Hindawi
Security and Communication Networks
Volume 2023, Article ID 9810961, 1 page
https://doi.org/10.1155/2023/9810961

Retraction

Retracted: DDoS Attack Detection by Hybrid Deep Learning Methodologies

Security and Communication Networks

Received 5 December 2023; Accepted 5 December 2023; Published 6 December 2023

This article has been retracted by Hindawi, as publisher, following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of systematic manipulation of the publication and peer-review process. We cannot, therefore, vouch for the reliability or integrity of this article.

Please note that this notice is intended solely to alert readers that the peer-review process of this article has been compromised.

Wiley and Hindawi regret that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] L. Xinlong and C. Zhibin, "DDoS Attack Detection by Hybrid Deep Learning Methodologies," *Security and Communication Networks*, vol. 2022, Article ID 7866096, 7 pages, 2022.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 7866096, 7 pages
https://doi.org/10.1155/2022/7866096

Research Article

DDoS Attack Detection by Hybrid Deep Learning Methodologies

Li Xinlong[1] and Chen Zhibin[2]

[1]*School of Computer Science, Hunan Institute of Technology, Hengyang 421002, China*
[2]*Admissions and Career Service Office, Hunan Institute of Engineering, Xiangtan 411104, China*

Correspondence should be addressed to Chen Zhibin; czb@hnie.edu.cn

Received 15 April 2022; Revised 15 May 2022; Accepted 19 May 2022; Published 31 May 2022

Academic Editor: Muhammad Arif

A Distributed Denial of Service (DDoS) attack occurs when large amounts of traffic from hundreds, thousands, or even millions of other computers are routed to a network or server to crash the system and disrupt its function. These attacks are commonly used to shut down websites or applications temporarily. Such problems often need to be addressed with models that can manage the time information contained in network traffic flows. In this work, we apply a Hybrid Deep Learning method to detect malicious web traffic in the form of DDoS attacks, controlling the web flow of information reaching a server, using any dependencies between the different elements of a data stream. An original and cutting-edge Hierarchical Temporal Memory (HTM) hybrid model has been proposed. The operation of this model is predicated primarily on the portion of the cerebral cortex known as the neocortex. The neocortex is in charge of various fundamental brain functions, including the perception of senses, the comprehension of language, and the control of movement. For the hybrid implementation to be capable of encoding time sequences that incorporate incoming data, a Long Short-Term Memory (LSTM) shell is added.

1. Introduction

An attempt to render an online service inaccessible owing to an excessive volume of traffic coming from several dispersed sources is known as a DDoS attack [1]. These attacks, which target a wide variety of crucial resources ranging from banks to news websites, provide a massive obstacle in ensuring that individuals have unfettered access to vital information and can freely share it. DDoS attacks are designed to look like a flood of calls, or requests, made by browsers asking a web page to load. It is the equivalent of thousands of visitors to a given site getting there simultaneously and visiting. The high number of calls causes the server that hosts the website to become overwhelmed, and as a result, it gives a message stating that it is unable to provide service. Visitors interested in accessing the website will be unable to do so as a result of this action [2].

During a DDoS attack, it is inevitable that the victim server will receive a large amount of information in a relatively short period [3]. This information is separated into data packets, which will at least share several standard features if they are not identical. By looking at each of these packages separately, identifying them as part of a malicious network can be tedious. These packages are prebuilt to not deviate significantly from the nonmalicious packets. On the other hand, considering each packet only as part of a more extensive sequence that extends over time, we are allowed to collectively examine them, capable of revealing their true significance [3]. To put it simply, organizing data into time sequences enable us to take a step back and look at the "big picture," which makes it more evident whether a server is under attack or not. We, therefore, conclude that the time frame in which each element of the data set is located in a piece of critical information should in no case be ignored [4].

Various neural architectures, e.g., Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) networks [5], are well suited for managing such data sequences. Our goal is to train the above models using data sets, the elements of which have been organized in individual time sequences to lead to algorithms of the highest possible performance. In particular, an innovative hybrid model of the HTM [6] system is proposed, in the architecture of which

an LSTM cell is added, so that the system can encode time sequences containing inflow data [7–9].

To achieve this goal, we will first look at some contemporary work that deals with solving the problem of cyber-attack detection, focusing mainly on the use of neural network models. In the following, we will present the proposed application method for solving the problem, the scenario considered for the modeling of the problem, the results of the process, and finally, a commentary on the methods and the possible ways of developing the research method in question.

2. Related Literature

As in numerous other areas in network security, several different deep learning algorithms continue to be used tirelessly to establish secure Internet communication between devices. Especially in recent years, early detection and response to cyber-attacks have become very important. Due to the pandemic, the smooth operation of servers related to the provision of services and products via the Internet is more critical than ever [10–12]. This has made such servers even bigger targets for attacks, which has led to the publication of various related tasks in a short period.

Barati et al. [13] suggested a DDoS assault detection system framework. In a hybrid technique, a Genetic Algorithm and an Artificial Neural Network were used for characteristic identification and threat detection, respectively. The most efficient features are chosen using a layering technique based on GA, and the recognition rate of DDoS attacks was increased using ANN's Multilayer Perceptron (MLP). The findings showed that the suggested approach could identify DDoS attacks with excellent precision and a low chance of false alarm. They intended to use similar themes to conduct further tests on other data sets to assess the experiment's resilience.

Hosseini and Azizi [14] provided a method for identifying DDoS attacks using gradual training depending on a data stream methodology. To quickly organize the activity, they devised a method that allocated the computation responsibility between the client and proxy components based on the resources available to each of those portions. The consumer side had three stages: first, the client service's data collection; second, component recovery based on forwarding classification for each method; and finally, the differentiation test. As a result, if the deviation exceeded a certain threshold, the assault was detected; otherwise, data were sent to the intermediary side. On the proxy side, they employed the nave Bayes, random forest, decision tree, multilayer perceptron (MLP), and k-nearest neighbors (K-NN) to get superior results. Distinct assaults have diverse tendencies, and the necessary efficiency for identifying assaults and more capacity to differentiate novel threat patterns is obtained thanks to different chosen characteristics for each method. The findings suggest that the random forest method outperforms the other techniques.

Shurman et al. [1] offered two approaches for detecting Distributed Reflection Denial of Service (DrDoS) assaults on the Internet of Things. The first way is a hybrid-based IDS for IoT networks, which involves providing an IDS framework scheme specified as an application capable of detecting abnormal data traffic from any network node and running IP datasets against it. It was able to detect strange IP packets and ban unaccepted IPs before they escalated into possible DoS threats. The second technique used a deep training system based on LSTM that was trained on the CICDDoS2019 dataset containing different types of DrDoS assaults and was able to identify them. Their findings showed that the suggested approaches could identify malicious behavior, ensuring the safety of the IoT network. They wanted to create a new deep learning model to identify the second kind of DDoS assault in the CICDDoS2019 dataset and evaluate the performance of these approaches in a real-world system.

To correctly forecast DDoS assaults utilizing benchmark data, Alghazzawi et al. [15] recommended employing a hybrid deep training (DL) model, specifically a CNN with BiLSTM (bidirectional long memory). Only the most essential characteristics were chosen by rating and selecting the features that rated the best in the supplied data set. The suggested CNN-BI-LSTM achieved an efficiency of up to 94.52 percent utilizing the data set CIC-DDoS2019 throughout training, testing, and validation, according to the results of the experiments. Using a unique data set, a single statistical approach, the chi-squared test to select relevant characteristics, and the utilization of hidden states instead of a pretrained CNN model were all limitations of their system. They intended to evaluate the usage of multiple traffic data sets and alternative feature selection techniques to the chi-squared evaluation and pretrained word encoding algorithms such as autoencoders, Glove, and Fasttext.

In a Software Defined Network setting, Deepa et al. [16] suggested a hybrid deep learning approach to identify DDoS attacks. They've also evaluated their work using three performance criteria: accuracy, precision, and false alert rate. In contrast to the SVM method, SOM is an uncontrolled machine learning algorithm that performs well in detecting assaults. However, when compared to a basic machine learning model, they obtained higher accuracy, detection rate, and reduced false alarm rate utilizing their suggested hybrid machine learning model. By enforcing security restrictions in the flow table, they planned to develop ensemble deep learning models to identify DDoS attacks in the data plane.

3. The Proposed Hybrid Deep Learning HTM

The proposed methodology aims to construct a model that separates benign from malicious traffic. Each HTM system consists of several regions or levels organized according to a hierarchy. The organization of the areas is quite reminiscent of the organization of the levels of classical neural networks. Specifically, the first area receives an input pattern based on which it produces an output, which is then fed to the next area defined by the hierarchy [6, 17]. This process is repeated until the last area produces the network's outcome. Depending on its position in the hierarchy, each region "learns" to recognize different characteristics of the input.

Areas that are low in the hierarchy are associated with learning primary and general features, while as we approach the higher areas, the features become more and more abstract. To understand the function of regions, we need to introduce two more concepts: column and cell. An area essentially consists of several columns, usually (not necessarily) organized in a two-dimensional table [18, 19]. Each column, in turn, consists of several cells that may be linked to other cells within the same region. These connections are made based on specific synapses, which belong to the dendrite segments of cells [17, 20].

There are generally two types of dendritic sections, depending on the type of connection they make [19, 21, 22]:

(1) *Proximal Dendrite Segment.* This type of dendrite segment includes the synapses that connect the columns of an area to the entrance to it, whether it comes from the immediately preceding area in the hierarchy or directly from a source, e.g., by a sensor. We, therefore, observe that the columns are treated as single computing units, each of which corresponds to a separate central dendritic section, which in turn contains a particular set of synapses. This means that each column may be linked to a slightly different portion of the entrance to the area.

(2) *Distal Dendrite Segment*. Each cell has more than one peripheral dendrite segment, the synapses of which connect it to other cells within the same region. It, therefore, becomes apparent that the number of peripheral dendritic sections is much larger than the number of central ones. For example, suppose a region has 100 columns of 10 cells, each corresponding to 5 peripheral dendritic segments. In that case, the number of central dendritic segments is only 100, while the number of peripherals equals $100 \cdot 10 \cdot 5 = 5{,}000$. Each of the synapses of the dendritic segments corresponds to a binary weight, or more simply, each synapse will be considered active or inactive. In addition, each conclusion corresponds to another value, also known as permanence value, which in essence determines whether a deduction is deemed to be connected or not.

The method by which an HTM system predicts the class of an input element is quite simple and understandable. The model's output is only a binary string, which indicates the active cells of the last region of the system. Based on this output string, the algorithm sorts the element at the input by examining the total number of activated cells and the general frequency of their activation, which should have been counted in advance during the training. More specifically [6, 18, 22, 23], the algorithm maintains a table C of dimensions $c \times n$, where c is the number of different classes of each problem. In contrast, n is the length of the output string, i.e., the total number of cells in the last region of the system. Each line C_i,: of table C corresponds to a different class i, while each value $C_{i,j}$ expresses the number of iterations during the system training for which cell j was active, given that each input element led to its activation belonged to class i. Therefore, through this table, the algorithm attempts to "learn" which classes lead to the activation of which cells and with what frequency. After completing the system training based on the respective data set, the algorithm divides the values of each column C_{ij} of the table by their sum; this value expresses the total frequency of activation of cell j, so that each column is converted into one probability distribution. The values of each column C_{ij} of the table therefore now express the probability of each class i given that cell j is active. After performing the above-mentioned procedure, the model is considered fully trained and can perform predictions about the class of a new element at the input by following the following three steps [6, 20]:

(1) Given an input element, the HTM system generates the output string, which indicates the active cells of the last region of the system

(2) Let A be the set containing the markers of each active cell. For each class i of the problem, we calculate the following value:

$$p_i = \prod_{j \in A} \mathbf{C}_{i,j}. \tag{1}$$

(3) Finally, the algorithm predicts the class of the input element as the class k, which corresponds to the highest of the p_i values, i.e.,

$$k = \arg\max_i (p_i). \tag{2}$$

It should be noted that although the values of the p_i values indicate which of the classes the input element is most likely to belong to, the values themselves are not probabilities. However, we can divide each p_i value by their total sum in each case to construct such a probabilistic distribution. Therefore, the algorithm can classify any new input element through the above-given procedure. At this point, we should also note that table C should not be further modified after the training, i.e., counting the cell activation frequency is considered complete.

To solve the problem, the HTM system should be able to rate each input element by assigning it a value of $a \in [0, 1]$, which expresses the probability that each component is an anomaly, always compared to the data based on which the algorithm is trained. Therefore, by feeding the algorithm elements exclusively related to the malicious network traffic, we can consider that any "abnormal" element detected by the system is a sign of malicious traffic. We, therefore, observe that this problem belongs to the category of Non-supervised Learning problems since the existence of the corresponding labels of the elements of the education set is not necessary [17, 18, 23].

The procedure we follow is based on the Explosion mechanism of the columns. One of the three possible states of cells is the "prediction state," through which the system expresses the predictions it makes regarding the next input element. When one or more cells in column i are in a predicted state during the t-th iteration of the algorithm, this

can be interpreted as a system prediction that the next input element might trigger column i. Therefore, if the activation of this column is observed during the repetition $t+1$, then the prediction of the system is considered successful with the algorithm proceeding with the activation of the cells of the column which are in the predicted state. If, on the other hand, the column is not activated, then this means that the prediction turned out to be wrong, resulting in a reduction in the permanence values of the conclusions that contributed to its execution as a kind of "punishment" [17, 24, 25].

However, if a column that does not contain any cells in a predicted state is activated, then the Explosion mechanism is executed through which each of its cells is activated. The triggering of this mechanism indicates that the model is receiving unexpected information, which it may have never encountered before. During the early phase of system training, the Explosion mechanism's high execution frequency is considered normal as the model does not stop discovering new patterns. Through the Explosion mechanism, the system is led to the activation of more cells, which increases the probability of the cells in the area entering a predicted state, which in turn leads to faster training of the system, as the more cells are in an expected state, the more training processes are performed during the execution of the algorithm. However, under the circumstances in which a fully trained model receives data consistent with the data on which it is trained, the Explosion mechanism should not be common. For this reason, this mechanism is considered an indication that the system may have accepted an input element with peculiarities. Specifically, the more columns trigger the Explosion mechanism, the more likely the input element is an anomaly [11, 26].

Since the number of columns that trigger the Explosion mechanism will always be less than or equal to the total number of active columns, and therefore, each value will belong to the value interval [0, 1], i.e., in essence, it will be a probability. If an input element x_j we have $a_j = 0$, then this means that the model successfully predicted the activation of each column. On the other hand, if the a_j value is equal to the unit, then this means that the model's predictions were all wrong, which is probably due to the peculiarity of each input element.

Having described the procedure for calculating the degree of an anomaly, we need to set a threshold value based on which it will be decided whether an item corresponds to a benign or malicious move. Therefore, the anomaly detection problem becomes a Binary Classification problem through this process. The threshold value calculation method assumes that the degree of anomaly extracted by a fully trained model based on its training data will follow the exponential distribution as a function of probability density $p(x;\lambda) = \lambda e - \lambda x$ of unknown parameter $\lambda > 0$. Given that the whole set of education represents the entire class data population, it is reasonable to assume that the degrees of the anomaly of the data will follow an exponential distribution [13, 27, 28].

Having made the above-given hypothesis, we calculate the threshold value since the anomalous values of the distribution are beyond the point defined as the sum Q3 + IQR, where Q3 is the third quadrant of the distribution, and IQR = Q3 − Q1, the interquartile range. Specifically for the case of the exponential distribution, this point, even if it is, is calculated based on the following equation [29–31]:

$$\theta_a = \frac{\ln(4) + 1.5 \cdot \ln(3)}{\lambda}. \tag{3}$$

Therefore, all that remains is to calculate the parameter λ of the exponential distribution of the degrees of anomaly extracted by the model. We use the Maximum Likelihood Estimation method to solve this problem, defining as the final value λ the value $\widehat{\lambda}$ which maximizes the probability of the degrees of an anomaly if their values follow the exponential distribution. More specifically, given a set of degrees of anomaly $A = \{a_1, a_2, \ldots, a_n \mid a_i \geq 0\}$, with the values a_i of the set coming from the same exponential distribution of unknown parameter λ, the probability $L_n(\lambda; A)$ of the values of the set A is calculated as follows [29, 31, 32]:

$$\begin{aligned} L_n(\lambda; A) &= p(A;\lambda) \\ &= \prod_{i=1}^{n} p(a_i;\lambda) = \prod_{i=1}^{n} \lambda e^{-\lambda \cdot a_i} = \lambda^n \cdot \exp\left(-\lambda \sum_{i=1}^{n} a_i\right). \end{aligned} \tag{4}$$

However, it is common to use the probability logarithm $l_n(L_n)$ instead of the probability itself, as this practice leads to simpler calculations. The result, of course, remains the same as the logarithmic function is genuinely increasing [25, 33, 34].

$$\begin{aligned} ln\left[L_n(\lambda; A)\right] &= ln\left[\lambda^n \cdot \exp\left(-\lambda \sum_{i=1}^{n} a_i\right)\right] \\ &= ln(\lambda^n) + ln\left[\exp\left(-\lambda \sum_{i=1}^{n} a_i\right)\right] \\ &= n \cdot ln(\lambda) - \lambda \sum_{i=1}^{n} a_i. \end{aligned} \tag{5}$$

The final value $\widehat{\lambda}$ is therefore calculated as follows:

$$\widehat{\lambda} = \arg\max_{\lambda > 0}\left[n \cdot \ln(\lambda) - \lambda \sum_{i=1}^{n} a_i\right]. \tag{6}$$

We can now define the threshold θ_α, based on which each item will be classified into one of the following two classes:

$$\theta_a = \frac{\ln(4) + 1.5 \cdot \ln(3)}{\widehat{\lambda}}. \tag{7}$$

The addition implemented by the hybrid scheme concerns the ability to compute the method of retrospective sequence relations using earlier data, the elements of which have been organized in individual time sequences to lead to the highest possible performance. In particular, the solid forms of the equations are used for the forward passage of an LSTM cell with a forget gate which are [7, 35, 36]

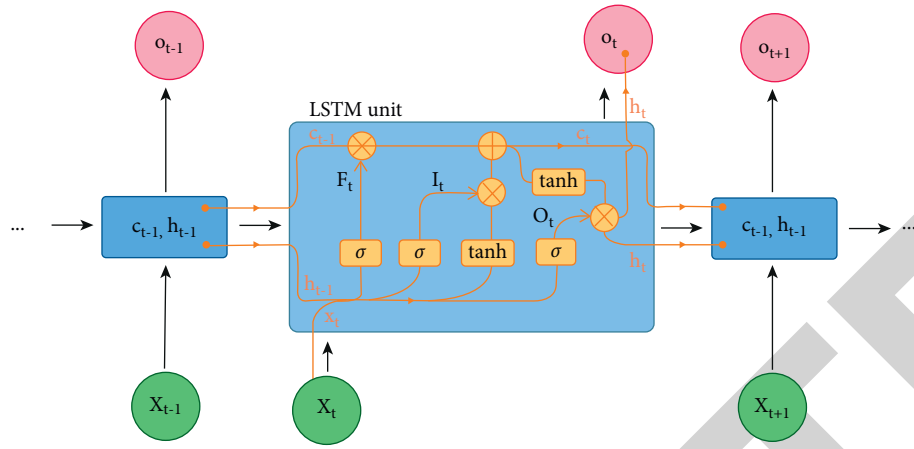

Figure 1: LSTM memory unit.

$$
\begin{aligned}
f_t &= \sigma_g\left(W_f x_t + U_f h_{t-1} + b_f\right),\\
i_t &= \sigma_g\left(W_i x_t + U_i h_{t-1} + b_i\right),\\
o_t &= \sigma_g\left(W_o x_t + U_o h_{t-1} + b_o\right),\\
\tilde{c}_t &= \sigma_c\left(W_c x_t + U_c h_{t-1} + b_c\right),\\
c_t &= f_t \circ c_{t-1} + i_t \circ \tilde{c}_t,\\
h_t &= o_t \circ \sigma_h\left(c_t\right),
\end{aligned}
\tag{8}
$$

with the index t handling the time step of the process. In the case of a retrospectively recurring continuous-time neural network, the model uses a system of ordinary differential equations to model the results on an incoming input neuron. For example, for a neuron i in the network with activation, the change activation rate is given by the following equation [7, 8, 37]:

$$
\tau_i \dot{y}_i = -y_i + \sum_{j=1}^{n} w_{ji}\sigma\left(y_j - \Theta_j\right) + I_i(t). \tag{9}
$$

Adding an LSTM memory unit like the one shown in Figure 1 prevents the disappearance or explosion of errors propagating to the rear architectures. Instead, errors can flow backward through an unlimited number of virtual levels that unfold in space. LSTM can learn tasks that require recollection of events that occurred thousands or even millions of discrete time steps earlier.

The proposed LSTM operates even with long delays between essential events and can handle signals that combine low and high-frequency components.

4. Experiments

For DDoS attacks, although many statistical methods have been designed to detect them, developing a real-time detector with a low computational cost is still one of the main concerns. On the other hand, the evaluation of new algorithms and detection techniques is primarily based on well-designed data sets. In this paper, we review the performance of the proposed system using a complete CICDDoS2019 dataset, which fixes all current deficiencies. The implementation of the set is based on

Table 1: Performance of compared methods.

Model	Accuracy	ROC	Recall	Precision	F-score
HTM-LSTM	0.9774	0.9982	0.9792	0.9720	0.9772
LSTM	0.9355	0.9767	0.9335	0.9342	0.9341
RNN	0.9142	0.9688	0.9129	0.9137	0.9141
XGBoost	0.8949	0.9666	0.8660	0.8960	0.8955
LightGBM	0.8944	0.9659	0.8945	0.8966	0.8945
CatBoost	0.8946	0.9663	0.8950	0.8950	0.8945
SVM	0.8958	0.9639	0.8985	0.8965	0.8968
Random Forest	0.8864	0.9646	0.8835	0.8854	0.8853
FFNN	0.8646	0.9498	0.8688	0.8665	0.8656
k-Neighbors	0.8686	0.9494	0.8636	0.8600	0.8652

*Hierarchical Temporal Memory (HTM); Long Short-Term Memory (LSTM); Recurrent Neural Network (RNN); Extreme Gradient Boosting (XGBoost); Light Gradient Boosted Machine (LightGBM); Support-Vector Machines (SVM); FeedForward Neural Network (FFNN).

techniques where TCP and UDP packets are sent to random ports on the target machine at a very high rate. As a result, the available network bandwidth is depleted, the system is shut down, and performance is degraded. The architecture of the testbed environment is shown in https://www.unb.ca/cic/datasets/ddos-2019.html.

A detailed description of the set and ways of creating the attacks can be found in the work of Sharafaldin et al. [3]. For the evaluation of the system, an extensive comparison was made with other classical machine learning models and competing deep learning models. The results obtained are presented below in Table 1.

Table 1 shows the clear superiority of the method in the totality of the evaluation characteristics. This fact proves that the proposed architecture provides us with two main advantages. First and foremost is that we do not need data labels, so the problem-solving method remains entirely in the context of Unsupervised Learning. The second advantage is that the threshold value is calculated based on the training set data, i.e., the method does not force us to look for further data. It should also be noted that when calculating the value of the threshold parameter λ, we prefer to subtract excessively high degrees of anomaly (≥ 0.9), if any, as their existence may mistakenly lead to higher threshold values. Finally, we must

refer to an observation that exclusively concerns the HTM systems and the Explosion mechanism. As we have seen, this mechanism plays a central role in calculating the degree of the anomaly of the input elements. The more times an element triggers the mechanism, the greater the probability that this element is an anomaly [11, 23, 38].

However, when supplying the HTM system with the first element of a new sequence, the Explosion mechanism is certain to be triggered as no cell in the region is in the predictive state, which sets each cell in the region as inactive. This means that any input element that is the first element of the sequence in which it is contained is doomed always to receive a degree of anomaly equal to one, which is not in line with reality. For this reason, in the context of anomaly detection, we should avoid organizing the data into time sequences, which simply means that retrospective operation should not be called at the beginning of each new sequence.

5. Conclusions

DDoS attacks, which are constantly being improved upon in terms of their methodology, are among the most essential and complicated concerns in information security. Dealing with these systems calls for highly developed computer programs that can use time sequences and other generally advanced intelligence qualities to conquer complex challenges. In this spirit, an innovative hybrid model of the HTM system is provided in this study. The system's architecture is modified by the addition of an LSTM cell so that the system can encode time sequences that comprise inflow data. Experiments showed that the proposed methodology successfully resolved the issue of accurately detecting DDoS attacks. This opened the door for additional research into how we may apply the process to sequential learning challenges.

An essential realization is that the proposed system can be a function of mapping the input data on the cells of the system's final area, incorporating any spatial and temporal information discovered between the data. This realization is important because it demonstrates how we can use the proposed approach. Different input data, that is, different values of the characteristics of the original input vector, and at different time frames within each sequence will lead to the activation of other cells. This can be thought of as the same thing as saying that the characteristics of the original input vector will have different values. Therefore, it is easy to see that by attempting to map the relationship between cells and input data, and it is highly likely that we will be able to interpret the decisions of the proposed system, as is the case with simpler models such as decision trees. This is because it is easy to see that by attempting to map the relationship between cells and input data, it is highly likely that we will be able to map the relationship between cells and input data. However, this method demands work and a significant amount of research, both of which we might address in a later line of investigation.

Data Availability

The data used to support the study are included in the paper.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This study was supported by the Foundation of Hunan Educational Committee (Grant Nos. 19C0533 and 20A144), Henan Province Department.

References

[1] M. Shurman, R. Khrais, and A. Yateem, "DoS and DDoS attack detection using deep learning and IDS," *The International Arab Journal of Information Technology*, vol. 17, no. 4A, pp. 655–661, 2020.

[2] B. Nugraha, N. Kulkarni, and A. Gopikrishnan, "Detecting adversarial DDoS attacks in software- defined networking using deep learning techniques and adversarial training," *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, in *Proceedings of the 2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, pp. 448–454, Rhodes, Greece, July. 2021.

[3] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proceedings of the 2019 International Carnahan Conference on Security Technology (ICCST)*, pp. 1–8, Chennai, India, October 2019.

[4] R. Chauhan and S. Shah Heydari, "Polymorphic Adversarial DDoS attack on IDS using GAN," *2020 International Symposium on Networks, Computers and Communications (ISNCC)*, in *Proceedings of the 2020 International Symposium on Networks, Computers and Communications (ISNCC)*, pp. 1–6, Montreal, QC, Canada, October. 2020.

[5] H. Sak, A. Senior, and F. Beaufays, "Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition," Feb. 2014, http://arxiv.org/abs/1402.1128.

[6] D. Maltoni, "Pattern recognition by hierarchical temporal memory," *SSRN Electronic Journal*, 2011.

[7] S. Yang, X. Yu, and Y. Zhou, "LSTM and GRU neural network performance comparison study: taking yelp review dataset as an example," *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, in *Proceedings of the 2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, pp. 98–101, Shanghai, China, June. 2020.

[8] L. Yao and Y. Guan, "An improved LSTM structure for natural language processing," *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, in *Proceedings of the 2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, pp. 565–569, Chongqing, China, December 2018.

[9] Y. Zhang, Y. Wang, and J. Yang, "Lattice LSTM for Chinese sentence representation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1506–1519, 2020.

[10] K. Demertzis, D. Taketzis, D. Tsiotas, L. Magafas, L. Iliadis, and P. Kikiras, "Pandemic analytics by advanced machine learning for improved decision making of COVID-19 crisis," *Processes*, vol. 9, no. 8, p. 1267, 2021.

[11] R. A. Shaikh, A. A. Iqbal, and K. Samad, "Review over anomaly detection algorithms for detecting SYN flooding attacks," in *Proceedings of the 2005 Student Conference on*

Engineering Sciences and Technology, pp. 1–5, Karachi, Pakistan, August. 2005.

[12] R. Garg and S. Jeevaraj, "Effective fake news classifier and its applications to COVID-19," in *Proceedings of the 2021 Effective Fake News Classifier and its Applications to COVID-19 (IBSSC)*, pp. 1–6, Gwalior, India, November 2021.

[13] M. Barati, A. Abdullah, N. I. Udzir, R. Mahmod, and N. Mustapha, "Distributed Denial of Service detection using hybrid machine learning technique," in *Proceedings of the 2014 International Symposium on Biometrics and Security Technologies (ISBAST)*, pp. 268–273, Kuala Lumpur, Malaysia, August. 2014.

[14] S. Hosseini and M. Azizi, "The hybrid technique for DDoS detection with supervised learning algorithms," *Computer Networks*, vol. 158, pp. 35–45, 2019.

[15] D. Alghazzawi, O. Bamasag, H. Ullah, and M. Z. Asghar, "Efficient detection of DDoS attacks using a hybrid deep learning model with improved feature selection," *Applied Sciences*, vol. 11, no. 24, p. 11634, Dec. 2021.

[16] V. Deepa, K. M. Sudar, and P. Deepalakshmi, "Detection of DDoS attack on SDN control plane using hybrid machine learning techniques," in *Proceedings of the 2018 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pp. 299–303, Tirunelveli, India, December. 2018.

[17] T. Van Nguyen, N.-T. Le, J. An, and K.-S. Min, "Defect-resilient memristor crossbar of hierarchical temporal memory (HTM) spatial pooling for near-IoT-sensor cognitive computing," in *Proceedings of the 2020 27th 2020 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 1–4, Glasgow, UK, August. 2020.

[18] A. Barua, D. Muthirayan, P. P. Khargonekar, and M. A. Al Faruque, "Hierarchical temporal memory based machine learning for real-time, unsupervised anomaly detection in smart grid: WiP abstract," in *Proceedings of the 2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, pp. 188-189, NSW, Australia, April. 2020.

[19] D. Fan, M. Sharad, A. Sengupta, and K. Roy, "Hierarchical temporal memory based on spin-neurons and resistive memory for energy-efficient brain-inspired computing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 9, pp. 1907–1919, 2016.

[20] X. Zhituo, R. Hao, and W. Hao, "A content-based image retrieval system using multiple hierarchical temporal memory classifiers," in *Proceedings of the 2012 2012 Fifth International Symposium on Computational Intelligence and Design*, vol. 2, pp. 438–441, Hangzhou, China, October. 2012.

[21] O. A. Kozhushko and M. S. Tarkov, "Using hierarchical temporal memory for document ranking system identification," in *Proceedings of the 2015 Using hierarchical temporal memory for document ranking system identification (SIBCON)*, pp. 1–4, Omsk, Russia, May . 2015.

[22] T. Xu, H. Yang, A. Yu, B. Bao, H. Guo, and Y. Jiang, "Traffic optimization and congestion prevention strategy based on hierarchical temporal memory of data center optical network," in *Proceedings of the 2020 Traffic Optimization and Congestion Prevention Strategy Based on Hierarchical Temporal Memory of Data Center Optical Network (OECC)*, pp. 1–3, Taipei, Taiwan, July. 2020.

[23] J. Diao and H. Kang, "An integrated hierarchical temporal memory network for real-time continuous multi-interval prediction of data streams," in *Proceedings of the 2014 Sixth International Symposium on Parallel Architectures, Algorithms and Programming*, pp. 285–288, Beijing, China, July. 2014.

[24] G. Alessandrini, M. V. D. Hoop, R. Gaburro, and E. Sincich, "Lipschitz stability for a piecewise linear Schrödinger potential from local Cauchy data," *Asymptotic Analysis*, vol. 108, no. 3, pp. 115–149, 2018.

[25] K. Demertzis, L. Iliadis, and P. Kikiras, "A lipschitz - shapley explainable defense methodology against adversarial attacks," *in IFIP International Conference on Artificial Intelligence Applications and Innovations, Hersonissos, Crete, Greece*, pp. 211–227, Springer, Cham, Switzerland, 2021.

[26] N. Elmrabit, F. Zhou, F. Li, and H. Zhou, "Evaluation of machine learning algorithms for anomaly detection," in *Proceedings of the 2020 Evaluation of Machine Learning Algorithms for Anomaly Detection (Cyber Security)*, pp. 1–8, Dublin, Ireland, June. 2020.

[27] Z. Gu and Y. Yang, "Detecting malicious model updates from federated learning on conditional variational autoencoder," in *Proceedings of the 2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 671–680, Portland, OR, USA, May 2021.

[28] S. Guopan, "The effect of probability on risk perception and risk preference in decision making," in *Proceedings of the 2010 The effect of probability on risk perception and risk preference in decision making*, pp. 690–693, Cairo, Egypt, November. 2010.

[29] B. H. H. Gade, C. N. Vooren, and M. Kloster, "Probability distribution for association of maneuvering vehicles," in *Proceedings of the 2019 22th International Conference on Information Fusion (FUSION)*, pp. 1–7, Ottawa, ON, Canada, July. 2019.

[30] T. W. Anderson, *An Introduction to Multivariate Statistical Analysis*, Wiley, NJ, USA, 2003.

[31] J. L. Pollock, "Reasoning and probability," *Law, Probability and Risk*, vol. 6, no. 1–4, pp. 43–58, 2007.

[32] M. Burgin and P. Rocchi, "Ample probability in cognition," in *Proceedings of the 2019 IEEE 18th International Conference on Cognitive Informatics & Cognitive Computing (ICCI∗CC)*, pp. 62–65, Milan, Italy, July. 2019.

[33] Y. Biao, "Abnormal event detection based on IPZM," in *Proceedings of the 2011 6th 2011 6th IEEE Joint International Information Technology and Artificial Intelligence Conference*, vol. 1, pp. 198–201, August 2011.

[34] O. Lee, "Probabilistic properties of a nonlinear ARMA process with markov switching," *Communications in Statistics - Theory and Methods*, vol. 34, no. 1, pp. 193–204, 2005.

[35] Y. Guo, "Stock price prediction based on LSTM neural network: the effectiveness of news sentiment analysis," in *Proceedings of the 2020 Stock Price Prediction Based on LSTM Neural Network: the Effectiveness of News Sentiment Analysis (ICEMME)*, pp. 1018–1024, Chongqing, China, November 2020.

[36] C. I. Orozco, M. E. Buemi, and J. J. Berlles, "Towards an attention mechanism LSTM framework for human action recognition in videos," in *Proceedings of the 2020 IEEE Congreso Bienal de Argentina (ARGENCON)*, pp. 1–6, Resistencia, Argentina, December 2020.

[37] L. Ma and P. Cao, "Comparative study of several improved firefly algorithms," in *Proceedings of the 2016 IEEE International Conference on Information and Automation (ICIA)*, pp. 910–914, Ningbo, China, August. 2016.

[38] D. Hamer, "Probability, anti-resilience, and the weight of expectation," *Law, Probability and Risk*, vol. 11, no. 2–3, pp. 135–158, Jun. 2012.