

## Artificial intelligence-driven method for the discovery and prevention of distributed denial of service attacks

Ashraf ALDabbas<sup>1</sup>, Laith H. Baniata<sup>2</sup>, Bayan A. AlSaaidah<sup>3</sup>, Zaid Mustafa<sup>4</sup>, Muath Alali<sup>5</sup>, Roqia Rateb<sup>6</sup>

<sup>1</sup>Department of Intelligent Systems, Faculty of Artificial Intelligence, Al-Balqa Applied University, As-Salt, Jordan

<sup>2</sup>Department of Autonomous Systems, Faculty of Artificial Intelligence, Al-Balqa Applied University, As-Salt, Jordan

<sup>3</sup>Department of Computer Science, Prince Abdullah bin Ghazi Faculty of Information and Communication Technology, Al-Balqa Applied University, As-Salt, Jordan

<sup>4</sup>Department of Computer Information Systems, Prince Abdullah bin Ghazi Faculty of Information and Communication Technology, Al-Balqa Applied University, As-Salt, Jordan

<sup>5</sup>Faculty of Information Technology, Applied Science Private University, Amman, Jordan

<sup>6</sup>Department of Computer Science, College of Information Technology, Al-Ahliyya Amman University, Amman, Jordan

### Article Info

#### Article history:

Received Feb 7, 2024

Revised Aug 6, 2024

Accepted Aug 30, 2024

#### Keywords:

Botnet

Deep learning

Distributed denial of service attacks

Distributed denial of service detection

long short-term memory

MaxPooling

### ABSTRACT

Distributed denial of service (DDoS) attacks has emerged as a prominent cyber threat in contemporary times. By impeding the machine's capacity to give services to legitimate clients, the impacted system performance and buffer size are reduced. Researchers are working to build sophisticated algorithms that can identify and thwart DDoS violations. An effective approach for DDoS attacks has been proposed in this work. This research presents a model as a potential explanation for DDoS assaults. In order to successfully identify this kind of attacks, which may stop or block the urgent and vital transmission of data, we present a distinctive method that integrates a pair of fully connected layers within an amalgamated deep learning (DL) framework with long short-term memory (LSTM) and a max pooling layer. The acquired accuracy reached 99.58%.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



### Corresponding Author:

Ashraf ALDabbas

Department of Intelligent Systems, Faculty of Artificial Intelligence, Al-Balqa Applied University

Salt 19117, Jordan

Email: ashraf.dabbas@bau.edu.jo

## 1. INTRODUCTION

The contemporary threat landscape for online services and websites is demonstrably expanding due to the proliferation of distributed denial of service (DDoS) attacks [1]. System uptime, a core tenet of information security, represents the paramount requirement for any network [2]. DDoS attack is an assault on a victim by means of various, dispersed resources with the intent to impede authorized users from gaining access to the resources of the target entity [3]–[6]. System resources, network connectivity, and other resources may all be attacked (for a visual representation of this, see Figure 1). It creates a significant bottleneck in terms of capacity and connection, which ultimately results in the interruption of all functions provided by the network [7]. Cloud ecosystems incur highest damages owing to total service disruptions and performance degradation [8]. The primary goal of DDoS attacks is to disrupt the accessibility of services for legitimate users. The malicious flooding causes the network to become overloaded, going beyond the bandwidth capacity of the network, and causing disruption to the services [9]. Because of the striking similarities between the attack traffic and the legitimate traffic [10], it might be challenging to identify the attack traffic during a DDoS assault. They operate in a manner that is quite similar to that of regular network packets, despite the fact that they are sent in greater amounts and concentrations toward the victim [11]. This occurs more often in the first phases

of an assault, in particular during poor and low-traffic attacks [10]. Note: The severity of the assault is often determined by volumetric characteristics such as the frequency of packets, connections, and bits per unit of time [11]. Identifying and defending against an illegal assault that emanates from a restricted quantity of devices is comparatively easier. The DDoS attack employs a significantly large quantity of devices and the actions exhibited by these machines in concert almost eliminates any possibility of satisfying legitimate user inquiries [12]. The hacked devices send a high amount of packets over the network without pausing for any pauses, which tricks the victim into thinking the traffic is real and allows them to access their data. As a direct consequence of this, not only does the host connect with a variety of devices, but it also does so with a wide variety of packet types [13]. Research has demonstrated that a DDoS attack can be understood as a competition for resources between the protection and the hackers. Specifically, the likelihood of achieving success in such an attack is positively correlated with the abundance of available resources [14].

Attacks on a DDoS may be broken down into three categories: brute force, spoofing, and flooding. The most prevalent and damaging of the three types of assaults is the flooding attack, which clogs up the available bandwidth on the network and prevents any genuine requests from being processed. The strategies for survival concentrate on a specific group of victims and require individuals to be aware of and respond appropriately to attacks on their own. On the other hand, a network-wide flood necessitates the implementation of mitigation strategies before it can reach the victims, giving it an ideal vector for multi-targeted assaults. The installation of software updates and the use of appliances are not sufficient means of blocking or preventing DDoS attacks entirely. Accordingly, internet service suppliers either use cleaning solutions or overprovision their networks in order to combat the problem. Both approaches cannot be implemented due to budgetary constraints [15]. The DDoS framework is comprised mostly of zombies that are modelled after their controllers and relayed via the internet. Generally speaking, any and all communications that take place among an individual responsible for controlling and directing a particular action or process, commonly referred to as the handler, and the act of initiating an aggressive action or assault, commonly known as the attack are encoded, so making the assault undetectable. Because attackers are geographically dispersed and may fake their media access control (MAC) and IP addresses, identification can be a time-consuming and difficult process. DDoS assaults have swiftly progressed and grown to be quite sophisticated throughout the course of time. Attacks using DDoS have a devastating influence on the computational, financial, and infrastructural resources of a company [16]. Even large cloud operators like Azure of Microsoft and EC2 from Amazon have been hit by the growing flood of DDoS assaults [10].

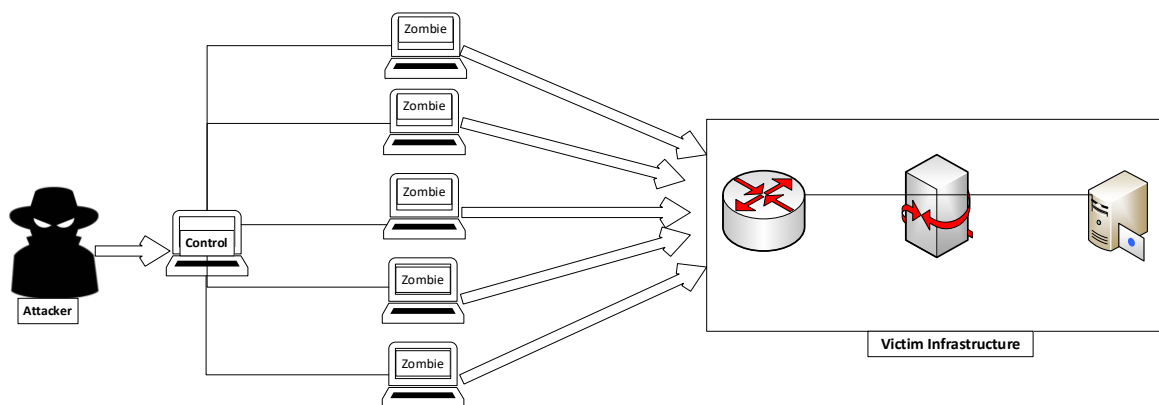


Figure 1. Attack in the form of a DDoS

## 2. BACKGROUND

The goal of a DDoS is to prohibit authorized users from accessing the resources they need by disrupting the availability of network resources. The method of attack largely focuses on gaining dispersed access to devices in order to exploit previously discovered flaws [17]. Attacks are aimed against different levels of the infrastructure that makes up the network, such as the transport layer and the application layer [18], [19]. DDoS attacks are categorized into the following categories according to the architecture of the network [20].

- a) Attacks at the application layer are a kind of layer seven network layout that aims to cause a denial of service by overwhelming the system's resources [21]. An attacker uses a known weakness in a program or operating system to disrupt the network's functionality. As these attacks need very modest rates of traffic to carry out, they are commonly misunderstood as implementation mistakes. An HTTP flood is a

kind of assault that denies access to a service in which numerous machines make simultaneous requests for access to a resource, overwhelming that resource.

- b) The second kind of DDoS attack is a resource depletion pounce, which focuses on security issues in the transport layer and network layer. These assaults, also known as state exhaustion attacks, are designed to use up all of a system's available resources, including its CPU time and main and secondary storage. This attack constitutes a mix between targeted messages and sheer volume since it takes the use of protocol flaws. When an attacker uses faked source IP addresses in a transmission control protocol (TCP) synchronize (SYN) flood, the victim receives SYN messages but is never confirmed with a response. As the victim replies to each handshake but never gets confirmation from the attacker, its resources will eventually be depleted [19].
- c) Volumetric attacks: The attacker sends massive volumes of data to the prey through botnets or other techniques of expansion, which causes the bandwidth that is available across the victim and the wider network or internet to be at risk of becoming depleted. The user datagram protocol (UDP) protocol is often used in order to take advantage of any excessive rise in the size of a packet. The amplification of domain name system (DNS) attacks involves the transmission of service demands with the intention of altering the source address space to include the address of the targeted victim. The aforementioned phenomenon leads to an increase in the magnitude of responses from the servers, ultimately resulting in the depletion of the victim's available bandwidth. The most visible sign of a DDoS attack is a website or service that suddenly becomes unusable or very sluggish. However, due to the fact that a variety of factors, such as a normal increase in traffic, are capable of causing the same performance concerns, more study is often necessary. Tools for traffic analytics may assist in spotting some of the following tell-tale symptoms of a DDoS attack [22]–[25].
  - Large quantities of potentially malicious communications coming from a specific IP address or group of IP addresses.
  - A deluge of traffic is generated by users who all share the same behavioural pattern, such as the sort of device they are using, or the version of their web browser.
  - An inexplicable rise in requests to a particular page or destination.
  - Strange patterns of traffic, such as peaks that occur at unusual times of the day or patterns that don't seem to fit in with nature (e.g. a boost every 15 minutes). There are further, more particular indicators of a DDoS assault, and these indicators might change based on the sort of attack that is being carried out.

### 3. RELATED WORK

In this section, many DDoS detection studies have employed machine learning (ML), mostly deep learning (DL). Here, most of these investigations are carefully depicted. DL systems identify and classify DDoS assaults using ML. Labelling attack sample data using a well-trained network. The biggest benefit of DL is accurate detection, which several suggested algorithms proved. Sometimes it fails to detect assaults. Wani *et al.* [3] introduced advanced ML methods for classification, such as support vector machine (SVM), random forest (RF), and naïve Bayes. Data was generated by an automated intrusion detection system employing Tor Hammer for hacker tagging. The three algorithms had 99.7%, 97.6%, and 98.0% accuracy. This investigation showed that SVM detects best among other algorithms. Multiple ML techniques have been developed for DDoS categorization utilizing various datasets and characteristics. Most research focuses on artificial neural network (ANN) [26]–[31] and compares SVM, RF, decision trees (CART), K-nearest neighbours (KNN), and others to identify and classify attacks as innocuous or DDoS. [32]–[36].

Amma and Subramanian [37] trained their models on pre-processed DoS/DDoS samples on a simulated ANTS2019 dataset and assessed accuracy. Second, the authors' rigorously managed synthetic dataset seamlessly integrated with the famous CICIDS2017 dataset. Watch their creative technique blend these two extraordinary data sets, opening up new opportunities for groundbreaking study and analysis. The junction of these outstanding datasets will provide unprecedented insights. We train the suggested classifiers again and test their detection performance on these two datasets. These models improved in the second phase of the experiment, with long short-term memory (LSTM) and deep neural network (DNN) accuracy of 95.15% and 96.17%, respectively. LSTM and DNN have 0.987 and 0.989 area under the curve (AUCs). In the same year, VCDeepFL was launched for DoS detection. VCDeepFL uses several of techniques, including fully connected neural network (FCNN) and vector connected neural network (VCNN). Unsupervised preliminary VCNN learning and supervised FCNN multiclass classifier training comprise the training stage [38]. DDoS assaults may be identified using a diode array detection - multichannel convolutional neural network (DAD-MCNN), or multichannel CNN. The authors divide characteristics into traffic, host, and packet components to determine channel count. Researchers trained the MC-CNN gradually. The datasets CICIDS2017 and KDDCUP99 were used to test many models. The models were tested for binary classification across both datasets and many

KDDCUP99 classes [39]. DL model with feature extraction and classification framework was presented in [40]. An input layer with 69 units, three hidden layers with 50 units each, and a two-unit output layer make up the DNN architecture. The authors divided CICDDoS2019 into Dataset-1 and Dataset-2. Dataset-1 includes regular and hostile traffic. To classify DDoS hacks, Dataset2 was created. DNN modelling has achieved approximately 100% accuracy for Dataset1 DDoS attack identification in the early phases of intervention, suitable for real-time scenarios. It also classifies DDoS assaults with 95% accuracy using Dataset2. A deep CNN architecture was developed for software defined networks to identify DDoS attacks. The proposed ensemble technique and networks were analysed using CICIDS2017. The latest technique uses DL and hybrid methods. CNN's crew outperformed the other three DL-approaches but testing and training take time. This ensemble CNN is 99.45% accurate [41]. DL, especially CNN, has expanded DDoS detection [40]. CNN is the most popular method, however other research has used SVM and decision trees [42]–[44]. In Table 1, a detailed comparison with prior works is illustrated to highlight the advantages and improvements of this proposed method.

Table 1. A detailed comparison with prior works

Model name	Year	Accuracy (%)	Advantages/improvements	Framework/models used	Methodology used
Proposed model	2024	99.58	Highest accuracy among compared models, robust against varied DDoS attack patterns, integrates LSTM and MaxPooling	LSTM, MaxPooling	DL, LSTM
Stacked autoencoder [45]	2022	98.99	Good accuracy, utilizes unsupervised learning for feature extraction	Autoencoder	DL, Autoencoder
Firefly classification algorithm [45]	2018	91	Novel application of Firefly Algorithm for classification, moderate accuracy	Firefly algorithm	Machine learning
Naive Bayes and RF [46]	2019	90.90	Traditional models, lower accuracy compared to DL approaches	Naive Bayes, RF	Machine learning
Ensemble CNN [41]	2022	99.45	High accuracy using hybrid methods, time-consuming training and testing processes	Convolutional neural network (CNN)	DL, CNN
LSTM and DNN [38]	2019	96.17	Integration of unsupervised VCNN and supervised FCNN, moderate to high accuracy	VCNN, FCNN	DL, VCNN, FCNN
Deep CNN [40]	2020	100 (Dataset1), 95 (Dataset2)	Early phase DDoS attack identification with near-perfect accuracy on initial dataset, effective but slightly lower accuracy on second dataset	Deep CNN	DL, CNN
Cuckoo search algorithm-trained RBF [47]	2022	96.9	Effective feature selection using Cuckoo Search, stable accuracy	Radial basis function, cuckoo search	Machine learning, feature selection

a) Advantages of the proposed model:

- Highest accuracy: The proposed model achieves an accuracy of 99.58%, outperforming other models in the comparison.
- Integration of LSTM and MaxPooling: The unique combination of LSTM and MaxPooling layers provides robustness against varied DDoS attack patterns.
- Effective handling of temporal sequences: LSTM's ability to detect both long-term correlations and temporal sequences enhances the detection of even minor DDoS attacks.
- Feature engineering: The model effectively selects and utilizes the best 40 features out of 77 to achieve the highest accuracy, ensuring efficient and accurate prediction.
- Adaptability: LSTM's adaptive learning capabilities make the model suitable for dynamic environments, maintaining high performance under varying network conditions.

b) Comparison highlights:

- The stacked autoencoder model achieves high accuracy but lacks the robustness and adaptability of the proposed model.
- The cuckoo search algorithm-trained RBF provides strong feature selection but falls short in overall accuracy compared to the proposed model.
- The ensemble CNN model offers high accuracy but requires significantly more time for training and testing, making it less efficient for real-time applications.
- Traditional models like naive Bayes and RF show lower accuracy, highlighting the superiority of DL approaches for DDoS detection.

- The Deep CNN model achieves near-perfect accuracy on an initial dataset but shows slightly lower performance on a more complex second dataset.

The proposed model's integration of LSTM and MaxPooling, combined with effective feature engineering and adaptability, establishes it as a superior approach for DDoS detection in dynamic and varied network environments.

#### 4. DISCUSSION

After a research and detailed investigation of ML's potential DDoS defense applications, additional upgrades are needed to establish a solid ML-based mitigation system. Many research used a synthetic dataset, which affected accuracy since ML approaches build a pattern-detection profile from the input dataset. Many studies have tried to stop particular DDoS attacks, but attackers may always learn new tactics and work harder to succeed. Many experiments used software to simulate an attacker and user behaviour, but a real-world DDoS attack uses a swarm of infected PCs, or "zombies," to inflict their damage. This technique should evaluate the system's effectiveness and durability in real life.

##### 4.1. Dataset

This data is gained from the system's application layer. The labels are classified into three groups based on the network analysis: i) Benign: Legitimate, ii) DoS slowloris: DoS assault, and iii) DDoS by DoS Hulk. Comprised of harmless and the newest examples of typical DDoS assaults, which are analogous to data taken from the actual world. Additionally, included are the findings of the investigation of the flow on the network. Contains flows that have been annotated according to the time stamp, origin IP address, destination IP address, source port, destination port, protocols for communication, and vulnerability signatures (CSV files). The dataset can be found in: <https://www.kaggle.com/code/hamzasamiullah/ml-analysis-application-layer-dos-attack-dataset/data>. The dataset is structured to help researchers and practitioners develop and test ML models for intrusion detection systems (IDS).

##### 4.2. Feature engineering

ML uses mathematical models to acquire insights or forecast the future. The characteristics feed these models. Features are numerical representations of raw data components. Features sit between data and models in the ML process. Features are numerical representations of raw data. Raw data may be turned into quantitative measurements in many ways, which is why features may resemble many things. Also important is the number of qualities. Insufficient features and information will prevent the model from performing its primary job. The model will be harder and more expensive to train if there are too many or useless attributes. Anything that goes wrong during model training might influence its performance. Using feature selection processes to remove unhelpful attributes might simplify the final model. The final model should be simple, easy to compute, and lose minimal forecast accuracy. Figure 2 shows the data's journey through assessment and deployment.

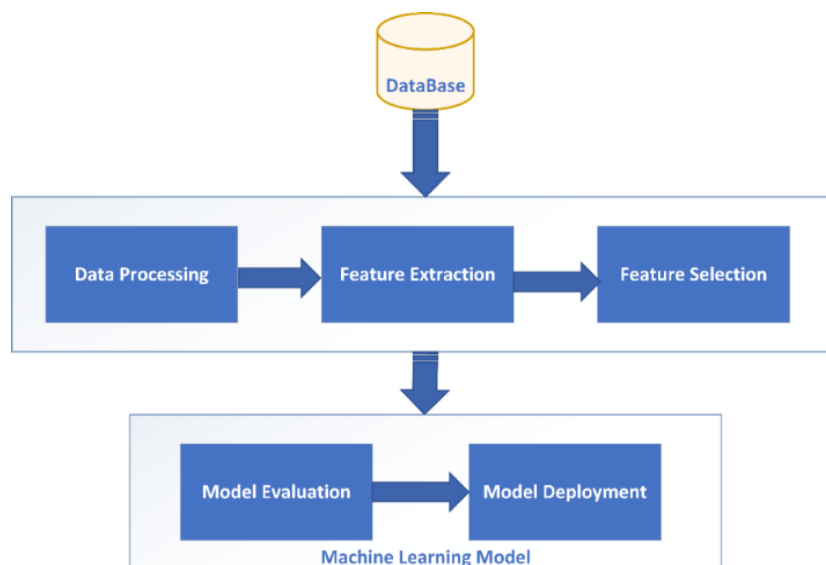


Figure 2. Representation of modelling including feature engineering

To find a model, certain feature selection approaches train numerous candidate models [48]. Feature engineering involves choosing, altering, and converting raw data into characteristics for supervised learning. Also called feature extraction [49]. Designing and training better features may be needed to make ML successful for new issues. A "feature" is any measurable input for a predictive model [50]. Colour and tone are examples of features. In its most basic form, feature engineering refers to the process of transforming raw observations into the necessary features via utilizing artificial intelligence techniques. The ML mechanism relies heavily on feature engineering. These mechanisms exploit these synthetic qualities to boost efficiency and provide better outcomes. Since data is where a data scientist's focus lies, model accuracy is of paramount importance. Feature selection tackles any probable obstacles by automatically picking a subset of data that is most relevant to the issue at hand. Our model was able to select the best 40 features among 77, in order to acquire the highest accuracy.

#### 4.3. Max pooling and pooling process

Pooling uses a filter or kernel like convolution. Pooling leaves the filter/kernel blank, like a blank array. The approach involves applying a filter over successive patches to process kernel pixels like convolution. Max pooling entails filter-scanning an item. Each cycle, the filter's highest pixel value is chosen as a new pixel. Take the highest pixel value from each occurrence and apply it to a new pixel to create a new portrayal. The resulting image is called a maximum pooling representation of the original data. A max-pooling operation [51] downsamples convolutional result bands, reducing unpredictability. The process of reducing the sample size in a 1-D max pooling layer involves splitting the input into 1-D pooling areas and then determining the maximum value within each zone. The dimension across which the layer pools is contingent upon the input of the layer:

- The layer performs pooling across the "T" (time) dimension for input data having three dimensions, namely "B" (batch), "C" (channel), and "T" (time), which may be either time series or vector sequences.
- In the case of one-dimensional picture input, where the data is represented by three dimensions denoted as "C" (channel), "S" (spatial), and "B" (batch), the layer performs pooling operations across the "S" (spatial) dimension.
- The layer conducts pooling in the dimension for a sequence of 1 image. These images are represented by data, with four dimensions; "B" (batch) "C" (channel) "S" (spatial) and "T" (time).

Pooling is a process that consolidates results into a range of values. We still use strides, padding and convolutional semantics as before. Pooling operations are applied independently to each channel. Do not alter the number of channels. Max pooling is preferred over pooling because it ensures output invariance. When downsampling or pooling feature maps we emphasize the feature in the patch rather than averaging occurrences as done in average pooling. This approach performs better than pooling in applications especially in computer vision tasks, like image classification.

#### 4.4. Long short-term memory

DL heavily relies on LSTM recurrent neural networks (RNNs). It excels in detecting long term connections, which makes it ideal for applications involving sequence prediction. Unlike regular neural networks, LSTM models utilize feedback connections to analyze sequential information rather than individual data points. This gives them a strong capability in understanding and predicting patterns in sequential data such as time series, linguistic analysis, and vocal syntax. Over time, the LSTM model has evolved into a powerful tool for artificial intelligence and DL. Its ability to analyze complex communication patterns and identify anomalies has greatly benefited various businesses, particularly in DDoS detection. By extracting packet headers, traffic rates, protocols and transmission patterns from past data, the ML model can differentiate between legitimate and malicious network behaviors after being trained with both instances. From the experimental results, we found that LSTM effectively detects both sudden and persistent and malicious network behaviors after it is trained with both types of instances. These results are attributable to the LSTMs ability to detect both long-term data correlations and temporal sequences of anomalous network traffic data. Thus, the LSTMs are capable of detecting even minor DDoS attacks by monitoring the network traffic. The main advantage of LSTMs over the existing techniques studied in this paper is that the LSTMs are designed for DDoS defense because LSTMs learn and adapt to dynamic environments. In summary, LSTM can be used to analyze the regularities in network traffic, detect abnormalities in this traffic, provide real-time notifications, and improve DDoS prevention and mitigation.

##### 4.4.1. Long short-term memory construction

In the previous section of LSTM, it is shown how it can solve the problem of vanishing gradients, from which RNN suffers. This section will look at how exactly this deficiency in RNN architecture is remedied using LSTM. In some ways, LSTMs are simply like an RNN cell. Time steps refer to chunks of time that are

single. These parts are called “gates.” They decide what information enters and leaves an LSTM cell. Those three gates have been named “output,” “forget,” and “input.” The LSTM unit consists of a group of neurons in a normal feedforward neural network. It has three gates and an LSTM cell in this unit. Each cell has two other layers beneath it: a hidden layer and a state layer at the moment in this layer. The RNNs also contain one hidden state and one  $C_{(t)}$  at the current layer, whereas in the case of LSTMs, they consist of  $H_t$  right now, which was  $H_{(t-1)}$ , reflecting its previous hidden stage from the last time step. The model of LSTM includes both states  $C_{(t-1)}$  and  $C_{(t)}$  for long-term and short-term memory, respectively, reflecting historical timestamps until now. Long-term memory (LTM) refers to cellular memory, while short term memory (STM) refers to hidden memory that involves information on where particular things happen but not when or how often they do so on average. Note that cellular states store timestamps along with information therein as well. The first step involves determining the data from the previous time step that will be removed and retained in building the LSTM neural network structure. The forget gate is given by the following equation.

Forget gate:

$$f_t = \sigma(X_t * U_f + H_{t-1} * W_f)$$

We should strive to fully understand the equation that is being discussed.

- As of the current timestamp, the input value is denoted by the variable  $X_t$ .
- $U_f$  represents the weight associated within the input.
- Hypothetical state  $H_{t-1}$ :: The concealed state from the time period before it.
- $W_f$ : This variable denotes the weight matrix associated with the concealed state.

It is then subjected to a sigmoid function. As a consequence,  $f_t$  will be limited to the interval between 0 and 1. The cell state of the timestamp before is then multiplied by this  $f_t$ .

$$C_{t-1} * f_t = 0 \dots \text{if } f_t = 0 \text{ (Disregard all)}$$

$$C_{t-1} * f_t = C_{t-1} \dots \text{if } f_t = 1 \text{ (Neglect none)}$$

Setting a weight for the importance of the recently learned information that the input conveys is the job of the input gate. The formula representing the input gate may be found as follows.

Input gate:

$$i_t = \sigma(X_t * U_i + H_{t-1} * W_i)$$

- Let us endeavour to comprehend the equation at hand.
- $X_t$ : Enter data starting at timestamp  $t$ .
- $U_i$ : The weight matrix associated with the input.
- $H_{t-1}$ : A previously concealed state at the timestamp
- $W_i$ : The weight matrix, denoted as  $W_i$ , indicates the relationship among the input and the concealed state of a given system.

The sigmoid function has been employed once again. Consequently, the value of variable  $i$  at time  $t$  will fall inside the range of 0 and 1.

New information:

$$N_t = \tanh(X_t * U_c + H_{t-1} * W_c) \text{ (Fresh Details)}$$

The incorporation of additional information into the state of the cell depends on the concealed condition at the previous timestamp ( $t - 1$ ) and the input ( $x$ ) at the current timestamp ( $t$ ). The used activation function in this context is the hyperbolic tangent  $\tanh$  function. The  $\tanh$  function restricts the range of new information values to be within the interval of -1 to 1. In the event that the value of  $N_t$  is negative, the data is deducted from the cell state. Conversely, if the value is positive, the data is incorporated into the cell state at the present timestamp. Nevertheless, the  $N_t$  will not be instantly incorporated into the cell state. The revised equation is now shown.

$$C_t = f_t * C_{t-1} + i_t * N_t \text{ (updating cell state)}$$

In this instance,  $C_{t-1}$  represents the current state of the cell as of the current time, while the other values are those that we have computed in the past.

Output gate: The output gate's equation, which is somewhat similar to the equations for the two earlier gates, is shown following.

$$o_t = \sigma(X_t * U_o + H_{t-1} * W_o)$$

The sigmoid function ensures that the value of the variable will be bounded between the range of 0 and 1. To compute the present concealed state, we shall use the product of  $o_t$  and the hyperbolic tangent of the modified cell state. As shown in the following illustration.

$$H_t = o_t * \tanh(C_t)$$

The hidden state is determined by the combination of the LTM  $C_t$  plus the present output. To get the present timestamp's output, one may simply use the *Softmax* activation function upon the hidden state  $H_t$ .

$$\text{Output} = \text{Softmax}(H_t)$$

The prediction is determined by identifying the element with the highest score in the output. Figure 3 presents a more understandable representation of the LSTM network.

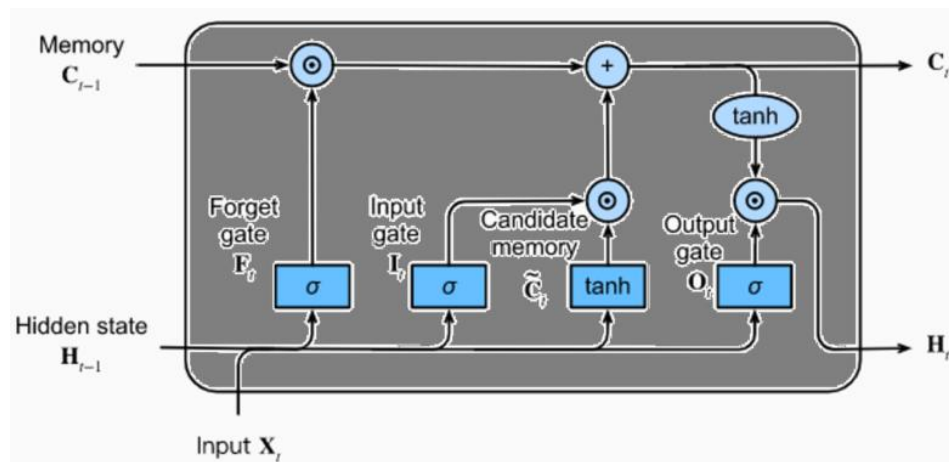


Figure 3. LSTM components

#### 4.5. Fully connected dense layer (ReLU activation function)

To explain, the previous layer's values are multiplied by their weights and summed. Results are processed using an activation function. This scenario uses the rectified linear unit (ReLU) activation function. The mathematical formula for ReLU activation function is:  $F(x)$  represents the greatest value between 0 and  $x$ , where  $x$  is the neuron input. If positive, the function returns the value; else, zero. This function is extensively used in DL because it adds non-linearity, helps the network learn complicated patterns, and mitigates the vanishing gradient problem. The output of each neuron is calculated by a fully connected dense layer through adding the activation function into inputs' weighted sum. The next layer in the neural network takes as its input, the output from an earlier one. DDoS attacks can be detected and mitigated by an ML model with a fully connected dense layer that uses the ReLU activation function. Many sources, for example, traffic and network records provide input data for the fully connected layer. Full connection between all neurons in current and previous layers characterizes this dense completely linked layer and has direct information transmittance capability. These layers are usually using ReLU activation function because they are able to generate non-linearities and decrypt sophisticated data patterns. The full connected dense layer gets trained on incoming data to help it identify patterns and relationships using rule extraction. The accuracy of distinguishing between DDoS activities and normal network traffic is high with this model. For computer networks to hold steady and safe, DDoS mitigation is vital. After training, fully linked dense layers can distinguish between DDoS-related events happening on a network or regular internet activity. Dense-layer outputs are thresholded by fixing thresholds for their outputs such that only signals above these thresholds pass through them. This technique helps to find out abnormal traffic features which may call for investigation or arrest completely unusual activities on the networks being monitored by software agents or other monitoring mechanisms inside these



systems. The concept of full connection thick layer can be adopted in real-time. Fully linked dense layers are commonly part of DNNs or CNNs. In order to improve detection, these models may include multiple layers, including entirely connected dense layers and other layers. For accurate and consistent detection, the model's training data must cover a variety of regular and DDoS assault situations.

#### 4.6. Complete layers of the proposed model architecture

In this part, we will discuss the numerous architectural approaches that were included in our model. Figure 4 depicts the suggested model's network layer design. The 'architecture' of a DL model may be thought of as the 'layers' that make up the model. When it comes to the models, there are many different kinds of layers that may be employed. Each layer has unique properties that make it essential. A fully connected layer processes input. To build a neural network, traffic flow characteristics are recorded initially. Each fully linked layer in this network changes the underlying problem's feature space. Researchers found that the rectified linear activation ReLU outperforms the softmax unit. The vanishing gradient problem in deep networks may explain this empirical observation. Softmax has zero slope for most input values. Thus, deeper networks typically have zero gradients. Nonzero gradients propagate via the ReLU function because the slope is nonzero over a wider input space. Fully linked networks can remember all training data given enough time, which is astonishing. Therefore, "convergence" training on a fully connected network is not a legitimate statistic. Network training and instruction will continue as long as the user can wait.

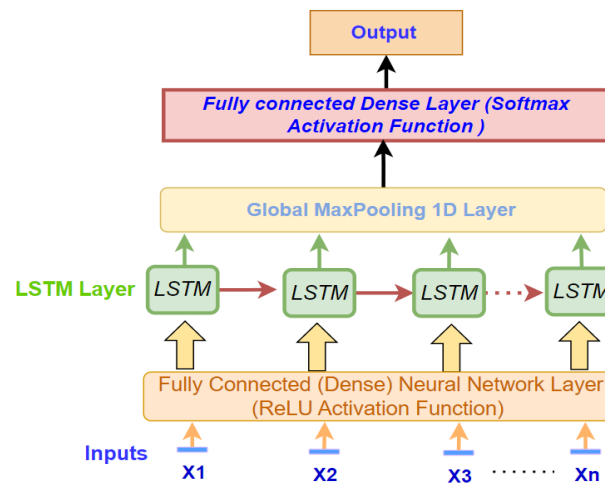


Figure 4. The architecture of the proposed model

Actually, fully linked networks may find and exploit erroneous correlations in their data. Controlling networks and preventing them from responding inappropriately is crucial to modelling success. The extracted features include packet length, transport layer protocol, data delivered and received, time between packet arrivals, and flow duration. There are many ways to train a completely linked layer. First, we trained our models on a large dataset. When dealing with large datasets, which may not fit in memory, gradients cannot be calculated at each step. Instead, practitioners compute the gradient on a tiny piece of the data (typically 50–500 data points). Minibatch is the conventional term for this little data set. The next layer has a more complicated network structure. A form of RNN called LSTM. RNNs concentrate on time series learning. This works well for us since our database contains many interrelated properties. RNNs, which enable data to be fed back into earlier layers of the network, are often used for time-series data analysis because they provide more precise findings. LSTM units are included in one extensively used form [52]. These networks keep track of the current state of their data in the cells of their nodes, and that information is updated in response to new inputs and past outputs. The output is determined using the present situation of the cells and the input data. The cell states and gate layers that makeup LSTM networks are combined into a single entity. Each gate is representative of a completely linked layer and is equipped with an appropriate activation function. This function accepts as input a concatenation of the data from the most recent time step as well as the output from the gate that came before it. After that, a whole LSTM layer is constructed by performing element-wise multiplication and addition on those gates in order to merge them. Whereas the input and cell gates both lead to an increase in

value, the forget gate might actually cause a drop in the values stored in the cell. In this instance, the output gate decides the output and classification values. The output gate calculates output.

Mini-batching seems to improve convergence by allowing more gradient descent steps with the same processing capacity. Minibatch size is sometimes chosen empirically through hyper-parameter fiddling. The learning rate determines how much importance each gradient descent step receives. Finding the best study pace might be difficult. Many inexperienced deep learners set their models' learning rates incorrectly, which causes them to be surprised when their models fail to learn or return a number. Adjusting the learning rate may assist if models aren't learning anything new. This issue is made much better by ADAM's ability to make learning rate selection simpler. The dense layer's input shape goes via the input layer before it, so we can't change its attributes after calling it.

Softmax is a popular nonlinear activation function for neural networks. It simulates probability with output values between 0 and 1. This contributes to its popularity. Thus, it converts the real-valued output of a linear layer to a probability, which may be used as a probability output. It is an essential component of logistic regression, which may be used for binary classification. DDoS attack classification studies were done using the suggested LSTM model. The model under consideration was trained using a dataset consisting of 809,361 samples and subsequently evaluated using a separate dataset containing 346,869 samples. Training the LSTM scheme for identification was trained using the Keras framework in conjunction with Python. The classification task employed the ADAM optimization technique with a learning rate of 0.01 and a momentum of 0.0. The model's batch size was set to 6. The initial model consisted of 331,525 parameters. The utilized model consisted of 256 LSTM units and recurrent dropout set to 0.23.

## 5. RESULTS

The data that is being entered, which is our feature vector, goes into a layer that is completely linked. In order to achieve a certain degree of non-linearity while maintaining training stability, rectified linear functions are employed as activation functions. After the activation layer comes the dropout layer, which is used for regularization of the network. There are many occurrences of this particular pattern, which consists of completely linked, activation (Softmax), and dropout layers. Regularization using dropout removes a particular proportion of nodes from a completely connected layer. A deleted node has no influence on the activation function. Without activation, gradients at deleted nodes converge to zero. The infrastructure will be limited from "co-adaptation." We shall briefly explain co-adaptation in non-regularized deep networks in this study. Consider a single deep network neuron that has learned something. When this occurs, other neurons in the network will rapidly learn to trust the neuron's information. This method will make the network unstable since it will depend disproportionately on the neuron's features, which may be an aberration in the dataset, rather than learning a general rule. This will impede network accuracy growth.

Dropout prevents co-adaptation because neuronal death during training is unpredictable, making even a single powerful neuron unreliable. Because of this, more neurons must "fill in the gaps" and acquire appropriate representations. This approach theoretically enhances learned models. Two empirical effects of dropout exist. Even for huge deep networks, dropout slows training loss approaching 0. This prevents the network from overlearning by remembering training data. Finally, dropout somewhat improves the model's incoming data prediction. Dropout is generally considered more than a statistical hack due to its vast application. Turning off dropout is recommended before making any forecasts. Predictions may be substantially less accurate and valuable if dropout isn't disabled. Early stopping is difficult to accomplish in practice. In Figure 5 shows how loss curves for deep nets may fluctuate widely during regular training.

When given enough time, fully-connected networks will remember whatever information is presented to them. Therefore, in reality, it is typically helpful to monitor the network's efficiency on a separate, reserved "validation" set and terminate the network if its efficiency on the validation set begins to decline. The term "early stopping" describes this easy strategy. In actual reality, putting early halting into effect may be rather challenging. As is evident by looking at Figure 6. Developing a criterion to distinguish typical variations from a decreasing trend may take time. Many practitioners train models with changing (constant) epochs and choose the best model on the validation set. This is the most popular method. For the previous two figures, it's clear that we have stopped at epoch number three; as we are seeking the optimal loss and accuracy. Experience the assurance of meticulous experimentation as our dedicated researchers delve into the realm of LSTM models. With unwavering commitment, we have meticulously tested an array of hyper-parameters, leaving no stone unturned.

Experience the cutting-edge capabilities of our proposed model as it undergoes rigorous experimentation with three distinct configurations. Brace the power of LSTM and prepare to be amazed as we push the boundaries even further with LSTM featuring emphasis and dropout layers. As illustrated in Table 2, the performance of various classification algorithms was evaluated based on their achieved accuracies along with other metrics. Among these, the stacked autoencoder [45] demonstrated exceptional accuracy, achieving

an impressive 98.99%. Following closely behind our proposed model, which outperformed all other methods with a remarkable accuracy of 99.58%. In comparison, the firefly classification algorithm [45] secured a respectable accuracy of 91%, showcasing its effectiveness. Additionally, the naive Bayes and RF techniques [46] yielded accuracies of 90.90%, indicating their proficiency in the classification task. The cuckoo search algorithm-trained radial basis function [47] also demonstrated strong performance, garnering an accuracy of 96.9%.

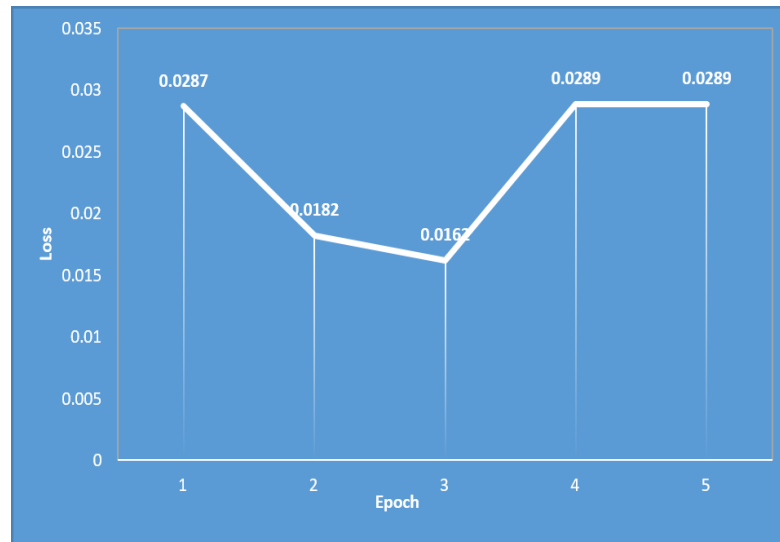


Figure 5. Model loss Vs. epochs

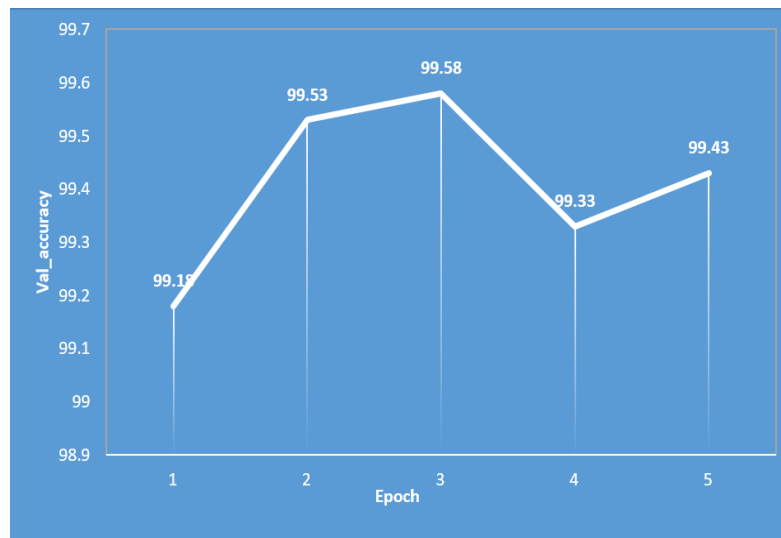


Figure 6. Model accuracy Vs. epochs

Overall, these results highlight the substantial advancements in classification accuracy, with the proposed model emerging as the most promising approach in this evaluation. The study's results section includes precision, recall, and F1 score to comprehensively evaluate the proposed model's performance. The model achieved an accuracy of 99.58%, with precision, recall, and F1 scores all around 0.99, indicating robust detection capabilities with minimal false positives and negatives. It outperformed other algorithms, making it a reliable solution for early DDoS attack detection and prevention. The comprehensive metrics highlight the model's effectiveness in ensuring network security and stability. This underscores the model's potential in enhancing DDoS attack mitigation strategies.

Table 2. Experimental results of the proposed model along with other techniques

Used Technique	Accuracy (%)	Precision	Recall	F1 score
Stacked autoencoder [45]	98.99	0.98	0.99	0.99
Firefly classification algorithm [45]	91	0.90	0.92	0.91
Naive Bayes and RF techniques [46]	90.90	0.89	0.91	0.90
Using cuckoo search algorithm-trained radial basis function [47]	96.9	0.95	0.97	0.96
The proposed archeticure	99.58	0.99	0.99	0.99

The proposed LSTM model for detecting DDoS attacks demonstrates several practical implications, including a high detection accuracy of 99.58%, real-time detection capabilities, and adaptive learning to dynamic environments. This flexibility makes the model applicable across various network scenarios, reducing the need for constant manual monitoring and intervention, thus offering significant cost savings. However, the research also encountered several limitations. The use of synthetic datasets may not fully capture the complexity of real-world DDoS attacks, impacting the model's generalizability. The performance of the model relies heavily on precise feature selection and engineering, and the training process is resource-intensive. While the model is effective in detection, it requires additional mechanisms for mitigation. Furthermore, the evolving nature of DDoS attack patterns necessitates continuous updates to the model to maintain its effectiveness, presenting logistical challenges. Lastly, the accuracy of the model depends on the quality and representativeness of the training data, with potential biases leading to inaccurate detection of novel attacks. Addressing these limitations can enhance the model's robustness and practical applicability in real-world scenarios.

## 6. CONCLUSION

While several tools exist to identify DDoS assaults, many of them don't start looking into the attack until after the fact, when the harm has already been done. A critical component of any security system is a method that can identify DDoS assaults early on and block access to services while they are still in their infancy. The use of such data may aid in the creation of a set of automatic steps to turn on system tools and detect abnormalities or anomalous behaviour early on. A developer of a defensive system needs to put in place an automatic reduction model that can detect and prevent assaults without any intervention from the administration. We think there ought to be a universal criterion for gauging a system's overall capabilities in the context of DDoS mitigation and detection. and this metric should be applicable across all environments. As DDoS attackers have been more creative in their use of current technology to overcome protection, DDoS assaults have grown considerably more challenging to avoid in contemporary years. As a result, DDoS assaults continue to be a serious problem for service providers, despite the fact that researchers have presented a variety of solutions to this issue. In this research, we provide a ML/DL-based DDoS detection strategy that operates across a wide variety of today's networking scenarios, with an accuracy of over 99.5%.

## REFERENCES




- [1] L. Bagdadi and B. Messabih, "Distributed denial of service attacks classification system using features selection and ensemble techniques," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 34, no. 3, pp. 1540-1548, Dec. 2023, doi: 10.11591/ijeecs.v34.i3.pp1868-1878
- [2] S. A. Z. Mghames and A. A. Ibrahim, "Intrusion detection system for detecting distributed denial of service attacks using machine learning algorithms," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 32, no. 1, pp. 304-311, Oct. 2023, doi: 10.11591/ijeecs.v32.i1.pp304-311.
- [3] S. Wani, M. Imthiyas, H. Almohamedh, K. M. Alhamed, S. Almotairi, and Y. Gulzar, "Distributed denial of service (DDoS) mitigation using blockchain—A comprehensive insight," *Symmetry*, vol. 13, no. 2, Jan. 2021, doi: 10.3390/sym13020227.
- [4] M. Guri, Y. Mirsky, and Y. Elovici, "9-1-1 DDoS: attacks, analysis and mitigation," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 218-232, Apr. 2017, doi: 10.1109/EuroSP.2017.23.
- [5] X. Yuan, C. Li, and X. Li, "DeepDefense: identifying DDoS attack via deep learning," in *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 1-8, May 2017, doi: 10.1109/SMARTCOMP.2017.7946998.
- [6] J. K. Chahal, A. Bhandari, and S. Behal, "Distributed denial of service attacks: A threat or challenge," *New Review of Information Networking*, vol. 24, no. 1, pp. 31-103, 2019, doi: 10.1080/13614576.2019.1611468.
- [7] A. B. Dehkordi, M. Soltanaghaei, and F. Z. Boroujeni, "The DDoS attacks detection through machine learning and statistical methods in SDN," *The Journal of Supercomputing*, vol. 77, no. 3, pp. 2383-2415, Jun. 2021, doi: 10.1007/s11227-020-03323-w.
- [8] E. Fazeldehkordi, O. Owe, and T. Ramezanifarkhani, "A language-based approach to prevent DDoS attacks in distributed financial agent systems," in *Computer Security: ESORICS 2019 International Workshops, IOSec, MSTEC, and FINSEC, Luxembourg City, Luxembourg, Revised Selected Papers*, vol. 2, pp. 258-277, 2020, doi: 10.1007/978-3-030-42051-2\_18.
- [9] K. Singh, K. S. Dhindsa, and D. Nehra, "T-CAD: A threshold based collaborative DDoS attack detection in multiple autonomous systems," *Journal of Information Security and Applications*, vol. 51, 2020, doi: 10.1016/j.jisa.2020.102457.
- [10] R. Ndou, J. I. Fadiran, S. Chowdhury, and S. P. Chowdhury, "Performance comparison of voltage and frequency based loss of grid protection schemes for microgrids," in *2013 IEEE Power & Energy Society General Meeting*, pp. 1-5, 2013, doi: 10.1109/PESMG.2013.6672788.

- [11] M. J. Mirchev and S. T. Mirtchev, "System for DDoS attack mitigation by discovering the attack vectors through statistical traffic analysis," *International Journal of Information and Computer Security*, vol. 13, no. 3-4, pp. 309-321, 2020, doi: 10.1504/IJICS.2020.10029285.
- [12] H. Lotfalizadeh and D. S. Kim, "Investigating real-time entropy features of DDoS attack based on categorized partial-flows," in *2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, pp. 1-6, Jan. 2020, doi: 10.1109/IMCOM48794.2020.9001690.
- [13] R. Abubakar *et al.*, "An effective mechanism to mitigate real-time DDoS attack," *IEEE Access*, vol. 8, pp. 126215-126227, 2020, doi: 10.1109/ACCESS.2020.2995820.
- [14] B. Yuan *et al.*, "Minimizing financial cost of DDoS attack defense in clouds with fine-grained resource management," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2541-2554, 2020, doi: 10.1109/TNSE.2020.2981449.
- [15] X. Z. Khooi, L. Csikor, D. M. Divakaran, and M. S. Kang, "DIDA: Distributed in-network defense architecture against amplified reflection DDoS attacks," in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, pp. 277-281, Jun. 2020, doi: 10.1109/NetSoft48620.2020.9165488.
- [16] S. Choi, Y. An, and I. Sasase, "A lightweight detection using bloom filter against flooding DDoS attack," *IEICE Transactions on Information and Systems*, vol. 103, no. 12, pp. 2600-2610, Dec. 2020, doi: 10.1587/transinf.2020EDP7115.
- [17] K. Arora, K. Kumar, and M. Sachdeva, "Impact analysis of recent DDoS attacks," *International Journal on Computer Science and Engineering*, vol. 3, no. 2, pp. 877-883, Feb. 2011.
- [18] A. Praseed and P. S. Thilagam, "DDoS attacks at the application layer: Challenges and research perspectives for safeguarding web applications," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 661-685, Sep. 2018, doi: 10.1109/COMST.2018.2870658.
- [19] F. S. D. Silva *et al.*, "A taxonomy of DDoS attack mitigation approaches featured by SDN technologies in IoT scenarios," *Sensors*, vol. 20, no. 11, p. 3078, 2020, doi: 10.3390/s20113078.
- [20] K. Adhikary, S. Bhushan, S. Kumar, and K. Dutta, "Hybrid algorithm to detect DDoS attacks in VANETs," *Wireless Personal Communications*, vol. 114, pp. 3613-3634, 2020, doi: 10.1007/s11277-020-07549-y.
- [21] K. Srinivasan, A. Mubarakali, A. S. Alqahtani, and A. D. Kumar, "A survey on the impact of DDoS attacks in cloud computing: prevention, detection and mitigation techniques," in *Intelligent Communication Technologies and Virtual Mobile Networks: ICICV 2019*, Springer International Publishing, pp. 252-270, 2020.
- [22] V. Maheshwari, A. Bhatia, and K. Kumar, "Faster detection and prediction of DDoS attacks using MapReduce and time series analysis," in *2018 International Conference on Information Networking (ICOIN)*, pp. 556-561, Jan. 2018, doi: 10.1109/ICOIN.2018.8343180.
- [23] G. Sujatha, Y. Kanchal, and G. George, "An advanced approach for detection of distributed denial of service (DDoS) attacks using machine learning techniques," in *2022 3rd International Conference on Smart Electronics and Communication (ICOSEC)*, pp. 821-827, Oct. 2022, doi: 10.1109/ICOSEC54921.2022.9951944.
- [24] A. Householder, A. Manion, L. Pesante, G. M. Weaver, and R. Thomas, *Managing the threat of denial-of-service attacks*, Pittsburgh, USA: Carnegie Mellon University, Oct. 2001.
- [25] P. W. Singer and A. Friedman, *Cybersecurity: What everyone needs to know*, USA: Oxford University Press, 2014.
- [26] X. Liang and T. Znati, "A long short-term memory enabled framework for DDoS detection," in *2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1-6, Dec. 2019, doi: 10.1109/GLOBECOM38437.2019.9013450.
- [27] K. Yang, J. Zhang, Y. Xu, and J. Chao, "DDoS attacks detection with autoencoder," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*, pp. 1-9, Apr. 2020, doi: 10.1109/NOMS47738.2020.9110372.
- [28] J. Li, Y. Liu, and L. Gu, "DDoS attack detection based on neural network," in *2010 2nd International Symposium on Aware Computing (ISAC)*, pp. 196-199, Nov. 2010, doi: 10.1109/ISAC.2010.5670479.
- [29] Y. N. Soe, P. I. Santosa, and R. Hartanto, "DDoS attack detection based on simple ANN with SMOTE for IoT environment," in *2019 Fourth International Conference on Informatics and Computing (ICIC)*, pp. 1-5, Oct. 2019, doi: 10.1109/ICIC47613.2019.8985853.
- [30] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for consumer internet of things devices," in *2018 IEEE Security and Privacy Workshops (SPW)*, pp. 29-35, May 2018, doi: 10.1109/SPW.2018.00013.
- [31] A. Gaurav, B. B. Gupta, and P. K. Panigrahi, "A novel approach for DDoS attacks detection in COVID-19 scenario for small entrepreneurs," *Technological Forecasting and Social Change*, vol. 177, pp. 1-11, Apr. 2022, doi: 10.1016/j.techfore.2022.121554.
- [32] T. Roempluk and O. Surinta, "A machine learning approach for detecting distributed denial of service attacks," in *2019 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT-NCON)*, pp. 146-149, Jan. 2019, doi: 10.1109/ECTI-NCON.2019.8692243.
- [33] V. Morfino and S. Rampone, "Towards near-real-time intrusion detection for IoT devices using supervised learning and Apache Spark," *Electronics*, vol. 9, no. 3, 2020, doi: 10.3390/electronics9030444.
- [34] J. N. Bakker, B. Ng, and W. K. Seah, "Can machine learning techniques be effectively used in real networks against DDoS attacks?," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1-6, Jul. 2018, doi: 10.1109/ICCCN.2018.8487445.
- [35] E. Balkanli, J. Alves, and A. N. Zincir-Heywood, "Supervised learning to detect DDoS attacks," in *2014 IEEE Symposium on Computational Intelligence in Cyber Security (CICS)*, Orlando, FL, USA, pp. 1-8, Dec. 2014, doi: 10.1109/CICYBS.2014.7013367.
- [36] U. Sabeel *et al.*, "Evaluation of deep learning in detecting unknown network attacks," in *2019 International Conference on Smart Applications, Communications and Networking (SmartNets)*, Sharm El Sheikh, Egypt, pp. 1-6, Dec. 2019, doi: 10.1109/SmartNets48225.2019.9069788.
- [37] N. G. Amma and S. Subramanian, "VCDDeepFL: Vector convolutional deep feature learning approach for identification of known and unknown denial of service attacks," in *TENCON 2018-2018 IEEE Region 10 Conference*, pp. 0640-0645, Oct. 2018, doi: 10.1109/TENCON.2018.8650225.
- [38] J. Chen, Y. T. Yang, K. K. Hu, H. B. Zheng, and Z. Wang, "DAD-MCNN: DDoS attack detection via multi-channel CNN," in *Proceedings of the 2019 11th International Conference on Machine Learning and Computing*, pp. 484-488, Feb. 2019, doi: 10.1145/3318299.3318329.
- [39] A. E. Cil, K. Yildiz, and A. Buldu, "Detection of DDoS attacks with feed forward based deep neural network model," *Expert Systems with Applications*, vol. 169, May 2021, doi: 10.1016/j.eswa.2020.114520.
- [40] S. Haider *et al.*, "A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks," *IEEE Access*, vol. 8, pp. 53972-53983, Feb. 2020, doi: 10.1109/ACCESS.2020.2976908.




- [41] R. R. R. Rinish, S. Rachamalla, M. S. Yoosuf, and G. R. Anil, "Convolutional neural network based intrusion detection system and predicting the DDoS attack," in *Data Intelligence and Cognitive Informatics: Proceedings of ICDICI 2022*, Springer Nature Singapore, pp. 81-94, Dec. 2022, doi: 10.1007/978-981-19-6004-8\_7.
- [42] R. F. Fouladi, O. Ermiş, and E. Anarim, "A novel approach for distributed denial of service defense using continuous wavelet transform and convolutional neural network for software-defined network," *Computers & Security*, vol. 112, Jan. 2022, doi: 10.1016/j.cose.2021.102524.
- [43] S. Singh and S. K. Jayakumar, "DDoS attack detection in SDN: optimized deep convolutional neural network with optimal feature set," *Wireless Personal Communications*, vol. 125, no. 3, pp. 2781-2797, Aug. 2022, doi: 10.1007/s11277-022-09685-z.
- [44] B. I. Hairab, M. S. Elsayed, A. D. Jurcut, and M. A. Azer, "Anomaly detection based on CNN and regularization techniques against zero-day attacks in IoT networks," *IEEE Access*, vol. 10, pp. 98427-98440, Sep. 2022, doi: 10.1109/ACCESS.2022.3206367.
- [45] A. Zheng and A. Casari, *Feature engineering for machine learning: principles and techniques for data scientists*, California, USA: O'Reilly Media, Inc., Apr. 2018.
- [46] E. Kamel, S. Sheikh, and X. Huang, "Data-driven predictive models for residential building energy use based on the segregation of heating and cooling days," *Energy*, vol. 206, 2020, doi: 10.1016/j.energy.2020.118045.
- [47] M. A. Ranzato, Y. L. Boureau, and Y. Cun, "Sparse feature learning for deep belief networks," *Advances in Neural Information Processing Systems*, vol. 20, pp. 1-8, 2007.
- [48] B. Ramsundar and R. B. Zadeh, *TensorFlow for deep learning: from linear regression to reinforcement learning*, California, USA: O'Reilly Media, Inc., 2018.
- [49] S. Yadav and S. Subramanian, "Detection of application layer DDoS attack by feature learning using stacked autoencoder," in *2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)*, pp. 361-366, 2016, doi: 10.1109/ICCTICT.2016.7514608.
- [50] A. Kaliki and K. M. Prasad, "Machine learning based application layer DDoS Attack detection using firefly classification algorithm," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 17, pp. 635-645, 2018.
- [51] G. Ajeetha and G. Madhu Priya, "Machine learning based DDoS attack detection," in *2019 Innovations in Power and Advanced Computing Technologies (i-PACT)*, Vellore, India, pp. 1-5, 2019, pp. 1-5, doi: 10.1109/i-PACT44901.2019.8959961.
- [52] H. Beitollahi, D. M. Sharif, and M. Fazeli, "Application layer DDoS attack detection using cuckoo search algorithm-trained radial basis function," *IEEE Access*, vol. 10, pp. 63844-63854, 2022, doi: 10.1109/ACCESS.2022.3149554.

## BIOGRAPHIES OF AUTHORS






**Ashraf ALDabbas**    received his M.Sc. degree in computer science with a thesis recognized as the best dissertation for the academic year 2013/2014 from Al-Balqa Applied University, Al-Salt, Jordan. He then went on to achieve his Ph.D. degree in Engineering and technological informatics, graduating First in his class with Honors, from the University of Debrecen, Hungary. His research interests lie at the intersection of sensor technology, data science, and the internet of things (IoT), pattern recognition, artificial intelligent, and digital signal processing. He is particularly passionate about exploring remotely sensing methods, sensor data quality assessment techniques, and the application of deep learning in wireless sensor networks. This confluence of interests positions him to make significant contributions to advancements in data acquisition, analysis, and knowledge extraction from sensor networks. He can be contacted at email: ashraf.dabbas@bau.edu.jo.






**Laith H. Baniata**    earned his Ph.D. in computer science and engineering from Kyungpook National University, Daegu, South Korea, in 2019. He served as a post-doctoral Research Fellow at the NLP and Deep Learning Laboratory at Kyungpook National University from January 2021 to January 2022. From April 2022 to February 2024, he worked as an Assistant Professor at the School of Computing, Gachon University. Since April 2024, he has been an Assistant Professor at the IT College of Al-Ahliyya Amman University. His research interests include natural language processing and deep learning. He can be contacted at email: laith.baniata@bau.edu.jo.






**Bayan A. AlSaaidah**    holds a Ph.D. in computer engineering from the University of Liverpool, United Kingdom in 2019. She also received her B.Sc. and M.Sc. (computer engineering and computer science) from Al-Balqa Applied University, Jordan in 2007 and 2011, respectively. She is currently an Assistant Professor at the Department of Computer Science at Al-Balqa Applied University, Salt, Jordan. Her research includes computer vision, machine learning, artificial intelligence, pattern recognition, classification, image processing, feature selection, object recognition, signal, image and video processing, image segmentation, image data analysis, pattern classification, and cluster analysis. She has published many studies in international journals and conferences. She can be contacted at email: bayan-saaidah@bau.edu.jo.






**Zaid Mustafa**    received his Ph.D. Degree in computer information systems and computer graphics, from Santiago de Compostela University -Spain (2014-2019). Previously M.Sc. computer information systems (2012) from Middle East University and B.Sc. software engineering (2007) from Al-Balqa Applied University, Al-Salt, Hashemite Kingdom of Jordan. Currently, he serves as an Assistant Professor at Al-Balqa Applied University, Prince Abdullah bin Ghazi Faculty of Information and Communication Technology, Jordan (2019-present), and is also Head at Department of Computer Information Systems (2022–present). His research interests include machine learning, human-computer interaction, developing geographic information system (GIS) tools, virtual reality, augmented reality, internet of things (IoT), and smart cities. He can be contacted at email: Z\_lami@bau.edu.jo.



**Dr. Muath Alali**    earned his Ph.D. in intelligent systems from Universiti Putra Malaysia in 2023. He is currently an Assistant Professor at the Applied Science Private University, Faculty of Information Technology. From 2010 to 2011, he was a lecturer of computer science at Jerash University. He worked as a lecturer at Qassim University in Buridah, Saudi Arabia, from 2011 to 2016. His research interests include cybersecurity, sentiment analysis, text classification, machine learning, deep learning, ordinal classification, transfer learning, and transformers. He can be contacted at email: M\_alali@asu.edu.jo.



**Dr. Roqia Rateb**    has received her Ph.D. in computer science (artificial intelligence/data science) from University Utara Malaysia (UUM) in 2020. Her master has completed from Yarmouk University in computer information systems (data mining) with excellent. She succeeded as the first of the batch and her ranking became on honor's list of the University in all years of the study. Moreover, she has received her bachelor in computer science from Jordan University of Science and Technology with excellent. She is currently an Assistant Professor at Department of Computer Science in Al-Ahliyya Amman University (AAU), Amman, Jordan. Her research interests are modelling dynamics of (multi) agent in practical application areas, such as social simulation, agents and cognitive robotics, problem optimization, high-performance computing, cognitive simulation and modelling, and configuration algorithm. In addition, her latest publication achieved a Certificate of Award for the Best Special Session. She can be contacted at email: r.alshorman@ammanu.edu.jo.