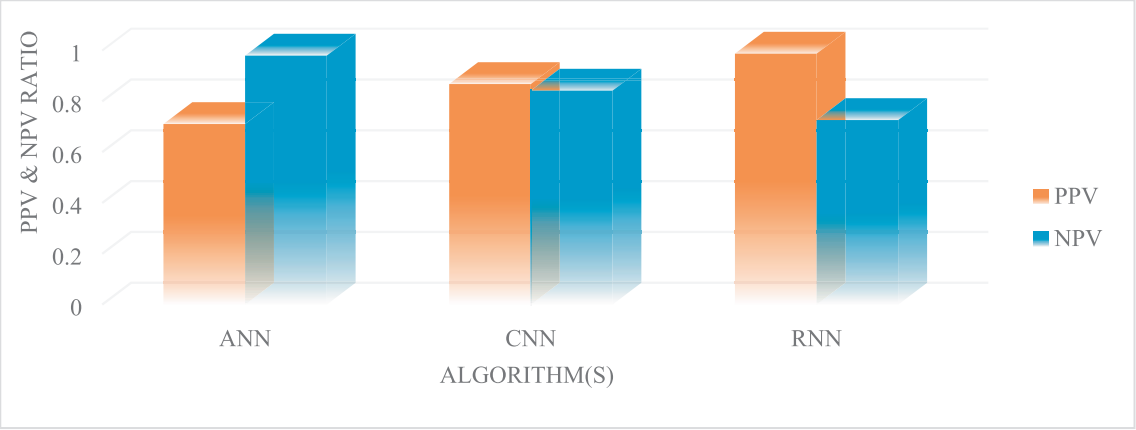

Fig. 7. PPV vs NPV.

This work [17] shows the anomalous HTTP payloads detection method with 85% accuracy. Our proposed model provides the 94% accuracy with AI Intelligence.

5. Conclusion

This work described the model for web attack detection with deep learning strategies to verify the model's ability to detect attacks. The proposed scheme used log data as an input dataset, and the dataset was then processed through the phases of the proposed scheme to perform detection. A dataset is preprocessed in order to remove duplicate values and missing values. The cleaned dataset was normalized in order to create a unique format. The formatted dataset is added to deep learning strategies to build a detection model of web attacks. Classifiers such as ANN, CNN, and RNN are used in the detection model construct. When detecting attacks such as SQL injections, XSS the RNN classifier outperforms with high accuracy and low error rates. In the performance evaluation, RNN provided 94% accuracy and 6% error rate, higher than any other method.

CRediT authorship contribution statement

J.I. Christy Eunaicy: Conceptualization, Methodology, Software, Data curation, Writing – original draft. **S. Suguna:** Visualization, Investigation, Supervision.

Declaration of Competing Interest

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

References

[1] X. Wang, L.H. Wang, G. Wei, D. Zhang, Y. Yang, Hidden web crawling for SQL injection detection, in: 2010 3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT), 2010, pp. 14–18, https://doi.org/10.1109/ICBNMT.2010.5704860.

[2] Y. Park, J. Park, Web application intrusion detection system for input validation attack, in: 2008 Third International Conference on Convergence and Hybrid Information Technology, 2008, pp. 498–504, https://doi.org/10.1109/ICCIT.2008.338.

[3] C. Lin, J. Liu, C. Lien, Detection method based on reverse proxy against web flooding attacks, in: Eighth International Conference on Intelligent Systems Design and Applications, 2008, pp. 281–284, https://doi.org/10.1109/ISDA.2008.72.

[4] P.A. Sonewar, S.D. Thosar, Detection of SQL injection and XSS attacks in three-tier web applications, in: International Conference on Computing Communication Control and Automation (ICCUBEA), 2016, pp. 1–4, https://doi.org/10.1109/ICCUBEA.2016.7860069.

[5] A. Tekerek, C. Gemci, O.F. Bay, Development of a hybrid web application firewall to prevent web-based attacks, in: 2014 IEEE 8th International Conference on Application of Information and Communication Technologies (AICT), 2014, pp. 1–4, https://doi.org/10.1109/ICAICT.2014.7035910.

[6] D. Das, U. Sharma, D.K. Bhattacharyya, A web intrusion detection mechanism based on feature-based data clustering, in: IEEE International Advance Computing Conference, 2009, pp. 1124–1129, https://doi.org/10.1109/IADCC.2009.4809172.

[7] H. Jelodar, J. Aramideh, Presenting a pattern for detection of denial-of-service attacks with web mining technique and fuzzy logic approach, in: 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014, pp. 156–160. doi: 10.1109/ICCICCT.2014.6992947.

[8] T. Nathezhtha, D. Sangeetha, V. Vaidehi, WC-PAD: web crawling based phishing attack detection, in: International Carnahan Conference on Security Technology (ICCST), 2019, pp. 1–6, https://doi.org/10.1109/CCST.2019.8888416.

[9] J. Wang, M. Zhang, X. Yang, K. Long, J. Xu, HTTP-sCAN: detecting HTTP-flooding attack by modeling multi-features of web browsing behavior from noisy web-logs, China Commun. 12 (2) (2015) 118–128, https://doi.org/10.1109/CC.2015.7084407.

[10] P.A. Sonewar, N.A. Mhetre, A novel approach for detection of SQL injection and cross-site scripting attacks, in: International Conference on Pervasive Computing (ICPC), 2015, pp. 1–4, https://doi.org/10.1109/PERVASIVE.2015.7087131.

[11] J. Tian, H. Zhu, X. Li, Z. Tian, Real-time online detection method for web attack based on flow data analysis, in: 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), 2018, pp. 991–994, https://doi.org/10.1109/ICSESS.2018.8663848.

[12] Q. Niu, X. Li, A high-performance web attack detection method based on CNN-GRU model, in: 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2020, pp. 804–808, doi:10.1109/ITNEC48623.2020.9085028.

[13] M. Ito, H. Iyatomi, Web application firewall using character-level convolutional neural network, in: 2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA), 2018, pp. 103–106, https://doi.org/10.1109/CSPA.2018.8368694.

[14] C. Liu, J. Yang, J. Wu, Web intrusion detection system combined with feature analysis and SVM optimization, J. Wirel. Com Network 2020 (2020) 33, https://doi.org/10.1186/s13638-019-1591-1.

[16] S. Sharma, Pavol Zavarsky, Sergey Buakov, Machine learning based intrusion detection system for web-based attacks, in: 2020 IEEE 6th Intl Conference on Big Data Security on Cloud, 2020, pp. 227–230.

[17] Z. Zhang, R. George, K. Shujaee, Efficient detection of anomolous HTTP payloads in networks, SoutheastCon 2016 (2016) 1–3.

Further reading

[15] HTTP DATASET CSIC 2010. <https://www.isi.csic.es/dataset/> (Accessed 02 July 2020).