Journal of Al-Azhar University Engineering Sector

Vol. 18, No. 19, October 2023, 869 - 883

Multi-Class Intrusion Detection System using Deep Learning

Sara M. Mohamed[1,*], Mohamed A. Rohaim[1]

Systems & Computers Dept, Faculty of Engineering, Al-Azhar University, Cairo, Egypt.

* Correspondence: mahroussara@gmail.com

Citation:

S. M. Mohamed , M. A. Rohaim "Multi-Class Intrusion Detection System using Deep Learning", Journal of Al-Azhar University Engineering Sector, vol. 18, pp. 869-883, 2023

Received: 24 May 2023

Accepted: 19 July 2023

DoI:10.21608/auej.2023.213003.1375

ABSTRACT

Web applications are a critical means of accessing information in today's world. However, as the number of internet users continues to grow rapidly, cybersecurity has become a major concern. In this study, a deep learning-based approach to detect web attacks is proposed. Our system explores incoming requests, categorizing them as either normal or attacks, and further identifies the type of attack. The approach is evaluated on three different datasets (ECML-PKDD, HTTPPARAM, and CSIC-2012) and used four classification algorithms (Bi-LSTM, LSTM, RNN, and CNN). The Bi-LSTM algorithm achieves high accuracy with the ECML-PKDD and HTTPPARAM datasets (90.6% and 99.66%, respectively), while the CNN algorithm performs best with the CSIC-2012 dataset, achieving an accuracy of 99.28%.

This research provides a valuable contribution to the field of web security and has practical applications for companies and website owners who need to protect their data from potential attacks, making it a powerful tool in the fight against cybercrime.

KEYWORDS: Web Application attacks, Cyber Security, deep learning algorithms, multi-classification.

نظام كشف التسلل متعدد الفئات باستخدام التعلم العميق

ساره محروس محمد[1]*، محمد على رحيم[1]

[1] قسم النظم والحاسبات , كلية الهندسة , جامعة الازهر, القاهرة , مصر

*البريد الاليكتروني: mahroussara@gmail.com

الملخص

تعد تطبيقات الويب وسيلة مهمة للوصول إلى المعلومات في عالم اليوم. ومع ذلك ، مع استمرار نمو عدد مستخدمي الإنترنت بسرعة ، أصبح الأمن السيبراني مصدر قلق كبير. في هذه الدراسة ، تم اقتراح نهج قائم على التعلم العميق للكشف عن هجمات الويب. يستكشف نظامنا الطلبات الواردة ويصنفها على أنها إما عادية أو هجمات ، كما يحدد نوع الهجوم. تم تقييم النهج على ثلاث مجموعات بيانات مختلفة (ECML-PKDD و HTTPPARAM و CSIC-2012) واستخدمت أربعة خوارزميات تصنيف (Bi-LSTM و LSTM و RNN و CNN). تحقق خوارزمية Bi-LSTM دقة عالية مع مجموعات بيانات ECML-PKDD و HTTPPARAM (90.6٪ و 99.66٪ على التوالي) ، بينما تعمل خوارزمية CNN بشكل أفضل مع مجموعة بيانات CSIC-2012 ، محققة دقة 99.28٪. يقدم هذا البحث مساهمة قيمة في مجال أمان الويب وله تطبيقات عملية للشركات ومالكي مواقع الويب الذين يحتاجون إلى حماية بياناتهم من الهجمات المحتملة ، مما يجعلها أداة قوية في مكافحة الجرائم الإلكترونية.

الكلمات الرئيسية: هجمات تطبيقات الويب ، الأمن السيبراني ، خوارزميات التعلم العميق ، التصنيف المتعدد.

1. INTRODUCTION

Over the past few years, there has been significant growth in Internet usage. Internet World data [1] show that from 2013 to 2023, the number of Internet users rose by 13.9%. Our lives are increasingly reliant on the Internet, from making appointments for medical exams to doing our shopping online. These public-facing services have significantly increased the number of web application assaults, making prompt identification and prevention necessary. Ransomware will affect 66% of the organizations examined in 2021, up from 37% in 2020, per a Sophos report [2]. For the second year in a row, web application assaults are ranked third in the cyber domain by the ENISA Threat Landscape Study (ETL) of 2020 [3].

Security Systems called intrusion detection systems (IDS) [4] keep an eye out for dangers or malicious behavior on a computer network or its hosts. They alert the system administrators when they find it [8, 9]. Alerting can take many different forms, such as notifying an administrator via security dashboards to take the appropriate measures or logging assaults in log files. The ideal Intrusion Detection System should be uncomplicated, rapid, and accurate. However, as no detection technique is 100% accurate, an IDS cannot offer total security. False positives occur when regular access is incorrectly seen as a threat, while false negatives occur when an actual attack is missed by the IDS systems. Because they are sent to the administrator for examination, false positives are permissible; however, if there are too many false positives, it might be difficult for them. False negatives, however, are never examined by the administrator since they were mistakenly classified as non-malicious threats. As a result, most businesses continuously tweak their intrusion detection systems to assure nearly zero false negatives and a certain number of false positives.

Deep learning techniques enable the implementation of an anomaly detection system that can learn from training (labeled) data and make decisions based on test (unlabeled) data. Extracting features from the raw data and choosing which features to use for classification are two difficulties that must be overcome in traditional machine learning algorithms. Additionally, cutting-edge techniques like deep learning allow feature extraction and feature reduction to be integrated directly into the models, avoiding the need for feature engineering operations [5].

For IDS to determine whether the HTTP Request traffic contains an attack, the request's content must be examined. Text-based fields such as "protocol," "method," "content-type," "path," "URI," "content-length," "body," and "cookie," are part of HTTP requests. Some web security studies [6], [7], [8] only use payloads, whereas others [9], [10] only use URL-based features. Client payloads are visible and subject to length constraints when utilizing the GET method. Contrarily, client payloads submitted to the web server via the POST method are saved in the body of the HTTP request and are therefore delivered unconstrained and undetected. Therefore, when delivering lengthy or delicate inputs, such as permission data, the POST method is favored. It is hard to identify web assaults provided via payloads in the POST method simply using URL-based models because payloads are absent from the URL.

This paper aims to utilize deep learning techniques to classify multiple types of attacks in HTTP traffic, considering the number of classes in the datasets. Furthermore, the experimental results are compared with those of others. The rest of this paper is organized as follows: The second section discusses related work, the third section outlines the Methodology, and the fourth section describes the experiment and results. Finally, in the final section, the paper concludes with a summary.

2. RELATED WORK

Aref Shaheed [12] Developed WAF (Web Application Firewall) for preventing web attacks, he extracted four features from data (length of request, percentage of characters allowed, percentage of special characters, and attack weight) and use 4 types of data set (HTTPPARAM, CSIC-2010, Hypered dataset, and real web server logs), and also he used various classification algorithms that work more efficiently on binary classification cases, such as Linear Regression, Decision Tree, and Naive Bayes but he focus on Naive Bayes algorithm. The accuracy is 99.4 for CSIC-2010 and 97.91% for HTTPPARAM.

Sharma et al [13] depend on Seven features that were extracted from the incoming request using features engineering. It applied preprocessing techniques to the CSIC-2010 dataset to detect attack requests. To address the issue of lacking features, and then tested the efficiency of three classification algorithms (J48 from the decision tree, One Rule (OneR) from the rule system, and Naïve Bayes from Bayesian ML models), the J48 decision tree algorithm provided the highest True Positive rate, Precision, and Recall.

Abdelrahman S. Hussaini [14] used RNN Architecture and LSTM algorithm to detect two types of attack (SQLi and XSS script), the process of extracting features is done automatically because LSTM is one of the deep learning techniques, Dataset is collected randomly from Kaggle about (SQLi and XSS) to represent attack value and used a normal password to represent normal value, also used only parameter value of the payload. The system reads the most recent HTTP request (one request) from the log file in the Apache web server for XAMPP, and then the model predicts whether the request was malicious (SQL Injection or XSS attack) or not. The model achieved an accuracy of 99.3%.

WENCHUAN [15] a convolutional neural network was used to extract the features of the URL To detect malicious URLs. He used a malicious keyword as a dictionary to match each word in the URL with this world then embedded each character from the original URL into an allow-dimensional vector where each keyword is embedded as a unique world and the rest of the URL is embedded as a letter. To process the obtained feature sequences, the GRU is used as a pooling layer. Chose 407,212 different URLs which originated from a well-known Chinese Internet security company as the training set. The proposed model achieves 99.6% accuracy.

Adem TEKEREK [16]: using a convolutional neural network for web-based attack detection depends on two steps. The creation of dictionaries comes first, followed by the creation of matrixes. The Bag-of-Words (BoW) approach is used to calculate the frequency of identical payloads in various HTTP requests. Every word count is also considered as a feature. Utilizing the CSIC-2010 v2 dataset, a detection rate of 97.07% for binary classification was attained.

HACER KARACAN [17] Using deep learning algorithms and data augmentation, conducted a binary classification on a real-world dataset, the ECML-PKDD dataset, and the CSIC-2010 dataset, Also made multi-classification on the ECML-PKDD dataset and real-world dataset. The augmentation technique provides an average of 6.52% improvement, and the Bi-LSTM algorithm achieved the highest result equal to 93.91% in the ECML-PKDD dataset for multi-classification.

3. METHODOLOGY

Most of the studies discussed in the related work section concentrate on one or two types of attacks, such as SQLi and/or XSS attacks. Our research, on the other hand, investigates a variety of Types of attacks {Structure Query Language injection (SQLi), Command injection (CMDi), Carriage Return and Line Feed injection (CRLFi), cross-site script (XSS), Server Side Include (SSI)), Lightweight Directory Access Protocol injection (LDAPi), Xml path Injection (XPathi), Path-traversal, Bufferoverflow, and FormatString}.

3.1. DATASETS

The header parts of HTTP requests are removed first in the preprocessing. Because the header portions in the HTTP requests dataset are generally the same, they reduce the size of the data by removing unnecessary data and eliminating data duplication in datasets. In datasets, the payload portions of the HTTP request are chosen. The most important reason for choosing the payload segment is that the vast majority of Web attacks involve manipulating payloads. Also, in the case of the post method the payload is got from the request body using Python script. All datasets are organized as two columns (payload and label that refer to a type of attack) in CSV file format. our proposed model is applied to three datasets:

ECML-PKDD DATASET: In response to a web traffic analysis challenge, the combined 18th and 11th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases [19] inspired the creation of PKDD-2007. The dataset contains benign, normal, and abnormal (various types of attacks), with a total of 50116 instances. In addition, the distribution of all the eight classes is shown in **Table 1**.

CSIC-2012 DATASET: As part of the Torpedo framework, the CSIC-2012 dataset was presented. The framework was used to generate labelled web traffic for the purposes of evaluating and testing web-attack detection systems [18]. The dataset contains benign, normal, and abnormal (various types of attacks), with a total of 24318 instances. Moreover, the distribution of all the nine classes as shown in **Table 1**.

HTTPPARAM DATASET: Several freely available sources were used to create the HTTPPARAM Dataset [20]. This data set contains 31,067 web request URI payloads, including payload length and payload labels. The dataset contains 5 classes of benign, normal, and abnormal (different types of attacks). Distributed as shown in **Table 1**.

Table 1. Number of classes inside all datasets.

Data Set / Label	ECML-PKDD		HTTPPARAM		CSIC-2012	
	Count	**Percentage (%)**	**Count**	**Percentage (%)**	**Count**	**Percentage (%)**
Valid	35006	53.83%	19304	62.14%	8182	33.65%
SQLi	2274	4.54%	10852	34.93%	10000	41.12%
XSS	1825	3.64%	532	1.71%	4748	19.52%
Path-traversal	2295	4.58%	290	0.93%	N/A	0%
CMDi	2302	4.60%	89	0.29%	N/A	0%
Xpathi	2279	4.55%	N/A	0%	173	0.71%
LDAPi	2279	4.55%	N/A	0%	73	0.30%
SSI	1856	3.70%	N/A	0%	386	1.59%
Bufferoverflow	N/A	0%	N/A	0%	396	1.63%
CRLFi	N/A	0%	N/A	0%	319	1.31%
FormatString	N/A	0%	N/A	0%	41	0.17%
Total	**50116**	**100%**	**31067**	**100%**	**24318**	**100%**

3.2. Proposed Model to Classify Attacks

The proposed model is capable of distinguishing between malicious and benign URLs and can classify malicious URLs into various attack types based on the types of datasets used. Deep learning algorithms, such as Bi-LSTM, LSTM, CNN, and RNN, were utilized to handle large amounts of data with high accuracy. The feature selection and extraction process can be performed automatically.

The model comprises two phases: the data preparation phase and the model building phase, as depicted in Fig 1.

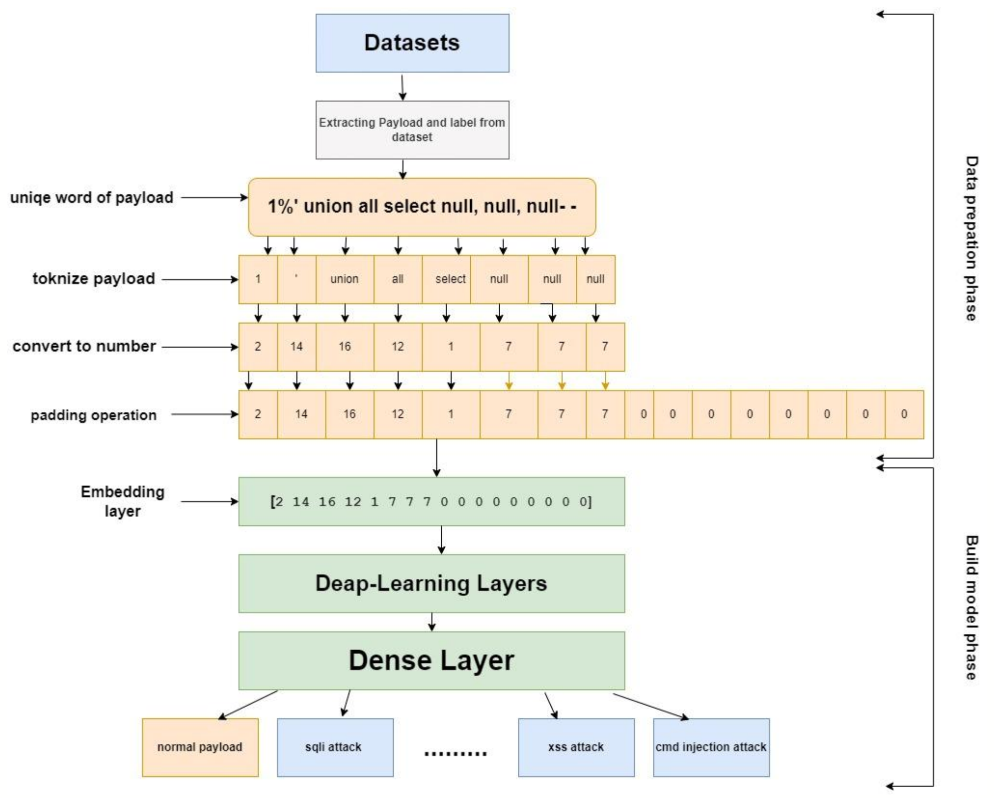


Fig.1. Proposed model for classifying attack types

3.2.1. Data Preparation Phase: The data preparation phase includes the following steps as shown in the **Fig.1**:

1) Unique words extraction: It is the extraction and counting of unique words from text columns (payload) for use in tokenization and deep learning models.

2) Tokenization: Tokenization is the process of splitting strings into tokens in order to prepare words for conversion to integer numbers, as our model can only understand integer sequences. For example, the URL payload **"1%' union all select null, null, null- -"** in the HTTPPARAM dataset at row (11004) would look like below after tokenization :(1, ', union, all, select, null, null, null).

3) Convert tokens to integers: After tokenization, the tokens were **c**onverted to integer numbers, as shown below, so that they can be fed into a deep-learning model.

[2, 14, 16, 12, 1, 7, 7, 7]

The number **2** corresponds to the token (**1**), the number **14** corresponds to the token ('), the number **16** corresponds to the token (**union**), the number 1**2** corresponds to the token (**all**), the number **1** corresponds to the token (**select**), the number **7** corresponds to the token (**null**), and all these tokens are indexes in the whole dataset words.

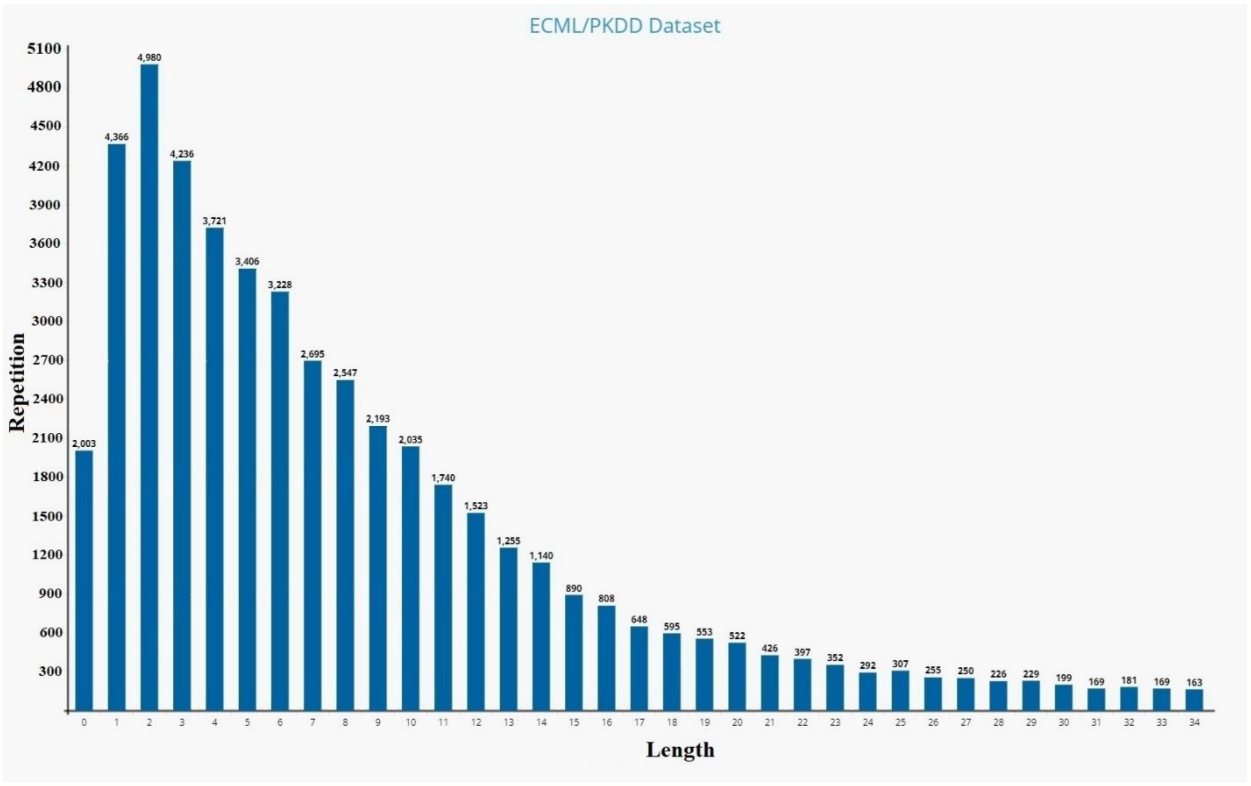
4) Sequence padding: Pad sequences refer to making all of the lists sequences the same length. For example, if the length is 17 a trailing padding of zeros will be added to the tokens sequence if its length is less than 17 as follow.
[2 14 16 12 1 7 7 7 0 0 0 0 0 0 0 0 0]

To determine the max length in padding operation, some strategies are put as follow:

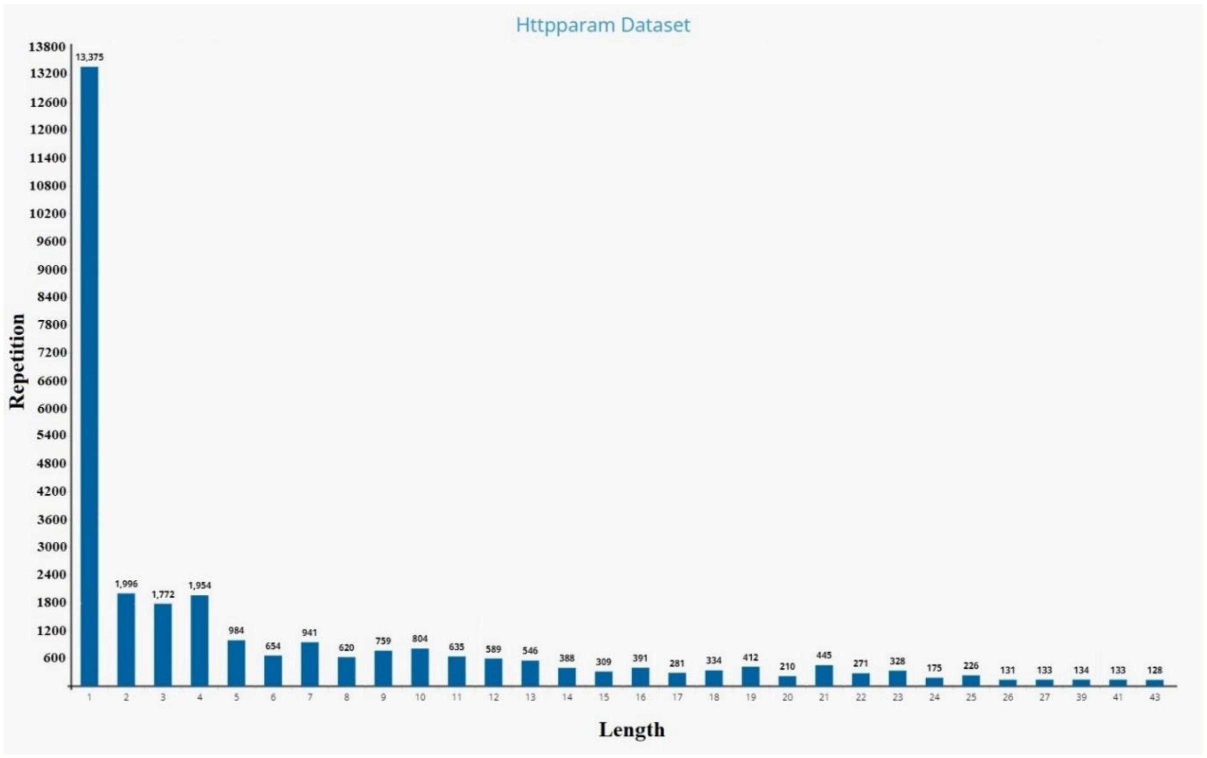
1) Extracted all possible token sequence lengths with their frequencies for every dataset by a Python script, which are developed to make this process much easier.

2) A sample of the most frequented lengths from every dataset is taken which every sample in every dataset is different from other datasets and represented in histogram as shown in **Fig.2 (a)** for PKDD, **Fig.2 (b)** for HTTPPARAM, and **Fig.2 (c)** for CSIC-2012, where the y-axis represents the number of repetitions of the length. However, every time the length increases, the number will be longer than the number max length which means that the length of data will be truncated to fit on the max number means that some of the data will be lost, and every time the length of decreases, the padded token will complete the token with 0s to make the length of a padded token is exactly the number of max length means that the model will learn a lot of 0s instead of fact data.

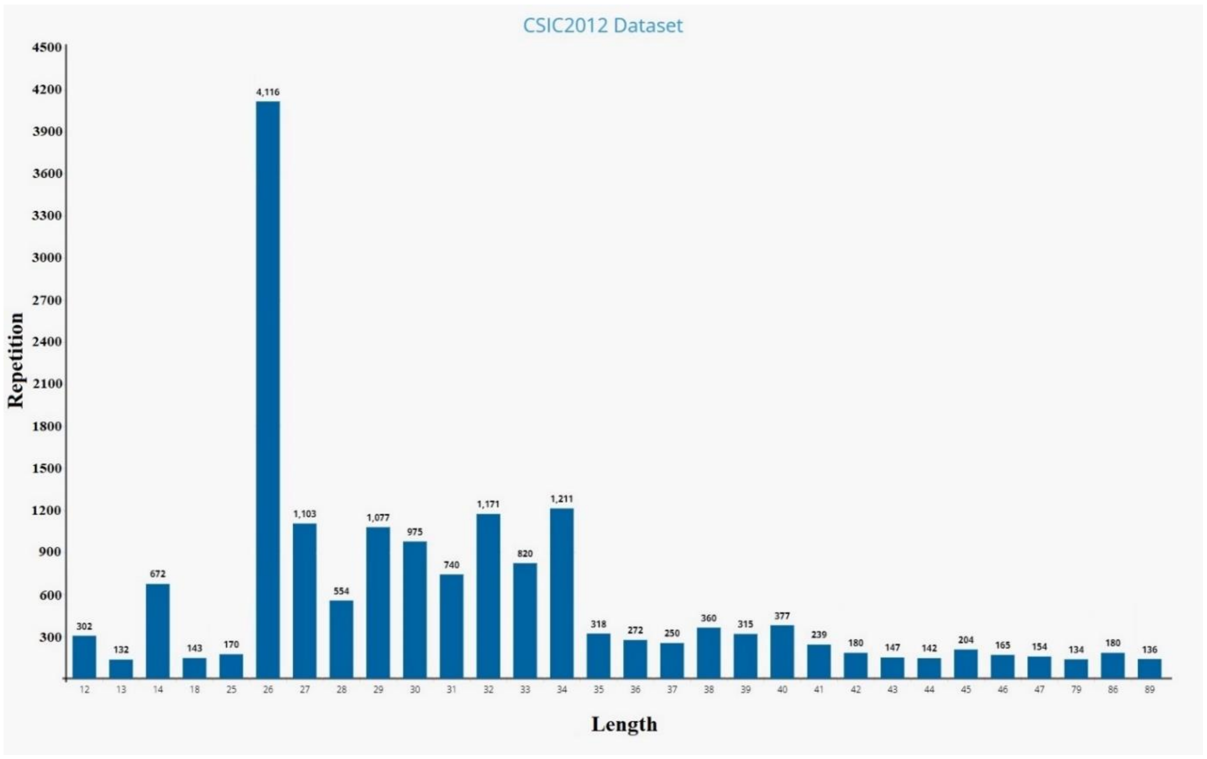
3) The best accuracy is achieved when the average values for the most 30 frequency lengths are used in every dataset, The max length of the ECML-PKDD dataset = 15, in HTTPPARAM the max length = 17, and in CSIC-2012 the max length = 37.



a) ECML-PKDD dataset distribution



b) HTTPPARAM dataset distribution



c) CSIC-2012 dataset distribution

Fig.2. Samples of length token sequence for each dataset

3.2.2. Building Classifier Model Phase

To build a multi-class classifier model for the intrusion detection system the following four deep learning algorithms are used:

1) **Convolutional Neural Networks (CNN):**

The Convolutional Neural Network (CNN) is a type of neural network that consists of several layers. Each layer has a specific function and is represented in **Fig.3**. The CNN architecture includes the following layers [21]:

- **The Embedding Layer**: This layer is the input layer and is used to accept training input. The input data is divided into three sets: training validation, and testing.
- **The Convolution Layer**: This layer learns filters that activate when specific features are detected in certain areas of the input. By swiping on different local areas of the input, different filter sizes can extract richer features.
- **The Max-Pooling Layer**: This layer reduces the dimensionality of each feature map produced by the convolution layer while retaining the most important information.
- **The Dense Layer (Output)**: This layer is used to categorize attacks. In the CSIC-2012 dataset, 9 layers are used from the dense layer.

2) **Recurrent Neural Network (RNN):**

The Recurrent Neural Network (RNN) is another type of neural network that is used to handle short-term dependencies. However, it fails in long-term dependency problems due to gradient vanishing or exploding. the model includes four layers, as shown in **Fig.4** [22]:

- **The Embedding Layer**: This layer is used to accept the input data.
- **The GRU Layer**: This layer handles the problem of gradient vanishing, making the results of the RNN algorithm are close to the LSTM algorithm.
- **The RNN Layer**: This layer allows previous outputs to be used as inputs while having hidden states.
- **The Dense Layer**: This layer is responsible for the classification stage according to the number of classes within each dataset.

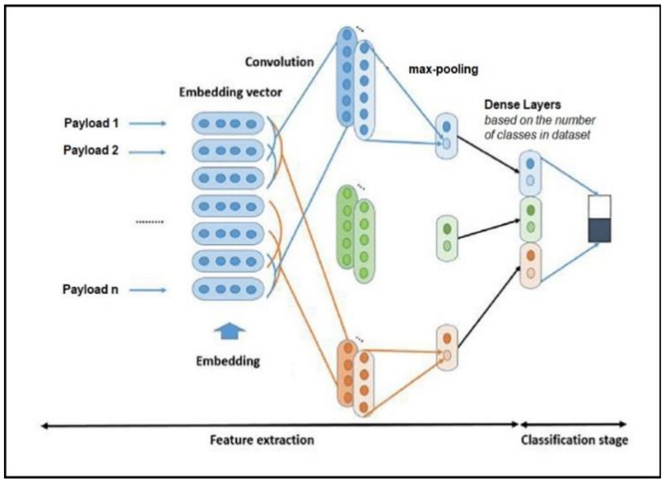


Fig.3. Structure of CNN Model

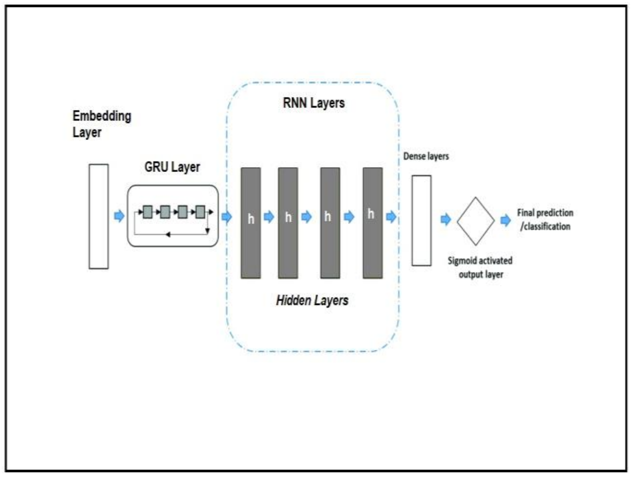

Embedding Layer
GRU Layer
RNN Layers
Hidden Layers
Dense layers
Sigmoid activated output layer
Final prediction /classification

Fig.4. Structure of RNN Model

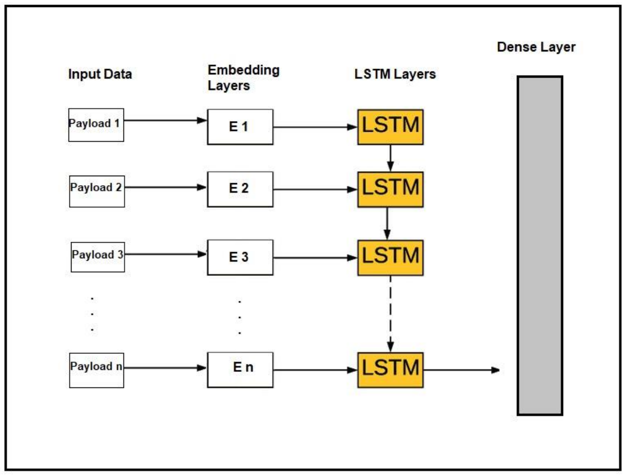
3) Long-Short-Term Memory (LSTM):

LSTM is a variant of Recurrent Neural Networks (RNN) designed to address the problem of vanishing gradients that occurs in traditional RNNs. LSTM networks use memory cells that can retain information over long periods, allowing them to process sequences with long-term dependencies [23]. In our model, the LSTM layer consists of three layers, as shown in **Fig.5**:

- **The Embedding Layer**: This layer converts the input data into a vector representation that can be used by the LSTM layer.
- **The LSTM Layer**: This layer is used in deep learning for solving complex problems such as recognition and text classification. It uses memory cells to retain information over time, making it particularly useful for processing sequences with long-term dependencies.
- **The Dense Layer**: The output of the LSTM layer is converted from a vector representation to an input that can be used for classification.

4) Bidirectional Long-Short Term Memory (Bi-LSTM):

Bidirectional Long Short-Term Memory (Bi-LSTM) [24] is a Long Short-Term Memory (LSTM) variant that is intended to use both past and future context when processing sequential data. Unlike traditional LSTM networks, which process input sequentially from start to finish, Bi-LSTM networks process input in both forward and backward directions at the same time. Layers in our model are shown in **Fig.6**.



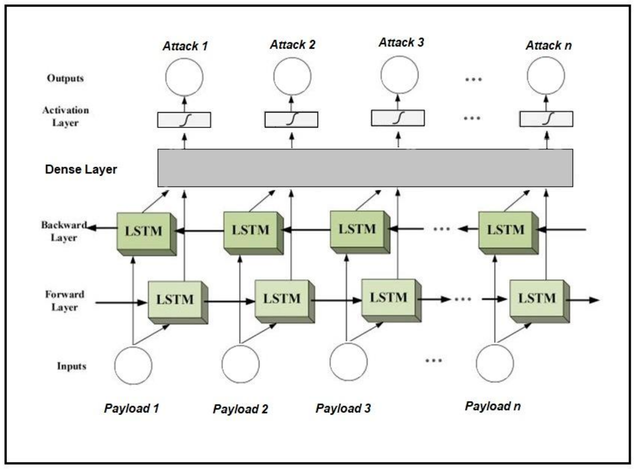


Fig.5. Design structure of LSTM model

Fig.6. Design structure of Bi-LSTM model

The experiment was with different non-linear activation functions, including relu, sigmoid, and tangent hyperbolic (tanh), in all our models. After testing, the sigmoid function performed the best in terms of accuracy. Furthermore, as dealing with a classification problem. the Binary Cross Entropy function is used, which is commonly used for classification tasks, to improve the accuracy of our models.

4. Experimental and Evaluation

4.1. Experimental Tools

A computer with an Intel Core i7-4610M 3.00 GHz processor is utilized, 16 GB RAM, AMD Radeon HD 8790M - 2 GB graphics card, and 512 GB SSD hard disk for implementing the proposed model. The model was developed using Python 3. x environment on a Windows platform with Tensor Flow and Keras. Additionally, data pre-processing was performed using pandas, NumPy, sci-kit Learn, matplotlib libraries, and collections. K-fold evaluation and printing of result values and confusion matrix, statistical analysis, and visualization were carried out using Kfold and StratifiedKFold. Some experiments were conducted using Google Collaboratory to expedite the training process.

4.2 Results And Discussion

To evaluate the performance of different models, they are trained and tested using a 5-fold cross-validation technique. This method involves splitting the dataset into five equal parts, using four parts for training the model and one part for testing. This process is repeated five times, with each part being used once for testing.

Among the models, the Bi-LSTM algorithm achieved the highest accuracy for the ECML-PKDD dataset, with an accuracy of 90.60%. On the other hand, for the CSIC-2012 dataset, the CNN algorithm achieved the highest accuracy of 99.28%. When evaluated on the HTTPPARAM dataset, the Bi-LSTM algorithm performed the best, achieving an accuracy of 99.66%. These results are summarized in **Table 2**.

The proposed multi-class classifier models are evaluated using the most common metrics which are defined as follows:

1) **Accuracy** is calculated as the total of accurate predictions divided by the total number of data sets [25].

$$\text{Accuracy} = \frac{TN+TP}{TN+FN+TP+FP}$$

where TP denotes true Positive, TN denotes True Negative, FP denotes False Positive, and FN denotes False Negative

2) **Precision** is calculated as the number of correct positive predictions (TP), divided by the total number of positive predictions (TP + FP) [25].

$$\text{Precision} = \frac{TP}{TP+FP}$$

3) **Recall** is calculated as the number of accurate positive predictions (TP) divided by the total number of positive (P) [25]

$$\text{Recall} = \frac{TP}{TP+FN}$$

4) **F1-Measure** a measure of the accuracy of the test. It is calculated, based on precision and recall, by the formula [25]:

$$\text{F1-Measure} = \frac{2\ X\ \text{Precision X Recall}}{\text{Precision + Recall}}$$

Overall, the performance of the models varied depending on the dataset and the algorithm used. The Bi-LSTM algorithm showed good performance on two out of three datasets, while the CNN algorithm performed exceptionally well on the CSIC-2012 dataset. The results of the 5-fold cross-validation provide a measure of the model's generalization ability and can be used to compare the performance of different models on the same dataset.

Table 2. Results of 5-fold cross-validation for ECML-PKDD, HTTPPARAM, and CSIC-2012 datasets.

Datasets	Algorithms	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
ECML-PKDD	**Bi LSTM**	**90.73**	**90.56**	**90.47**	**90.60**
	LSTM	89.37	89.07	89.06	89.07
	CNN	90.47	90.56	89.95	90.56
	RNN	88.75	88.42	88.51	88.42
CSIC-2012	Bi LSTM	99.37	99.16	99.21	99.16
	LSTM	97.42	97.96	97.59	97.96
	CNN	**99.18**	**99.28**	**99.22**	**99.28**
	RNN	98.60	98.56	98.49	98.56
HTTPPARAM	**Bi LSTM**	**99.78**	**99.74**	**99.75**	**99.66**
	LSTM	99.44	99.20	99.32	99.20
	CNN	99.73	99.74	99.76	99.42
	RNN	99.10	97.23	97.96	97.23

4.3. COMPARISONS WITH OTHER STUDIES:

Web attack detection using multi-class classification has been the subject of limited research. When the results of our proposed model with the results of previous studies are compared, it became evident that our model outperformed them. Our model achieved an accuracy of 90.60% for the ECML-PKDD dataset and 99.66% for the HTTPARAM dataset, as shown in **Table 3**.

Detecting web attacks using multi-class classification is a challenging task due to the similarity of attack payloads. Therefore, it is essential to develop accurate models that can distinguish between different attack types. Our proposed model showed promising results in this regard and outperformed previous studies. Achieving high accuracy in web attack detection is crucial for ensuring the security of web applications and protecting against cyber threats.

Table 3. Comparison of results with previous research.

Author	ECML-PKDD	HTTPARAM
Mehmet Sevri [17]	87.66	N/A
Hoang [27]	N/A	98.56
Raissi et al [26]	69.00	N/A
Our proposed system	**90.60**	**99.66**

5. CONCLUSION

In this paper, a deep learning-based methodology for detecting web attacks is presented by analyzing HTTP traffic passing through websites. Our approach utilized various deep learning models, including CNN, RNN, LSTM, and Bi-LSTM, and was implemented and evaluated on three different datasets: CSIC-2012, HTTPPARAM, and ECML-PKDD. A data preparation approach is developed that automatically tokenized HTTP data payloads and determined the optimal tokens length to feed deep learning algorithms to improve accuracy. Our models were trained successfully on the different algorithms and were able to identify the type of attack, realizing the principle of multi-classification. Through 5-fold cross-validation, the accuracy rates of 90.60% for ECML-PKDD are achieved, 99.66% for HTTPARAM, and 99.28% for CSIC-2012. Our review of related studies suggests that these classification performances are the best in the field.

In conclusion, our research provides an approach to detecting web attacks using deep learning models, which can significantly improve the security of web applications. Our findings demonstrate the effectiveness of our approach in detecting various types of web attacks with high accuracy rates. Our research can be a valuable contribution to the field of web security and can have practical applications for institutions, companies, and website owners who need to protect their data from potential attacks.

6. REFERENCES

[1]. https://www.internetworldstats.com/stats.htm

[2]. Sophos survey: https://www.sophos.com/en-us/press/press-releases/2022/04/ransomware-hit-66-percent-of-organizations-surveyed-for-sophos-annual-state-of-ransomware-2022

[3]. ENISA: Enisa threat landscape report 2020. https://www.enisa.europa.eu/publications/enisa-threat-landscape-2022

[4]. Kumar, B.S., Ch, T., Raju, R.S.P., Ratnakar, M., Baba, S.D., Sudhakar, N.: Intrusion detection system-types and prevention. Int. J. Comput. Sci. Info. Tech. (IJCSIT) 4(1), 77–82 (2013)

[5]. R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, ‘‘Deep learning approach for intelligent intrusion detection system,’’in IEEE Access, vol. 7, pp. 41525–41550, 2019.

[6]. Y. Zhou and P. Wang, ‘‘An ensemble learning approach for XSS attack detection with domain knowledge and threat intelligence,’’in Comput. Secur., vol. 82, pp. 261–269, May 2019.

[7]. F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, ‘‘A novel two-stage deep learning model for efficient network intrusion detection,’’in IEEE Access, vol. 7, pp. 30373–30385, 2019.

[8]. S. Hao, J. Long, and Y. Yang, ‘‘BL-IDS: Detecting web attacks using bi-LSTM model based on deep learning,’’ in Proc. Int. Conf. Secur. Privacy New Comput. Environ. Cham, Switzerland: Springer, Apr. 2019, pp. 551–563.

[9]. N. M. Thang, ‘‘Improving efficiency of web application firewall to detect code injection attacks with random forest method and analysis attributes HTTP request,’’in Program. Comput. Softw., vol. 46, no. 5, pp. 351–361, Oct. 2020.

[10]. M. Zhang, B. Xu, S. Bai, S. Lu, and Z. Lin, ‘‘A deep learning method to detect web attacks using a specially designed CNN,’’ in Proc. Int. Conf. Neural Inf. Process. Cham, Switzerland: Springer, Nov. 2017, pp. 828–836.

[11]. Liu, C., Yang, J. and Wu, J., 2020. Web intrusion detection system combined with feature analysis and SVM optimization.in EURASIP Journal on Wireless Communications and Networking, 2020, pp.1-9.

[12]. Shaheed, A. and Kurdy, M.H.D., Web Application Firewall Using Machine Learning and Features Engineering. In Security and Communication Networks, 2022. doi: 10.1155/2022/5280158.

[13]. S. Sharma, P. Zavarsky, and S. Butakov, “Machine learning based intrusion detection system for web-based attacks,” in *IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)* 2020, pp. 227-230, doi: 10.1109/BigDataSecurity-HPSC-IDS49724.2020.00048.

[14]. Hussainy, A.S., Khalifa, M.A., Elsayed, A., Hussien, A. and Razek, M.A., 2022, March. “Deep Learning Toward Preventing Web Attacks”. In 2022 5th International Conference on Computing and Informatics (ICCI) IEEE.2022 doi: 10.1109/ICCI54321.2022.9756057

[15]. Yang, W., Zuo, W. and Cui, B., 2019. “Detecting malicious URLs via a keyword-based convolutional gated-recurrent-unit neural network” in *Ieee Access*,vol.7,pp. 29891-29900 ,2019, doi: 0.1109/ACCESS.2019.2895751.

[16]. Tekerek, A., 2021. “A novel architecture for web-based attack detection using convolutional neural network”. In *Computers & Security* 100 .102096, 2021, doi: 10.1016/j.cose.2020.102096.

[17]. Karacan, H. and Sevri, M., 2021. “A Novel Data Augmentation Technique and Deep Learning Model for Web Application Security”. *IEEE Access*,vol.9,pp. 150781-150797 ,2021, doi: 10.1109/ACCESS.2021.3125785.

[18]. CSIC2012. Available online: https://www.tic.itefi.csic.es/torpeda/datasets.html (accessed on 27 February 2022).

[19]. ECML/PKDD Workshop. Available online: http://www.lirmm.fr/pkdd2007-challenge/index.html#dataset (accessed on 28 February 2022).

[20]. https://github.com/Morzeux/HttpParamsDataset

[21]. M. Zhang, B. Xu, S. Bai, S. Lu, and Z. Lin, ‘‘A deep learning method to detect web attacks using a specially designed CNN,’’ in Proc. Int. Conf. Neural Inf. Process. Cham, Switzerland: Springer, Nov,pp. 828–836, 2017. doi: 10.1007/978-3-319-70139-4_84

[22]. S. Hochreiter and J. Schmidhuber, ‘‘Long short-term memory,’’ in Neural Comput., vol. 9, no. 8, pp. 1735–1780 Nov. 1997. doi: 10.1162/neco.1997.9.8.1735.

[23]. M. Schuster and K. K. Paliwal, ‘‘Bidirectional recurrent neural networks,’’in IEEE Trans. Signal Process., vol. 45, no. 11, pp. 2673–2681, Nov. 1997, doi: 10.1109/78.650093.

[24]. Graves and J. Schmidhuber, ‘‘Framewise phoneme classification with bidirectional LSTM and other neural network architectures,’’in Neural Netw., vol. 18, no. 5, pp. 602–610, Jul. 2005, doi: 10.1109/IJCNN.2005.1556215

[25]. Vujovic, Zeljko. (2021).” Classification Model Evaluation Metrics”. in International Journal of Advanced Computer Science and Applications. Volume 12. 599-606. doi: 10.14569/IJACSA.2021.0120670..

[26]. C. Raissi, J. Brissaud, G. Dray, P. Poncelet, M. Roche, and M. Teisseire, ‘‘Web analyzing traffic challenge: Description and results,’’ in Proc. ECML/PKDD, Warsaw, Poland, Sep. 2007, pp. 47–52

[27]. X. D. Hoang, “Detecting common web attacks based on machine learning using web log,” in Proceedings of the International Conference on Engineering Research and Applications, pp. 311–318, Springer, -ai Nguyen, December 2020, doi: 10.1007/978-3-030-64719-3_35.