APSIPA Transactions on Signal and Information Processing, 2023, 12, e8

Original Paper

A Real-Time DDoS Attack Detection and Classification System Using Hierarchical Temporal Memory

Yu-Kuen Lai* and Manh-Hung Nguyen

Department of Electrical Engineering, Chung-Yuan Christian University, Chung-Li, Taiwan

ABSTRACT

This paper presents a system implementation to detect and classify different DDoS attacks. The system adopts features of inter-arrival time, entropy, and packet length distribution for a hybrid machine learning model, which is based on the hierarchical temporal memory (HTM) with a k-nearest neighbors (KNN) classifier that can mine network traffic anomalies. Furthermore, it can incrementally learn new traffic behavior and recognize new types of attacks. Finally, system evaluation is conducted based on the CICDDoS 2019 dataset. Thus, the proposed system can successfully identify different attacks with high detection rate, accuracy, and precision.

Keywords: Multiclassification, hierarchical temporal memory, DDoS attack, incremental learning, entropy.

*Corresponding author: Yu-Kuen Lai, ylai@cnsrl.cycu.edu.tw. The work is supported in part by the Ministry of Science and Technology, Taiwan under MOST 111-2221-E-033-032-MY2 and MOST 109-2221-E-033-032-MY2. We thank Mr. Haven K. of ENAGO for his editorial assistance.

Received 27 July 2022; Revised 27 October 2022
ISSN 2048-7703; DOI 10.1561/116.00000147

1 Introduction

In this paper, we propose a new DDoS classification method using entropy, packet length distribution, and inter-arrival time (IAT) related features with hierarchical temporal memory (HTM) algorithm [5]. This new method can detect DDoS attacks in real time, and distinguish between different types of attacks. Moreover, it can update new attack patterns using incremental learning in a one-pass fashion, identify unknown attacks, and work with imperfect data. We developed a better DDoS detector that can solve current problems by detecting and classifying different DDoS attacks. Finding feature sets that detect as many different types of DDoS attacks as possible is challenging. Moreover, detectors must work in high-speed networks with a high detection rate but with low false negatives and positives. Finally, this model should have continuous learning and update capabilities in real time. The proposed method uses HTM and machine learning algorithms to recognize patterns in a data stream. Combined with the HTM modules in multilayers, this method can analyze complex data and build more robust models with higher accuracy, precision, and detection rate. We obtain significant results when classifying different types of DDoS attacks from regular traffic to detect DDoS attacks in our experiments.

The main contribution of this study is summarized as follows:

1. We propose features related to entropy, IAT, and packet length distribution to detect DDoS attacks for each time interval in a real-time network environment. The proposed features can be extracted with FPGA accelerators to process data in a high-speed network.

2. We design a machine learning model based on the HTM algorithm, which is inspired by the human brain, to learn and detect different types of DDoS attacks. The model can process sequence, trending, and distribution data.

3. We evaluate the ability of the model to detect new types or variants of attacks and update the model with incremental learning and one-pass learning.

This paper is organized as follows: Section 2 provides a general background of the HTM and entropy methodologies. Section 3 presents related studies. Section 4 describes the proposed system architecture. Section 5 discusses the experimental details and evaluation results. Finally, Section 6 presents the conclusion and future studies.

2 Background

2.1 Entropy-based Method

Shannon entropy presents the uncertainty or randomness of a distribution. It can effectively monitor network traffic behavior for abnormal detection [13]. Entropy-based features are used along together with machine learning methods to achieve significant results for traffic analysis.

The Shannon entropy is defined as $H = -\sum_{i=1}^{n} p_i log_2 p_i$, where n is the total number of distinct items and p_i is the occurrence rate of item i. In monitoring network traffic, n can be the number of distinct values of a network data field, such as TCP port, IP address, and packet length, whereas p_i is the occurrence rate of a value in a time interval of a data field. For example, we can calculate the entropy of source IP address in a time interval by counting the occurrence numbers of each distinct source IP address, which can be read in IP headers of the packet that belongs to the observed time interval. We then calculate all corresponding occurrence rates (p_i) and apply the Shannon entropy equation.

2.2 Hierarchical Temporal Memory Algorithm

HTM [5] imitates the human brain to learn and recognize patterns. The HTM model uses the simple Hebbian algorithm in the learning phase, which allows it to learn each input data record only once. Therefore, HTM is suitable for processing online data streams. HTM can effectively recognize patterns via an unsupervised method because unlabeled inputs are encoded to binary arrays, which can be quickly compared. HTM models are less affected by noise and can be trained quickly with incremental learning. HTM can map infinite input patterns to finite numbers of sparse distributed representations (SDR). SDRs are binary arrays with two essential characteristics: first, those similar to human memory, can be compared with other SDRs to recognize their similarities, and second, they are highly noise resistant and can be sampled without losing much information. An HTM model, as shown in Figure 1, has three most important parts: semantic encoder, spatial pooler, and temporal memory.

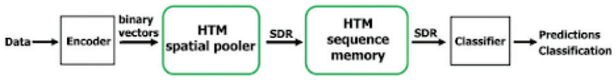

Figure 1: The basic structure of the HTM model consists of semantic encoder, spatial pooler, and temporal memory [5].

Semantic Encoder: This part encodes the feature vectors or different data types to binary vectors on the input space. The minimal element of the input space is called a cell. One type of this encoder is scalar encoder.

Spatial Pooler: This part converts binary vectors on the input space to sparse arrays. The properties of the spatial pooler allow the HTM to maintain sparsity and overlap of the input space. Thus, similar input data have high overlap, and different input data have low overlap.

Temporal Memory: This part is responsible for two important processes: first, it learns from sequences of SDRs created by the spatial pooler over time, and second, it predicts the next incoming pattern based on the temporal context of each input. In temporal memory, each mini-column has numerous cells and each cell represents a different temporal context.

Finally, we can separate HTM applications into two stages [5]. The first stage is the training phase, where the HTM application learns all patterns in the dataset and creates invariant representations (SDRs) before saving them in memory. The second stage is the inference phase, where the HTM application can use that memory to interpret new input patterns and predict the next pattern using continuous learning. The HTM can have all the invariant object representations in its world after full training.

3 Related Studies

3.1 Entropy-based Method

Daneshgadeh *et al.* [6] proposed a method to detect DDoS attacks and distinguish between high-rate DDoS attacks, low-rate DDoS attacks, and flash events. Their study uses Shannon entropy and machine learning algorithms to detect abnormal events using the Mahalanobis distance metric. Furthermore, they used the KOAD algorithm to classify abnormal and normal traffic unsupervised without labeled data.

Koay *et al.* [12] proposed a method that uses entropy-based features andmulticlassifier to detect abnormal traffic events. They experimented using two types of entropy: regular and separation. Separation entropy can provide variation in two distinct entropy-based features. Furthermore, this method can use the rich information of numerous entropy features to improve detection rate and reduce false alarm rate. They proposed the E3ML system, which can utilize rich information of multiple entropy features and three machine learning algorithms recurrent neural network (RNN), multilayer perceptron (MLPs), and alternating decision tree (ADTree), to classify abnormal events.

Ma *et al.* [14] proposed a method to detect DDoS by analyzing the relationship between source and destination IP addresses using chaos theory. This technique collects network traffic and calculates normalized entropy of the source and destination IP addresses. Their model uses the Lyapunov exponent to calculate the separation rate between two related entropy series and defines a threshold separation rate to detect DDoS attacks. Their experiment showed that the separation rate changes significantly when a DDoS attack occurs.

3.2 Machine Learning-based Methods

Machine learning allows computers to learn from data and explore hidden patterns and relationships to predict new data. Supervised machine learning algorithms require labeled data, whereas unsupervised machine learning algorithms can describe the data structure using unlabeled data. Input data for machine learning algorithms are features that should be carefully chosen to improve accuracy and reduce computation time. Feature selection is a necessary phase for analyzing high-dimensional and noisy data.

Implementing machine learning in abnormal detection can provide certain advantages. Machine learning can detect unknown attacks by identifying the relationships between anomalous data and attack patterns or deviation from normal patterns. It can also detect variations of attacks correctly with a low false alarm rate. Furthermore, it can learn all attacks and normal patterns to detect evolving attacks without regular updates. As a result, it can improve the detection speed and accuracy more than signature-based methods.

Thaseen *et al.* [25] used a multi-class support vector machine (SVM) and chi-square feature selection to decrease training and testing time and increase the accuracy of each type of classification. The random forest is more appropriate with a large dataset than an SVM or naive Bayes classifier, which can also adapt to data size. However, it takes a longer time to train, but less time to predict. Furthermore, random forest and decision trees can learn from data features and define rules to separate the dataset into numerous branches. Jalil *et al.* [10] compared the performance of a decision tree, an SVM, and neural network.

Sangkatsanee *et al.* [21] proposed a real-time IDS using a decision tree. Nearest neighbor and logistic regression are popular regression algorithms for finding the most similar training data with the observation. However, they are memory-intensive and may perform poorly with high-dimension data.

Deep learning neural network model is suitable for modeling complex nonlinear relationships by learning multiple levels of data representations that correspond to different levels of abstraction [9]. It can learn complex patterns using high-dimension data. Meng *et al.* [16] compared the performance of LSTM with that of other machine learning algorithms, such as the SVM, when classifying attacks and regular instances in the NSL-KDD dataset. Their results show that LSTM outperformed other methods by 99% detection rate and accuracy.

Singh *et al.* [24] proposed an online sequential extreme learning machine (OS-ELM) to classify different types of attacks in NSL-KDD. The OS-ELM is designed to overcome the high processing time of feed-forward neural networks. Khuphiran *et al.* [11] compared the performance of SVM and deep feed-forward (DFF) networks. They argued that SVM can deliver faster classification time, whereas DFF is more appropriate for high-accuracy detectors.

3.3 Survey of Features for DDoS Detection

In our survey, researchers leveraged source information extracted from packet headers to create new features to describe the nature of attacks. They used different feature sets to detect DDoS attacks. Some features are easy to extract from packet headers, whereas others are complicated to calculate in real time.

Khuphiran *et al.* [11] proposed two feature sets: window-based and packet-based. These features are used to detect DDoS attacks in the 2009 DARPA Intrusion Detection dataset. Qin *et al.* [19] proposed a method using entropy-based features to model normal patterns using a clustering algorithm. This technique calculates five entropy-based features, including source IP, destination IP, destination port, flow duration, and packet size. The entropy of packet size uses five different size levels for a high-speed network accordingly.

Daneshgadeh *et al.* [6] proposed a hybrid method to distinguish between normal traffic, DDoS, and flash event. The authors used two types of feature vectors. The first vector consists of the time interval, destination IP entropy, and source IP entropy for the online machine learning-based method. The second vector consists of the time interval.

Balkanli *et al.* [2] proposed two feature sets to detect DDoS attacks in backscatter darknet traffic. The paper proved that their method can detect DDoS attacks without features related to IP addresses and port numbers. Koay *et al.* [12] proposed a method for classifying normal and attack traffic in a dataset of different DDoS attacks. The method uses fifteen regular entropy-based features and five entropy variation features. The latter is based on the variation of two distinct common entropy-based features generated using the variation of the Lyapunov exponent separation method [14]. They claimed that their method can effectively detect DDoS attacks across datasets with different intensities.

In our survey, we noticed that entropy-based features are the most common, which were used by Qin *et al.* [19], Daneshgadeh *et al.* [6], Mao *et al.* [15], and Koay *et al.* [12] to detect different types of DDoS attacks in the DARPA or CAIDA dataset. Additionally, entropy is a compact form to describe the distribution of the changing feature, which is very important in network anomaly detection.

4 System Architecture

4.1 Recognize Pattern with HTM Cortical Column

1. *Create SDRs from sequences of input data using the HTM algorithm*: Each input data may have one or combined numerous features. The scalar encoder is used to convert input data into binary vectors. These binary vectors are sent to the spatial pooler module to create SP-SDR and then continuously sent to the temporal memory module to create TM-SDR. TM-SDR are outputs of the

temporal memory module. They are SDR binary arrays used as patterns to present a sequence of input data at a time interval. Each pattern represents the corresponding input data and its context. The final step is using classification techniques to label prototype patterns in the training and prediction phases.

2. *Find prototype patterns from SDRs in the training phase*: Here, we will use a clustering technique to find the prototype patterns from those created from the training datasets. Prototype patterns are SDR binary arrays that can affect the result of the KNN classifier, which is used to assign labels for observed patterns.

Two methods are used to calculate the distance between two SDRs: Hamming and overlap. Ahmad *et al.* [1] preferred to use the latter over the former. For Hamming distance method, we count the number of different bits (zero and one) between two SDRs. The lower the number of different bits, the shorter the distance and the more similar they are. For the overlap distance method, we count the number of overlap bits (one) between two binary SDRs, the higher the number of overlap bit (one), the shorter the distance and more similar they are.

We define the distance threshold as the minimum distance between two prototype patterns. It is an optimized parameter of the HTM-KNN model. A lower distance threshold creates more prototype patterns in the training phase, whereas higher distance threshold creates fewer prototype patterns. An optimized distance threshold is required to create enough prototype patterns, and each prototype pattern is present for a variant of a type of attack. When the HTM calculates a training pattern from input data in the training phase, the model finds the smallest distance between the new and all existing prototype patterns. If the smallest distance is higher than the distance threshold, the training pattern will be assigned as a new prototype pattern. If the smallest distance is lower than the distance threshold, the new pattern will be assigned as an absorbed pattern. We will provide labels for all prototype patterns similar to those in the corresponding input data.

3. *KNN assigns labels for observed patterns in the prediction phase*: KNN algorithm is used to assign the label for observed patterns in the prediction phase. To assign a label for input data, the model must convert the observed input data to an SDR binary array (observed pattern). Then, the model can compare the distance between the observed pattern and all existing prototype patterns specified in the training phase to find k-nearest prototype patterns. Finally, the observed pattern is assigned a label by major voting between its k-nearest prototype patterns.

4.2 Two-layered HTM-KNN Model

Using the HTM cortical column described in Section4.1, we can recognize the most similar attack signature based on the increasing and decreasing tendency of the features. We built a two-layered HTM model (Figure 2) to observe all network features and recognize attack signatures to assign labels to the observed patterns in the network traffic.

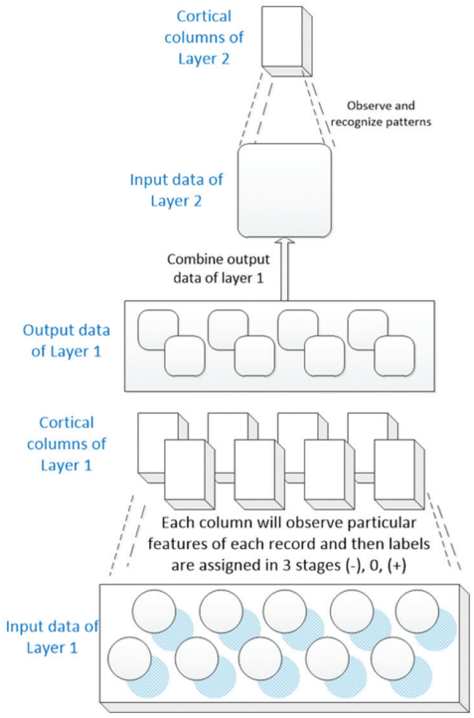
Cortical columns of Layer 2
Observe and recognize patterns
Input data of Layer 2
Combine output data of layer 1
Output data of Layer 1
Cortical columns of Layer 1
Each column will observe particular features of each record and then labels are assigned in 3 stages (-), 0, (+)
Input data of Layer 1

Figure 2: Block diagrams of the proposed HTM-KNN model in two layers.

Layer one is responsible for observing and assigning labels for all features. Each HTM cortical column observes a particular feature using only three types of labels: $(-)$, (0), and $(+)$. Label (0) indicates that the attack makes the feature value similar to that of regular traffic. Labels $(-)$ and $(+)$ represent the decreasing and increasing tendency, respectively, of the feature value. The input data of layer one are a series of records. Each record represents each time interval with values of a feature set.

Layer two is responsible for recognizing the matching attack signatures or most similar attack signatures and then predicting the type of attack. All output data from layer one are combined to become input data for layer two. In layer two, the HTM cortical column converts the input data to SDR binary and compares the observed SDR with the most similar signatures. On the basis of this , we can predict the type of attack on regular traffic. Furthermore, the two-layers HTM model can also remember the signatures of different types of attacks in the training phase and then recognize the attack signatures in the prediction phase.

4.3 Features Extraction

We extracted entropy features from DDoS datasets and observed entropy-based sequences of the network traffic. We noticed that entropy features can be strongly affected by DDoS attacks, and distinguished between different types of attacks. For example, when we viewed the entropy of source IP, destination IP, source port, destination port, and packet length in the CICDDoS 2019 training dataset, we discovered that entropy values changed as DDoS attacks occurred. Additionally, other proposed features were discovered using the distribution of packet size and mean packet size instead of using the entropy of packet size. Because calculating each distinct packet size in a high-speed network is difficult, we adopted the packet size in eight different levels. Finally, we proposed a set of feature vectors as the input data. They consist of eight features: entropy of TCP source, destination ports, packet length, the average packet length, total packet count, and the distinct number of TCP source and destination ports.

Another idea is the packet length distribution in a range of IAT. We extracted features to observe how attack packets affect the packet length distribution and calculated the IAT for each incoming packet. Next, we filtered all packets with IAT lower than five microseconds (5 ms IAT packets). Then, we created the packet length distribution for each time interval with those filtered 5 ms IAT packets. When DDoS attacks occur, attackers attempt to send numerous abnormal packets to the network and possibly, thus causing the number of 5 ms IAT packets to increase. We observed how the distribution of packet length changes in a time interval to recognize different types of attacks. We divide the different packet lengths into fifteen groups called Bins, and each Bin can be used as a feature of the proposed machine learning model. The feature set includes fifteen features. Each of them observed the number of packets in the 100 bytes. We recognize that different types of DDoS attacks affect the distribution of packet length in different ways. Therefore, we believe that our selected features can be used to discriminate against many DDoS attacks in the CICDDoS 2019 dataset [22].

4.4 Signature of DDoS Attacks

The proposed method classifies and recognizes the DDoS attack signatures in an observation time interval of 15 s. After extracting and analyzing features from the CICDDoS 2019 dataset, we observed that different types of DDoS attacks cause various changes in the distributions of IP, port, and packet size. We used entropy-based features, numbers of distinct items in distribution, average packet length, and packet count to record the changes of network characteristics during an attack. We can identify the signatures of different attacks using this method by identifying which types of attacks have identical signatures.

Table 1 shows how each feature changes when a DDoS attack occurs in the CICDDoS 2019 dataset. The (0) symbol means the attack has a feature value similar to regular traffic. The (−) symbol indicates that the attack causes the

Table 1: The trend for the features observed during the attacking phase in the CICDDoS 2019 dataset.

	IP Entropy		IP Distinct		Port Entropy		Port Distinct						
Attack type/ Feature	Src IP Entropy	Dst IP Entropy	Src IP Distinct	Des IP Distinct	Src Port Entropy	Dst Port Entropy	Src Port Distinct	Dst Port Distinct	Pkt Len Entropy	Ave-rage Pkt Len	Pkt Count	Protocol Entropy	Signature #
NTP (12)	−	−	0	0	+	+	0	+	−	−	+	−	1
DNS (1)	−	−	0	0	−	+	0	0	0	0	0	+	2
LDAP (2)	−	−	0	0	+	+	0	+	+	+	+	+	3
MSSQL (3)	−	−	0	0	+	+	+	+	+	0	+	+	4
NetBIOS (4)	−	−	0	0	+	+	0	+	−	−	+	+	5
SNMP (5)	−−	−−	0	0	+	+	0	+	+	+	+	+	3
SSDP (6)	−	−	0	0	+	+	+	+	+	−	+	+	7-1
UDP (7)	−−	−−	0	0	+	+	+	+	+	−	+	+	7-1
UDP (7)	−−	−−	0	0	+	+	+	+	0	−	+	+	7-2
UDP-Lag (8)	0	0	0	0	0	0	0	0	0	0	0	0	0
WebDDoS (9)	0	0	0	0	0	0	0	0	0	0	0	0	0
SYN (10)	−	−	0	0	+	+	+	+	−	−	+	−	8
TFTP (11)	−	−	0	0	+	+	+	+	−	−	+	+	9
Portmap (13)	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2: The attack code of different types of DDoS attacks.

Types of attack	DNS	LDAP	MS-SQL	Net-BIOS	SNMP	SSDP	UDP	UDP-Lag	Web-DDoS	SYN	TFTP	NTP	Port-map
Attack Code	1	2	3	4	5	6	7	8	9	10	11	12	13

feature value to decrease. The (+) symbol means that the attack causes the feature value to increase. Furthermore, the (−−) symbol) represents further decrease than that of (−), and similarly, the (++) symbol means further increase than that l of (+). Thus, as shown in Table 1, we can easily identify the different types of attacks based on the changing trend of features. For example, the protocol entropy feature differs when comparing SYN (10) and TFTP (11) attacks. SYN (10) decreases the feature value, whereas TFPT (11) increases. We can distinguish between nine types of DDoS attacks using the feature set. However, we can not distinguish between the attacks of SSDP (6) and UDP (7). Furthermore, we can not differentiate between UDP-Lag (8) and WebDDoS (9) from the regular traffic.

For more features, Figure 3 shows the distribution of the packet length of the CICDDoS 2019 dataset, and all packets have an IAT lower than 5 ms. The distribution has 15 bins representing packet lengths from 0 to 1,500 bytes, and each bin represents packet lengths in the 100-byte range. The number of packets is counted for each bin of each time interval. Figure 3 shows how DDoS packets change packet length distribution in a time interval of 15 ms.

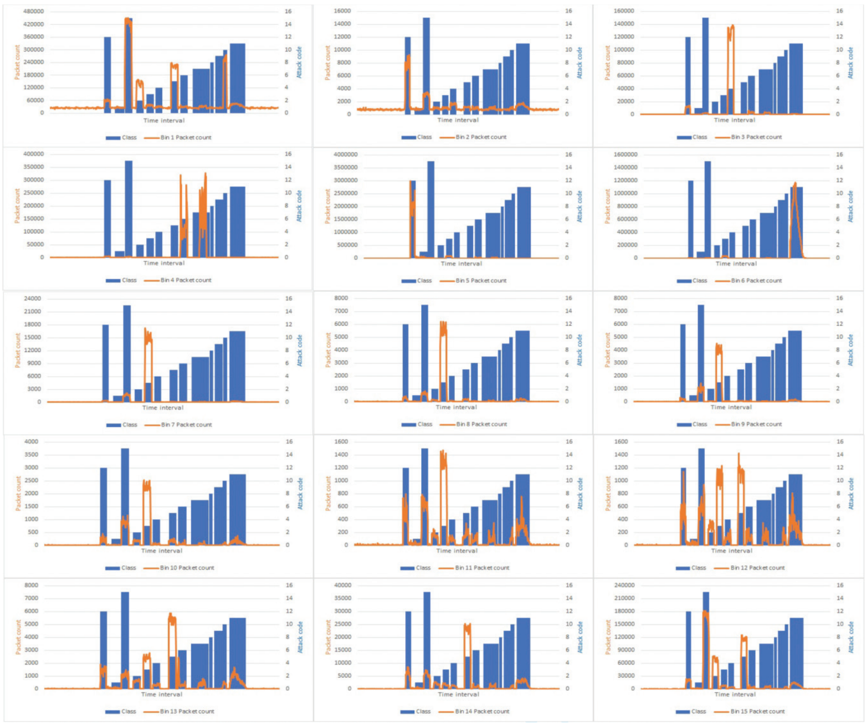

Figure 3: Fifteen features of packet length distribution with IAT lower than 5 ms of the training dataset.

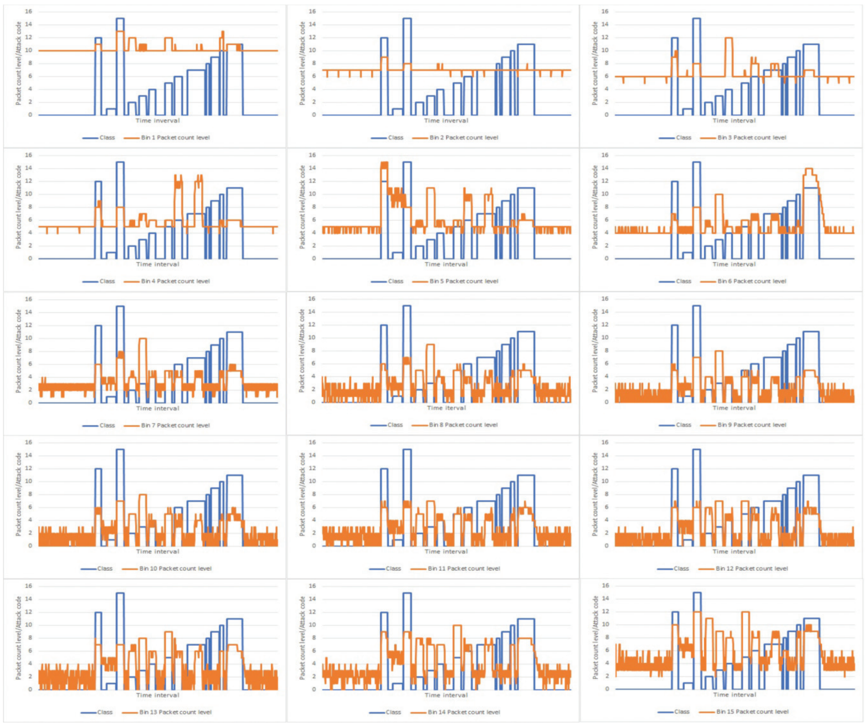

Figure 4: Fifteen features of packet length distribution with IAT lower than 5 ms in the training dataset. The packet-count level is calculated by natural logarithm with rounding.

In the figures, the attack codes listed in Table 2, represent the attack type of each time interval. For example, we observed that Bin 1 packet count, which counts the number of packets, has packet lengths from 0 to 100 bytes. Furthermore, we notice that when SYN DDoS (Attack code 10) occurs, the packet count increases from 240,000 to 270,000 packets in each time interval of 15 s. However, NTP DDoS (Attack code 12) increases to approximately 60,000 packets, LDAP DDoS (Attack code 2) increases from approximately 120,000 to 150,000 packets, and SNMP DDoS (Attack code 5) increases from approximately 210,000 to 240,000 packets. These types of differences also occur with other bins. Therefore, a machine learning model can combine and learn different types of information to distinguish different types of attacks as much as possible. Because the numbers of packets are big and vary extensively, we use logarithm and rounding to convert the numbers of packets to packet count level. Figure 4 shows fifteen features of packet length distribution for 5 ms IAT packets in packet-count level, which can be used as a feature set for the proposed machine learning model.

5 Experiment and Evaluation

5.1 Simulation and Testing Environment

Our experiment uses the merged CICDDoS 2019 and MAWI datasets [8]. CICDDoS 2019 provides PCAP files for benign traffic and the most updated common DDoS attacks. The attack flows were labeled using a timestamp. CICDDoS 2019 has one training and one testing dataset. The training dataset has twelve DDoS attack types including NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN, and TFTP. The testing dataset has seven DDoS attack types: PortScan, NetBIOS, LDAP, MSSQL, UDP, UDP-Lag, and SYN. In Figures 5 and 6, we recognized that the packet count in the normal time intervals is smaller than that of attack patterns. This indicates that the normal traffic of the CICDDoS 2019 dataset occurs when the network has almost no traffic, which is unrealistic. Therefore, the MAWI070201 dataset was adopted to served as the background traffic. Attack vectors of the CICDDoS 2019 dataset were added to background traffic to create DDoS events. The traffic traces of the MAWI dataset were collected

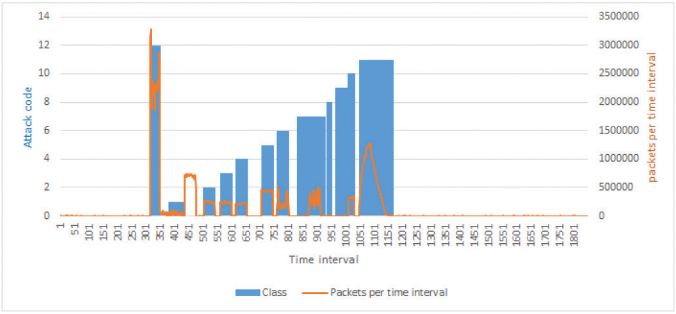

Figure 5: Attack class and volume in the CIC DDoS 2019 training dataset.

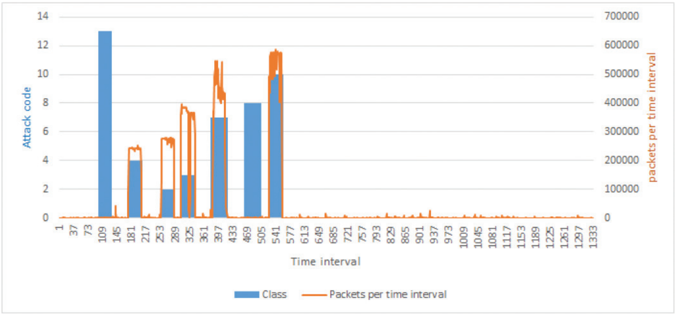

Figure 6: Attack class and volume in the CICDDoS 2019 testing dataset.

in a real backbone network. We used the 200702011400 trace of the MAWI dataset, which has a total time of 900 s and an average network throughput of 123 Mbps. The background traffic has approximately 276,882 distinct IPv4 addresses and 553,421 distinct flows. The 900 s trace file was merged with each 900 s time interval of the CICDDoS 2019 dataset PCAP files.

We obtained the PCAP files of the training and testing datasets to extract the features. We used two feature sets in the experiment. The first feature set has eight features: entropy of TCP source and destination port, entropy of packet length, distinct number of TCP source and destination port, average packet count, and total packet count. The second feature set has 15 features representing the packet length distribution for 5 ms IAT packets.

We created feature vectors for each 15 s time interval of the two datasets. Figure 5 shows the time intervals in DDoS attacks and the total packet of each time interval of the CICDDoS 2019 training dataset. Noticeably, LDAP, MSSQL, NetBIOS, SNMP, SSDP, and SYN DDoS attacks appeared with high volume traffic. However, UDP-Lag and Web DDoS appeared with low volume traffic, UDP attacks appeared with both low and high volume traffic. The proposed models attempted to detect LDAP, MSSQL, NetBIOS, UDP, UDP Lag, and SYN DDoS attacks in the testing dataset. Figure 6 shows that the attack volume of LDAP, MSSQL, NetBIOS, UDP, UDP Lag, and SYN DDoS attacks in the CICDDoS 2019 testing dataset is similar to that of the same type of attack in the training dataset. Table 2 shows the attack code used in Figure 5 and 6. Furthermore, the attack code replaces the name of the DDoS attack.

In the first phase simulation and experiments, we trained two HTM-KNN models with the training dataset and evaluated two models using the testing dataset. In the second phase, we updated two models using the testing dataset with incremental learning and one-pass learning. We then evaluated the two models using the testing dataset. The two models were changed to a learning mode to learn new patterns without retraining the whole model while updating. The models can still remember patterns they learned previously. New input patterns were connected to become a data stream, then each input pattern was then observed and learned once by the HTM models. The spatial pooler and the temporal memory of HTM can learn and create the corresponding SDR represented for each input pattern by reading the input exactly once.

5.2 Results and Evaluation

We used two HTM-KNN models to detect attack time intervals in the datasets. The HTM-KNN-1 model learns feature trends shown in Table 1. The HTM-KNN-2 model learns fifteen features of packet length distribution for packets with IAT lower than 5 ms shown in Figure 4. The HTM-KNN-1 model is the two-layered model shown in Figure 2. The HTM-KNN-2 model is a one-layered model with only one cortical column. The models learn all input feature vectors in the training phase, create corresponding SDRs, choose prototype patterns, and save all prototype patterns in memory. Prototype patterns represent normal behavior and all twelve types of DDoS attacks: NTP, DNS, LDAP, MSSQL, Net-BIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN, and TFTP.

These models convert the input data sequence in each observation time interval to the corresponding SDRs in the testing phase. Then, the KNN algorithm assigns labels for each observed SDR by analyzing the distance between the corresponding SDR and all prototype patterns detected.

Figure 2 shows the HTM-KNN-1 model, there are eight HTM cortical columns in layer one used to learn the trend of features shown in Table 1. Therefore, for all observed features, labels of $(-)$, (0), and $(+)$ are assigned to the eight cortical columns for each time interval. Layer one signals the observed signature of the current time interval to layer two. Then, layer two continues to create SDRs representing the attack signatures from layer one and compares the observed SDR with all learned attack signatures in the training phase. Finally, the matching or most similar attack signatures can be identified using distance methods. The most similar means the closest distance, wheres matching means the zero distance.

The testing dataset comprises seven different types of DDoS attacks. The HTM-KNN-1 and HTM-KNN-2 models learned six types of attack patterns, such as NetBIOS, LDAP, MSSQL, UDP, UDP-Lag, and SYN DDoS attacks in the training phase, and the models attempt to detect those types of attacks in the testing dataset.

In the first phase, we evaluated the pattern recognition and classification ability of the HTM models using two types of feature sets such as trend features (HTM-KNN-1) and distribution features (HTM-KNN-2), and three types of distance methods such as Hamming-matching, Hamming-most similar, and overlap. Table 3 presents the performance of HTM-KNN-1 and HTM-KNN-2 by detection rate, accuracy and precision metrics. Additionally, we evaluated the performance of each model using three types of distance methods, such as Hamming-matching, Hamming-most similar and overlap, as explained in Section 4.1.

Table 3 shows that the HTM-KNN-1 model achieves high detection rate, accuracy, and precision for different types of DDoS attacks. When we used the Hamming-matching g method as the evaluated result of the model, the results show that the model can detect LDAP-SNMP, MSSQL, NetBIOS, UDP-SSDP, and SYN DDoS attacks. Note that the HTM-KNN-1 model can not distinguish between LDAP and SNMP; this is similar for UDP and SSDP to the selected features. In our experiment, the true and false-positives of UDP-Lag are zero. Thus, both detection rate and precision are zero. For the attack signatures in Table 1, UDP-Lag has the same attack signature as a normal traffic pattern, so the HTM-KNN-1 model cannot detect any UDP-Lag pattern. To detect

Table 3: The performance of two HTM-KNN models. HTM-KNN-1 model cannot discriminate between LDAP and SNMP as well as between UDP and SSDP. The HTM-KNN-2 model can discriminate between LDAP and SNMP as well as between UDP and SSDP.

		HTM-KNN 1			**HTM-KNN 2**		
Metric	**Attack types**	Hamming-Matching	Hamming-Most Similar	Overlap	Hamming-Matching	Hamming-Most Similar	Overlap
Detection rate (%)	LDAP*	91	96	96	0	100	100
	MSSQL	77	86	86	0	91	91
	NetBIOS	87	93	93	0	96	96
	UDP*	80	80	80	0	36	36
	UDP-Lag	0	0	0	0	0	0
	SYN	90	93	93	0	78	78
Accuracy (%)	LDAP	99	99	99	97	100	100
	MSSQL	99	99	99	97	99	99
	NetBIOS	99	99	99	97	99	99
	UDP	99	99	99	96	96	95
	UDP-Lag	96	96	96	96	96	96
	SYN	99	99	99	97	99	99
Precision (%)	LDAP	100	100	100	0	100	100
	MSSQL	100	88	88	0	100	100
	NetBIOS	100	100	100	0	96	96
	UDP	89	89	89	0	38	34
	UDP-Lag	0	0	0	0	0	0
	SYN	100	100	100	0	100	100

UDP-Lag DDoS attacks and more types of attacks, we must add new features to the feature set in future experiments. These new features should show changes when attacks appear.

Table 4 presents the confusion matrix of the HTM-KNN-1 model using Hamming-matching distance. Furthermore, there are some false negatives in the time intervals of LDAP-SNMP, MSSQL, NetBIOS, UDP-SSDP, and SYN DDoS attacks. The false positives are due to the misses of the matching signature of observed attack patterns. Thus,, the model labeled as no matching. Rather than assigning labels to the no-matching patterns, we can adapt the strategy by referring to the most similar attack signatures, which can be specified on the basis of the distance between SDRs of the closest attack signatures and the observed patterns. In the Hamming-matching signature, if an observed attack pattern matches a prototype pattern (distance is zero), it will be labeled the same as the prototype pattern. The observed attack patterns will be marked as "no matching", if the model cannot find any matching prototype pattern. Using the Hamming-most similar and overlap methods, the model will assign the label of the closest prototype pattern to the observed attack pattern. Table 5 shows the confusion matrix of the testing dataset using the HTM-KNN-1 model with the Hamming-most similar distance method; the model always finds a label with the most similar pattern to assign to each observed pattern.

Table 3 shows that the HTM-KNN-2 model can achieve a better detection rate of LDAP, MSSQL, and NetBIOS than the HTM-KNN-1 model. However, when the model uses Hamming-matching signature as the distance method, the model can not detect any attack. This is because all attack patterns of the training dataset learned by the HTM-KNN-2 model in the training phase, do not reappear in the testing dataset. Thus, the HTM-KNN-2 model still works with the Hamming-most similar and overlap methods to find the most similar prototype patterns for each observed pattern. We attempted to distinguish between UDP and SSDP using the HTM-KNN-2 model, which is impossible using the HTM-KNN-1 model. The result shows that the detection rate for UDP is 36% for both Hamming-most similar and overlap distance methods. Table 6 shows the confusion matrix of the HTM-KNN2, where the HTM-KNN-2 model detected 15 UDP attack time intervals, and misclassified 18 UDP attack time intervals as SSDP attack. It also has 27 normal traffic time intervals misclassified by the UDP attack. This proves that some UDP attack prototype patterns in the learning phase of the HTM-KNN-2 model are similar to normal traffic prototype patterns. However, the HTM-KNN-2 model can not detect UDP-Lag attacks; it misclassified 40 UDP-Lag attack time intervals as normal traffic. We attempted to keep detecting UDP-Lag attacks by updating the models in the next phase.

In the second phase, we evaluate the continuous pattern-updating ability of the HTM models using incremental learning and one-pass learning. We

Table 4: Confusion matrix of the testing dataset using the HTM-KNN-1 model (Hamming-Matching).

	Predicted Class											
Actual Class	NTP	DNS	LDAP-SNMP	MS-SQL	Net-BIOS	UDP-SSDP	UDP-Lag	Web-DDoS	SYN	TFTP	Normal	No Matching
NTP	**0**	0	0	0	0	0	0	0	0	0	0	0
DNS	0	**0**	0	0	0	0	0	0	0	0	0	0
LDAP-SNMP	0	0	**30**	0	0	0	0	0	0	0	0	3
MSSQL	0	0	0	**28**	0	4	0	0	0	0	2	2
NetBIOS	0	0	0	0	**29**	0	0	0	0	0	0	4
UDP-SSDP	0	0	0	0	0	**33**	0	0	0	0	1	7
UDP-Lag	0	0	0	0	0	0	**0**	0	0	0	27	14
Web-DDoS	0	0	0	0	0	0	0	**0**	0	0	0	0
SYN	0	0	0	0	0	0	0	0	**30**	1	0	2
TFTP	0	0	0	0	0	0	0	0	0	**0**	0	0
Normal	0	0	0	0	0	0	0	0	0	0	**761**	321

Table 5: Confusion matrix of the testing dataset using the HTM-KNN-1 model (Hamming-Most Similar).

	Predicted Class										
Actual Class	NTP	DNS	LDAP-SNMP	MS-SQL	Net-BIOS	UDP-SSDP	UDP-Lag	Web-DDoS	SYN	TFTP	Normal
NTP	**0**	0	0	0	0	0	0	0	0	0	0
DNS	0	**0**	0	0	0	0	0	0	0	0	0
LDAP-SNMP	0	0	**32**	1	0	0	0	0	0	0	0
MSSQL	0	0	0	**31**	0	3	0	0	0	0	2
NetBIOS	0	1	0	0	**31**	0	0	0	0	0	1
UDP-SSDP	0	0	0	2	0	**33**	0	0	0	0	6
UDP-Lag	0	0	0	0	0	0	0	0	0	0	41
Web-DDoS	0	0	0	0	0	0	0	**0**	0	0	0
SYN	0	0	0	0	0	1	0	0	**31**	1	0
TFTP	0	0	0	0	0	0	0	0	0	**0**	0
Normal	0	0	0	1	0	0	0	0	0	0	**1081**

continuously updated the HTM-KNN models using new patterns of the testing dataset without retraining the models. The HTM can support incremental and one-pass learning, so it can learn new patterns by reading each input exactly once, and then updating new prototype patterns in memory. One-pass learning does not require memory to store the previously learned input data and does not need to learn the same input repeatedly. Thus, it can save numerous computing resources, and it is suitable for processing data streams. Table 8 shows the performance of the model after an update. The HTM-KNN-2 model achieves perfect detection rate, accuracy, and precision for the LDAP, MSSQL, NetBIOS, UDP, and SYN DDoS attacks. This proves that the model learned all new patterns after updating via incremental learning. Furthermore, the HTM-KNN-2 model can detect UDP-Lag with a 68% detection rate. Table 7 presents the confusion matrix of the HTM-KNN-2 model after an incremental learning update. It has thirteen UDP-Lag time intervals misclassified as normal traffic and four normal time intervals, which are misclassified as UDP-Lag time intervals. This proves that the UDP-Lag attack and normal traffic are very similar with HTM-KNN-2 model. However, the HTM-KNN-1 model showed no significant improvement after the incremental update. This is because the trend of features observed during the attacking phase in the CICDDoS 2019 dataset (Table 1) is the same for both training and testing datasets. Thus, there is no new information updated in the HTM-KNN-1 model.

Table 9 shows the comparison between the proposed method and that of the previous study [17] and other studies that also used the CICDDoS 2019. There are two types of models: multi-classify and binary-classify. The former can discriminate between different types of attacks and normal traffic, whereas the latter one only discriminates between normal traffic and attack. The detected objects include time window (time interval) and flow. The first three models classify DDoS attacks for each time window (window-based model), and other models classify DDoS attacks for each flow (flow-based model). Compared with the first three models, the proposed model can achieve better detection rate, accuracy, and precision. Furthermore, the proposed model can also distinguish between different types of attacks and detect UDP-Lag with a 68% detection rate. The proposed model can support the other flow-based models to improve their performance by combining their results. First, the DDoS attack flows can be rechecked to determine whether they occur in DDoS attack time windows, which is used to reduce false alarms. Second, some flow-based models can not discriminate between several types of attacks; thus, combining the result with window-based models can conduct deeper classifications. For example, Chartuni *et al.* [4] cannot distinguish between some types of DDoS flows including DNS and LDAP, NetBIOS and Portmap, and SSDP and UDP. We can detect the types of time windows, to which the flows belong for more detailed distinctions.

The results from all experiments show that the proposed HTM-KNN models can distinguish between different types of DDoS attacks and normal

Table 6: Confusion matrix of the testing dataset using the HTM-KNN-2 model (Overlap).

	Predicted Class												
Actual Class	NTP	DNS	LDAP	MS-SQL	Net-BIOS	SNMP	SSDP	UDP	UDP-Lag	Web-DDoS	SYN	TFTP	Normal
NTP	**0**	0	0	0	0	0	0	0	0	0	0	0	0
DNS	0	**0**	0	0	0	0	0	0	0	0	0	0	0
LDAP	0	0	**33**	0	0	0	0	0	0	0	0	0	0
MSSQL	0	1	0	**33**	0	0	0	0	0	0	0	0	2
NetBIOS	1	0	0	0	**32**	0	0	0	0	0	0	0	0
SNMP	0	0	0	0	0	**0**	0	0	0	0	0	0	0
SSDP	0	0	0	0	0	0	**0**	0	0	0	0	0	0
UDP	0	2	0	0	0	0	18	**15**	0	0	0	0	6
UDP-Lag	0	0	0	0	0	0	0	1	**0**	0	0	0	40
Web-DDoS	0	0	0	0	0	0	0	0	0	**0**	0	0	0
SYN	0	0	0	0	0	0	0	0	0	0	**26**	6	1
TFTP	0	0	0	0	0	0	0	0	0	0	0	**0**	0
Normal	0	0	0	0	1	0	3	27	2	32	0	0	**1017**

Table 7: Confusion matrix of the testing dataset using the HTM-KNN-2 model (overlap) after updating via incremental learning.

	Predicted Class												
Actual Class	NTP	DNS	LDAP	MS-SQL	Net-BIOS	SNMP	SSDP	UDP	UDP-Lag	Web-DDoS	SYN	TFTP	Normal
NTP	**0**	0	0	0	0	0	0	0	0	0	0	0	0
DNS	0	**0**	0	0	0	0	0	0	0	0	0	0	0
LDAP	0	0	**33**	0	0	0	0	0	0	0	0	0	0
MSSQL	0	0	0	**36**	0	0	0	0	0	0	0	0	0
NetBIOS	0	0	0	0	**33**	0	0	0	0	0	0	0	0
SNMP	0	0	0	0	0	**0**	0	0	0	0	0	0	0
SSDP	0	0	0	0	0	0	**0**	0	0	0	0	0	0
UDP	0	0	0	0	0	0	0	**41**	0	0	0	0	0
UDP-Lag	0	0	0	0	0	0	0	0	**28**	0	0	0	13
Web-DDoS	0	0	0	0	0	0	0	0	0	**0**	0	0	0
SYN	0	0	0	0	0	0	0	0	0	0	**33**	0	0
TFTP	0	0	0	0	0	0	0	0	0	0	0	**0**	0
Normal	0	0	0	0	0	0	0	0	4	0	0	0	**1078**

Table 8: The performance of the HTM-KNN model (Overlap) after updating via incremental learning.

Metric	Attack types	HTM-KNN-1 Overlap	HTM-KNN-2 Overlap
Detection rate (%)	LDAP	96	100
	MSSQL	86	100
	NetBIOS	93	100
	UDP	80	100
	UDP-Lag	0	68
	SYN	93	100
Accuracy (%)	LDAP	99	100
	MSSQL	99	100
	NetBIOS	99	100
	UDP	99	100
	UDP-Lag	96	98
	SYN	99	100
Precision (%)	LDAP	100	100
	MSSQL	86	100
	NetBIOS	100	100
	UDP	89	100
	UDP-Lag	0	84
	SYN	100	100

time intervals. The Hamming-matching signature method can recognize all prototype patterns learned in the learning phase. The Hamming-most similar and overlap method attempt to identify the most similar prototype patterns for observed patterns. In our experiments, the Hamming-matching signature method has a lower detection rate than the Hamming-most similar method and overlap method. The Hamming-most similar method and overlap method achieved high performance. The latter is preferred for finding the most similar SDR over the former [1]. However, the two distance methods achieved the same performance in our experiments.

6 Conclusions and Future Works

This study proposed methods for classifying different types of DDoS attacks using entropy-based features, packet length distribution, and IAT-related features on selected packet headers. This system was constructed based on the hierarchical temporal memory (HTM) and k-nearest neighbors algorithm. The proposed methods adapted the Shannon entropy as an essential indicator to

Table 9: Comparison between the proposed method (HTM-KNN-2) with that of other studies using the CICDDoS 2019 dataset.

Study	**Machine learning algorithm**	**Type of model**	**Detected object**	**Detection rate (%)**							**Accu-racy (%)**	**Pre-cision (%)**
				LDAP	MS-SQL	Net-BIOS	UDP	UDP-Lag	SYN	Ave-rage	Ave-rage	Ave-rage
Proposed Method	HTM, KNN	Multi-Classify	Time window	100	100	100	100	68	100	94.67	99.66	97.33
Nguyen *et al.* [17]	HTM, KNN	Multi-Classify	Time window	91	87	77	80	0	90	70.83	98.5	81.5
Novaes *et al.* [18]	LSTM-FUZZY	Binary- Classify	Time window	-	-	-	-	-	-	93.13	97.89	-
Chartuni *et al.* [4]	Neural Network	Multi-Classify	Flow	78	91	98	95	97	100	94.03	94.21	94.21
Shieh *et al.* [23]	BI-LSTM, GMM	Binary- Classify	Flow	-	-	-	-	-	-	96.9	99.75	96.65
Rajagopal *et al.* [20]	Neural Network	Multi-Classify	Flow	73	38	93	72	87	99	77	-	77.83
Ferrag *et al.* [7]	DNN, RNN, CNN	Multi-Classify	Flow	98	96	99	71	0	100	77	-	-
Can *et al.* [3]	MLP	Multi-Classify	Flow	97	52.5	92.9	99.9	13.9	100	76	-	91.16

detect DDoS attacks in real time. The HTM allowed the model to remember all prototype patterns of different attack types and assign labels for input patterns using other machine learning algorithms, such as KNN. The models can also implement incremental and one-pass learning by updating prototype patterns only once without retraining the entire model. The experiment was conducted using the merged CICDDoS 2019 and MAWI dataset. The simulation results showed that the proposed models achieved high performance when classifying several types of DDoS attacks in the dataset.

Compared with the previous method [17], we proposed a new HTM-KNN model with additional feature sets related to the distribution of packet length and IAT. We compared the performance of different distance methods including Hamming-matching, Hamming-most similar, and overlap, which were used to compare sparse distributed representations (SDRs). We also distinguished between UDP and SSDP attacks, LDAP and SNMP attacks, as well as UDP-Lag attacks and normal traffic with the new feature set. As a result, we achieved good performance when implementing the proposed models with incremental learning and one-pass learning. This is a substantial advantage obtained from the HTM compared with deep learning algorithms. The proposed approach is appropriate for machine learning-based real-time applications that must process data in a streaming fashion and maintain continuous update capability. In addition, the approach uses fewer computing resources to achieve multi-classification of objects in different contexts, which is suitable for some applications such as DDoS and anomaly detection.

For future studies, we plan to update the model by adding more features to detect more attacks based on various network traffic traces and implement the model on physical switches to detect DDoS in real time. Additionally, we will devise a method to combine the results of different HTM models for unified results achieving higher performance.

References

[1] S. Ahmad and J. Hawkins, "Properties of Sparse Distributed Representations and their Application to Hierarchical Temporal Memory," 18, 2015.

[2] 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)*, 2014 IEEE Symposium on Computational Intelligence in Cyber Security (CICS), December 2014, 1–8.

[3] D.-C. Can, H.-Q. Le, and Q. Ha, "Detection of Distributed Denial of Service Attacks Using Automatic Feature Selection with Enhancement for Imbalance Dataset," April 5, 2021, 386–98.

[4] A. Chartuni and J. Márquez, "Multi-Classifier of DDoS Attacks in Computer Networks Built on Neural Networks," *Applied Sciences*, 11(22), 2021, 10609, https://www.mdpi.com/2076-3417/11/22/10609 (accessed on 01/01/2022).

[5] Y. Cui, S. Ahmad, and J. Hawkins, "The HTM Spatial Pooler - A Neocortical Algorithm for Online Sparse Distributed Coding," *Frontiers in Computational Neuroscience*, 11, 2017, 111.

[6] S. Daneshgadeh, T. Ahmed, T. Kemmerich, and N. Baykal, "Detection of DDoS Attacks and Flash Events Using Shannon Entropy, KOAD and Mahalanobis Distance," in *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, February 2019, 222–9.

[7] M. A. Ferrag, L. Shu, H. Djallel, and K.-K. R. Choo, "Deep Learning-Based Intrusion Detection for Distributed Denial of Service Attack in Agriculture 4.0," *Electronics*, 10(11), 2021, 1257, https://www.mdpi.com/2079-9292/10/11/1257 (accessed on 09/01/2022).

[8] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking," in *ACM CoNEXT '10*, Philadelphia, PA, December 2010, http://www.fukuda-lab.org/mawilab/.

[9] Y. Fu, F. Lou, F. Meng, Z. Tian, H. Zhang, and F. Jiang, "An Intelligent Network Attack Detection Method Based on RNN," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, June 2018, 483–9.

[10] K. A. Jalil, M. H. Kamarudin, and M. N. Masrek, "Comparison of Machine Learning Algorithms Performance in Detecting Network Intrusion," in *2010 International Conference on Networking and Information Technology*, June 2010, 221–6, DOI: 10.1109/ICNIT.2010.5508526.

[11] P. Khuphiran, P. Leelaprute, P. Uthayopas, K. Ichikawa, and W. Watanakeesuntorn, "Performance Comparison of Machine Learning Models for DDoS Attacks Detection," in *2018 22nd International Computer Science and Engineering Conference (ICSEC)*, November 2018, 1–4, DOI: 10.1109/ICSEC.2018.8712757.

[12] A. Koay, A. Chen, I. Welch, and W. K. G. Seah, "A New Multi Classifier System using Entropy-based Features in DDoS Attack Detection," in *2018 International Conference on Information Networking (ICOIN)*, January 2018, 162–7, DOI: 10.1109/ICOIN.2018.8343104.

[13] Y.-K. Lai, P.-Y. Huang, H.-P. Lee, C.-L. Tsai, C.-S. Chang, M. H. Nguyen, Y.-J. Lin, T.-L. Liu, and J. H. Chen, "Real-Time DDoS Attack Detection using Sketch-based Entropy Estimation on the NetFPGA SUME Platform," in *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Auckland, New Zealand, December 2020.

[14] X. Ma and Y. Chen, "DDoS Detection Method Based on Chaos Analysis of Network Traffic Entropy," *IEEE Communications Letters*, 18(1), 2014, 114–7.

[15] J. Mao, W. Deng, and F. Shen, "DDoS Flooding Attack Detection Based on Joint-Entropy with Multiple Traffic Features," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, August 2018, 237–43, DOI: 10.1109/TrustCom/BigDataSE.2018.00045.

[16] F. Meng, Y. Fu, F. Lou, and Z. Chen, "An Effective Network Attack Detection Method Based on Kernel PCA and LSTM-RNN," in *2017 International Conference on Computer Systems, Electronics and Control (ICCSEC)*, December 2017, 568–72, DOI: 10.1109/ICCSEC.2017.8447022.

[17] M. H. Nguyen, Y.-K. Lai, and K.-P. Chang, "An Entropy-based DDoS attack Detection and Classification with Hierarchical Temporal Memory," in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, December 2021, 1942–8.

[18] M. P. Novaes, L. F. Carvalho, J. Lloret, and M. L. Proença, "Long Short-Term Memory and Fuzzy Logic for Anomaly Detection and Mitigation in Software-Defined Network Environment," *IEEE Access*, 8, 2020, 83765–81, Conference Name: IEEE Access, DOI: 10.1109/ACCESS.2020.2992044.

[19] X. Qin, T. Xu, and C. Wang, "DDoS Attack Detection Using Flow Entropy and Clustering Technique," in *2015 11th International Conference on Computational Intelligence and Security (CIS)*, December 2015, 412–5, DOI: 10.1109/CIS.2015.105.

[20] S. Rajagopal, P. P. Kundapur, and H. K. S., "Towards Effective Network Intrusion Detection: From Concept to Creation on Azure Cloud," *IEEE Access*, 9, 2021, 19723–42, https://ieeexplore.ieee.org/document/9335932/ (accessed on 05/21/2022).

[21] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, "Practical real-time intrusion detection using machine learning approaches," *Computer Communications*, 34(18), 2011, 2227–35, DOI: 10.1016/j.comcom.2011.07.001, (accessed on 10/28/2019).

[22] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy," in *2019 International Carnahan Conference on Security Technology (ICCST)*, CHENNAI, India: IEEE, October 2019, 1–8, DOI: 10.1109/CCST.2019.8888419, https://ieeexplore.ieee.org/document/8888419/ (accessed on 05/21/2022).

[23] C.-S. Shieh, W.-W. Lin, T.-T. Nguyen, C.-H. Chen, M.-F. Horng, and D. Miu, "Detection of Unknown DDoS Attacks with Deep Learning and Gaussian Mixture Model," *Applied Sciences*, 11(11), 2021, 5213, https://www.mdpi.com/2076-3417/11/11/5213 (accessed on 05/21/2022).

[24] R. Singh, H. Kumar, and R. K. Singla, "An intrusion detection system using network traffic profiling and online sequential extreme learning machine," *Expert Systems with Applications*, 42(22), 2015, 8609–24, http://www.sciencedirect.com/science/article/pii/S0957417415004753 (accessed on 10/28/2019).

[25] I. Sumaiya Thaseen and C. Aswani Kumar, "Intrusion detection model using fusion of chi-square feature selection and multi class SVM," *Journal of King Saud University - Computer and Information Sciences*, 29(4), 2017, 462–72, DOI: 10.1016/j.jksuci.2015.12.004.