Hybrid Approach for Botnet Detection Using K-Means and K-Medoids with Hopfield Neural Network

Atef Ahmed Obeidat

Department of Information Technology, Al-Huson University College, Al-Balqa Applied. University, Salt, Jordan

***Abstract*:** In the last few years, a number of attacks and malicious activities have been attributed to common channels between users. A botnet is considered as an important carrier of malicious and undesirable briskness. In this paper, we propose a support vector machine to classify botnet activities according to k-means, k-Medoids, and neural network clusters. The proposed approach is based on the features of transfer control protocol packets. System performance and accuracy are evaluated using a predefined data set. Results show the ability of the proposed approach to detect botnet activities with high accuracy and performance in a short execution time. The proposed system provides 95.7% accuracy rate with a false positive rate less than or equal to 3%.

***Keywords*:** Botnet, K-Means, K-Medoids, Hopfield, Neural Network, Clustering**.**

1. Introduction

The Internet is a public channel that allows several users and nodes to communicate with each other. Online services, especially financial ones, are encountered by attackers. Intrusions are one of the most important problems of electronic security. Many systems, such as government departments, business organizations, and individuals, suffer from this problem. Thus, all of them work on heightening the security level of their systems by addressing their concerns on security issues. Information infrastructure is essential to support critical operations in large systems, such as banking and telecommunications. Intrusions compromise the security of an information system through various means, thereby significantly threatening societies. The concept of intrusion relates to any procedure that attempts to compromise the confidentiality, integrity, and availability (CIA) of resources.

With the currently rapid growth of computer connection to publicly accessible networks, claiming immunity to network intrusions is impossible for any computer system. Early detection and action are significant in preventing any possible damage because no complete solution exists to prevent the occurrence of intrusions. Our work aims to construct a model that captures a set of transfer control protocol (TCP) attributes and determine if these attributes belong to a botnet or a normal network [1]. The set of input attributes comprise the TCP characteristics of the user network. The output result indicates whether logging on a network is botnet or normal.

Clustering techniques and the neural network model have become promising artificial intelligence (AI) approaches for improving the search for malicious events or invasion attempts in computer networks due to their capabilities to compact knowledge representation[2, 3].

A botnet is considered as a common means of achieving attacker goals. Furthermore, it is a network for sending commands and receiving results managed by a botmaster (attacker). Peer-to-Peer (P2P) botnet is a new type of botnet that aims to avoid a single point of failure as in the old botnet. In this work, we construct a hybrid framework for detecting botnet and normal networks.

A miscellaneous classification methods were used to determine the type of intrusion, some of them are: Statistic Analysis, Neural Network, Rule-Based Analysis, and Data Mining[4, 5]. In statistical analysis, the system will record the normal behavior of computer and the frequency of operation, and then compare them by incoming actions to determine if they are legal or not. The Neural Network builds a supervised or unsupervised model through training the system on normal and abnormal behavior in order to infer them in the future. In Rule-Based Analysis, a set of rules will be created by the computer security expert for safe and unsafe computer operations. The Bayesian Network classifies any new incoming data according to the probability of events and behaviors within system itself. Data mining techniques uses the feature of fields to construct clusters or labels in aim to detect the type, mother cluster or label of new data item. The proposed system merges neural network with data mining to give a hybrid system between them. It uses k-means and k-medoids as clustering methods for finding SVM features. The framework improves support vector machine (SVM) manipulation through the neural network. The proposed model consists of construction and detection phases. In the construction phase, the system uses a training data to build a support machine from the results of clustering algorithms and neural network. By contrast, the detection phase builds an SVM from a data store to check the type of records in the evaluation data set and evaluate our model.

K-medoids and k-means are two common simple classifiers that look to classify a given data by using the statistical properties and distance measures between data set items. They base on the value of K that represents the number of required clusters. They start from unknown (rubbish data) to construct a known data (clusters or labels) as you will see in ulterior subsections[6, 7]. The proposed model uses Hopfield neural network for training and detection. It is the simple fully connected single layer neural network[8, 9]. It is used for the classification problems by applying the binary pattern vectors. In our work, the fitness function was modified to work with the decimal data of training and evaluating data set as shown in the next subsections.

In phase one, the system performs the features extraction process from the input data to reduce the dimensionality of search space. This phase will be carried out using k-means and k-medoids clustering algorithms and the Hopfield neural network[10-12]. The neural network receives a main copy of vectors and features extracted from the clustering process to obtain and enhance new representative vectors. Meanwhile, in the detection phase, an SVM that works as a classifier tool to increase the speed of detecting the type of network will be constructed. SVM will receive data from the environment and then compare it with a set of vectors constructed in the first phase.

The rest of the paper is organized as follows. Section 2 presents Literature review. Section 3 describes the materials and methods. The experimental results are shown in Section 4. Finally, Section 5 concludes this paper with directions for future work.

2. Literature review

In recent years, botnet detection and tracking have been major research topics. Although several approaches exist in the literature as in [13-23], many of the methods cannot detect botnets efficiently. Early works on botnet detection are mainly based on payload analysis, a method to examine malicious signatures in TCP and UDP packets. Payload inspection methods are usually resource intensive and slow because they require the parsing of big packet data. New bots also frequently utilize encryption and other methods to conceal communication and packet inspection. Given the drawbacks of existing methods, botnet detection approaches based on flow analysis have been proposed [24].

Botnet detection can be classified into flow-based, resource-based, node-based, conversation-based, mining-based, and signature-based detection[25]. Flow-based methods as in [26] bear two key limitations. First, several flows between any two network nodes should be analyzed. However, most of these flows are integrated in normal network processes. Second, flow features must be extracted at runtime, which implies that flow-based analysis requires considerable computational overhead at runtime. At any given instance, a significant number of flows exists in a network, and this condition can further aggravate the aforementioned limitations. Resource-based methods are built into the training phase wherein the normality model of legitimate resources may not contain all the cases. Node-based methods, which also depend on the training phase, is constructed on the basis of the features extracted from certain P2P bots.

Conversation-based methods are based on network behavior anomalies, such as duration of conversation and number of packets exchanged in the conversation. However, C&C traffic usually does not reveal anomalous behavior. C&C traffic is also difficult to differentiate from usual traffic behavior. In such cases, conversation-based techniques may fail. Mining-based methods, when used as machine-learning techniques, are suitable for extracting unexpected network patterns. Signature-based methods fail to detect new types of botnets.

Other dimension can be used to study the methods of botnet detections[2, 25] is the classification based on detection algorithms. These algorithms are: 1) Instance-based Learner (including IBk, or k nearest neighbors), 2) Naive Bayes, 3) Support Vector Machines, 4) Decision Trees. Moreover, they also tried and evaluated classifier combination methods. Boosting is the used algorithm and it was performed on SVM, decision tress, and Naive Bayes.

Alauthaman et al. [27] recently proposed a P2P botnet detection scheme based on decision tree and adaptive multilayer neural networks. Specifically, the adaptive multilayer feed forwards neural network with a decision tree for P 2P botnet detection. The proposed method has many Limitations. It does not work in real-time, it does not respond directly to the new features. In addition it does not have the ability to detect botnets that use the UDP protocol to communicate.

Rahbarinia et al. [28] proposed PeerRush, which uses one-class classification to categorize various types of normal and abnormal P2P traffic. Initially, an application profile is created from the traffic samples of known P2P applications. Features, such as interval delays between packets and flow duration, are used to classify P2P applications. On the basis of selected features, the said approach achieves high accuracy for P2P application classification. However, the method does not clearly explain the process of P2P botnet detection. In addition, detection can be easily avoided by changing the delay in-between packets.

In 2014, Zhang et al. [29] proposed an approach to enhance system performance in terms of scalability and efficiency. The method includes two main phases: (1) recognition that all machines are possibly involved in P2P connections and the extraction of statistical fingerprints from the P2P traffic profile, and (2) analysis of P2P host traffic for classification as either P2P bots or legitimate P2P hosts. In the experiment, four P2P applications and two bot were used as dataset. P2P flows, which were clustered hierarchically, were used to identify legitimate P2P traffic from P2P botnet traffic with high rates. However, the method does not clearly explain the result with a new one.

Zhao and Traore [30] introduced a P2P botnet detection technique based on recognizing the malicious behavior of fast-flux networks. They calculated the metrics for captured network traffic, which were then used to identify botnet traffic. This approach based on decision tree algorithm is highly accurate. A decision tree is utilized as a feature set (i.e., reduction mechanism), to exclude the insignificant features of the network. The amount of required data is downsized, thus allowing for enhanced classification accuracy and learning rates and reduced computational time.

Zhao et al. [31] proposed a botnet detection system based on traffic behavior analysis and flow intervals. The Reduced Error Pruning algorithm (REPTree) was used to classify malicious and non-malicious traffic, However, the detection system produced high false-positive rates.

Saad et al.[14] studied the characteristics and behavior of network traffic to detect P2P botnet command and control under radar on the basis of malicious e-mail, websites, file-sharing networks, and ad hoc wireless networks. Using ISOT botnet dataset as benchmark, five different machine-learning algorithms were employed to isolate botnet traffic. The study obtained a low accuracy rate of 89%.

To isolate malicious IRC bot traffic from normal traffic, Lu et al.[32] proposed a detection system that analyzes the

temporal-frequent characteristics of 256 ASCII bytes on payload over a predefined time interval. However, the location and timing of the collected traces were not explicitly specified.

Kirubavathi et al.[33] designed an HTTP-based botnet detection system using the adaptive learning rates from multilayer feedforward neural networks. For detection purposes, TCP-connection-related features at specific time intervals were extracted.

Wang et al. [34] proposed a behavior-based botnet detection system based on fuzzy-pattern recognition techniques. However, this system may lead to false positives when network-generated traces contain regular network activities (e.g., checking new software updates).

Huang [35] developed a host-based botnet detection system based on the network failure model. Network failure was assumed as inherent to botnet traffic; that is, the failure is caused by missing C&C servers, peers, or attacked targets. The 34 features extracted from failure flows were classified into 6 categories. Huang proposed that the model could detect bots with 99% accuracy. However, in instances when bots failed to generate failure, they are missed.

The work in [36] a two-tier detection framework to detect parasite P2P botnets. The approach can detect botnets in their waiting stage and without any requirement of bots' signature. It is used two features: (i) long-living peers, search requests' (ii) intensity and (iii) temporal correlated behavior. The experiment was used to evaluation only one of both types of malicious traffic and benign traffic that may not be given accurate results.

Chen, R., et al. [37] present botnet detection system, it distinguishes malicious botnet traffic using conversation-based traffic analysis and supervised machine learning. They evaluated performances of the five well supervised machine learning algorithms. The work obtained a low accuracy rate of 93.6%. In addition, it's evaluation based on specific botnet category of dataset.

The proposed model overcomes a number of problems in existing botnet detection approaches. The proposed method aims to detect malicious node at its first occurrence and execute actions to reduce damage attack. The inputted attribute set consists of TCP characteristics of the user network. Network activities of various types of bots, such as Zeus, Storm, and Waledac, are observed. The significant features of traffic flows were extracted and applied with classification techniques to isolate botnets from normal traffic flows. The proposed model possesses a number of significant features compared with previous related works. The proposed system is novel because it uses k-means and k-medoids as clustering methods, as well as the Hopfield neural network, to find better SVM features compared with previous approaches.

3. Materials and Methods

Our research combines the data mining field with AI to improve the performance of the artificial neural network in botnet detection; therefore, the proposed method works in multi-directions and phases. As previously mentioned, the model consists of construction and detection phases. The general framework of the proposed system will be discussed in the succeeding subsections.

3.1 Construction Phase

The construction phase comprises of multi-units that work with each other to complete the main task of SVM construction. The phase starts by reading data or records from the training data set, computing similarity between records, extracting main features for each type of records, and improving SVM features by using the Hopfield neural network. Figure 1 shows the construction phase framework.

3.1.1 Fetching and Similarity Unit

Fetching unit reads data from the training data set and then converts them to a compatible form that can be manipulated by the similarity sub-unit. After completing the conversion, the similarity between records will be computed through similarity unit by utilizing the Euclidean distance shown in (1).

$$Simi.(F_{ki}, F_{kj}) = \sqrt{\sum_{k=1}^{8} \left(F_{ki} - F_{kj}\right)^2} \quad (1)$$

Where,
i and j refer to record i and record j.
k refers to feature index.
F refers to feature.

The dataset consists of records such that each one stores a real type of network and eight features. Each feature has a special meaning[38], as shown in Table 1. The similarity between records will be determined according to the value of each feature.

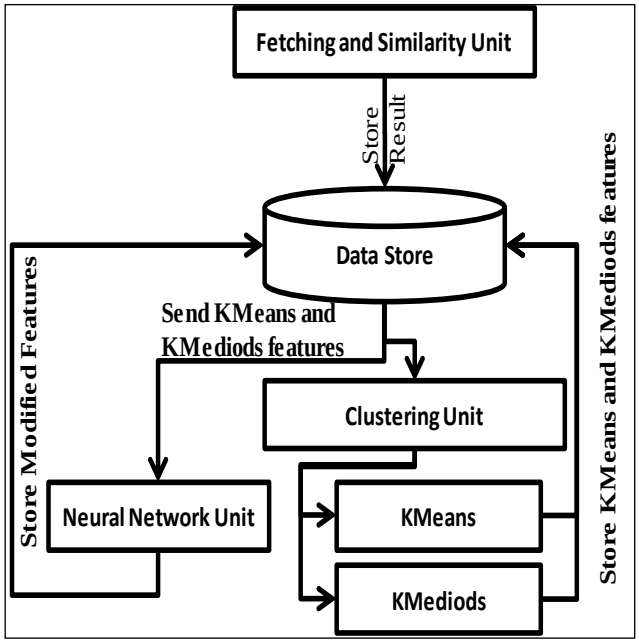

Figure1. Frame work of construction process

3.1.2 Clustering Unit

This unit receives data from the data store, which it then converts into an appropriate format and sends to the clustering algorithm. The clustering unit utilizes two common algorithms, namely, k-means and k-medoids [39, 40], to accomplish their task. The letter k in the name of the algorithm represents the number of clusters based on the main types of botnet and normal data. The center of each cluster (centroid) in the initial state is randomly assigned (centroid vectors will be assigned using a random number

generator function) by using a full period random generator to provide each record a chance to become a centroid. K-means, k-medoids, and their tasks will be discussed in the proceeding subsections.

Table 1. Data set features and their meaning

Feature	Meaning
F1	Total number of sent control TCP packets per connections
F2	Total number of received TCP control packets
F3	Total number of TCP control packets
F4	Total length of sent TCP control packets
F5	Total length of received TCP control packets
F6	Average length of sent TCP control packets
F7	Average length of received TCP control packets
F8	Average length of TCP control packets

- *K-Means algorithm*

The k-means nearest neighbor classifier is a simple and popular classifier that uses statistical properties and distance measures to cluster items into specific classes. The main idea of this technique is to define k centroids or means one for each cluster [7, 39]. The k-means method is reasonably effective. The workflow of the k-means algorithm starts by determining parameter k (number of centroids). Instances are then assigned to their closest cluster center according to the ordinary Euclidian distance function. Next, the centroid or mean of all instances in each cluster is calculated, and this is called the "means" part. Afterwards, these centroids are taken to be new center values for their respective clusters. Finally, the entire process is repeated with new centers. Iteration continues until the same points are assigned to each cluster in consecutive rounds or the maximum number of iterations is reached. The k-means algorithm steps are shown in Figure 2.

- *K-Medoids Algorithm*

The k-medoids algorithm is a clustering algorithm related to the k-means algorithm.

Algorithm: kMean.
Input: *The number of clusters K and a dataset for intrusion detection*
Output: *A set of K-clusters that minimize the squared-error criterion.*
1.Initialize K clusters (randomly select k elements from the data points as the Means).
2. Repeat Until Cluster Structure does not change.
2.1. Determine the cluster to which each data point belongs i.e. to the closest Mean.("closest" here is defined using any valid distance metric; " use Euclidean distance").
2.2. Calculate the new Means of the Clusters.
2.3. Change Clusters Centroids to means obtained, using step 2.1.

Figure 2. KMeans algorithm

K-medoids is more robust to noise and outliers as compared to k-means. A medoid can be defined as the cluster's object, whose average dissimilarity to all the objects in the cluster is minimal, that is, it is the most centrally located point in the given data set [7, 39].

The k-medoids clustering aims to find a non-overlapping set of clusters such that each has the most representative point, that is, a point that is most centrally located with respect to a number of measures, such as distance. These representative points are called medoids. The k-medoid algorithm is relatively simple, and this algorithm is clearly expensive compared to k-means. The k-medoids steps are shown Figure 3.

Algorithm: KMediods.
Input: *The number of clusters K and a dataset for intrusion detection*
Output: *A set of K-clusters that minimize the squared-error criterion.*
1. Initialize: randomly select k elements from the n data points as the Medoids
2. Associate each data point to the closest Medoid. ("closest" here is defined using any valid distance metric; "use Euclidean distance").
3. For each Medoid m
3.1. For each non-Medoid data point n
3.1.1. Swap m and n and compute the total cost of the configuration
4. Select the configuration with the lowest cost.
5. Repeat steps 2 to 5 until there is no change in the Medoid in the old and new model.

Figure 3. KMediods algorithm

3.1.3 Training Unit

Algorithm: Hopfield Learning.
Input: *Result of KMeans or KMediods and training data set records.*
Output: *Set of Centers that represents data set records to be used as SVM in detection phase .*
1. Initialize and assign connections Weights.

$$W_{ij} = \sum_{S=0}^{N-1} x_i^S x_j^S \mu \qquad i \neq j, \; x^S \in \{-1,1\}, \; 0 \leq i, j \leq N-1$$

Where , Wij : is the connection weight from record i to record j
2. Initialize with Unknown Input Pattern.

$$\mu_i(t) = x_i, \qquad 0 \leq i \leq N-1, \; x_i \in \{-1,1\}$$

Where, μi (t) is the output of record i at current time t
3. Iterate Until Convergence

$$\mu_i(t+1) = \sum_{i=0}^{N-1} w_{ij} \mu_i(t) \qquad 0 \leq j \leq N-1$$

Where, μi (t+1) is the output of record i at next time t+1
4. Repeat from step 2 until the output remains unchanged with further iterations.

Figure 4. Hopfield Learning Algorithm

This unit generates a new set of comparable vectors. Furthermore, this unit consists of two parts that can be summarized as follows: main clusters vectors which represent the sample set of normal and attack types produced in the clustering process and the hybrid neural network model used to generate a new set of vectors to represent each class of normal and attack data types.

Hopfield neural network is the simplest form of neural network. This network is a fully connected single layer auto-associative network, which means that it has a single layer with each neuron connected to every other neuron. Hopfield networks are typically used for classification problems with binary pattern vectors. The Hopfield network is created by supplying input data or pattern vectors corresponding to different classes. These patterns are called class patterns [11, 40]. Figure 4 shows the main steps of the Hopfield learning algorithm.

3.2 Detection Phase

After completing the construction phase, our system starts the evaluation process through the detection phase. K-means, k-medoids, and Hopfield neural network stored the results of their training in a data store, which may be utilized in building a comprehensive SVM system. Fig.5 shows the general framework for the detection phase, which consists of the data store, the environment, and the detection unit. The environment provides the proposed system with the evaluation data set and receives the detection unit results.

Data store stores the results of the used method in a special buffer called an SVM. SVM contains the method index, feature name, and feature value. The detection unit constructs vectors of classifiers by using the SVM buffer of the data store. Vectors of classifier units receive records from the environment and then compute distances between the environment and the classifier vectors. The results of classifiers will be returned to the environment in order to provide the user with an indication regarding the type of records and to evaluate the system. Fig.6 shows the main steps of the detection phase and the manner in which tasks are accomplished.

4. Results and Analysis

A data set was utilized to train and test the feasibility of the model and evaluate its performance. The training data set consists of 33367 records, such that 2% of the samples are normal data and the others are botnets. Conficker, Waledac, and Storm Bot are the main types of bot obtained in the evaluation process, representing 89%, 1%, and 8%, respectively, from the main data. The use of new data outside the training set is significant for making the testing data more realistic.

The system was evaluated by using several criteria, such as accuracy and true-false positive and negative rates. True negative rate (TNR) and true positive rate (TPR) indicate the correctness classification for a record type. False positive rate (FPR) classifies a record as anomalous (a possible botnet) when such record is legitimate. A good system eliminates or reduces this type of errors to provide useful information for the user. Meanwhile, the false negative rate (FNR) system may classify an anomalous data as a normal one, thereby allowing it to pass without any alert for the user.

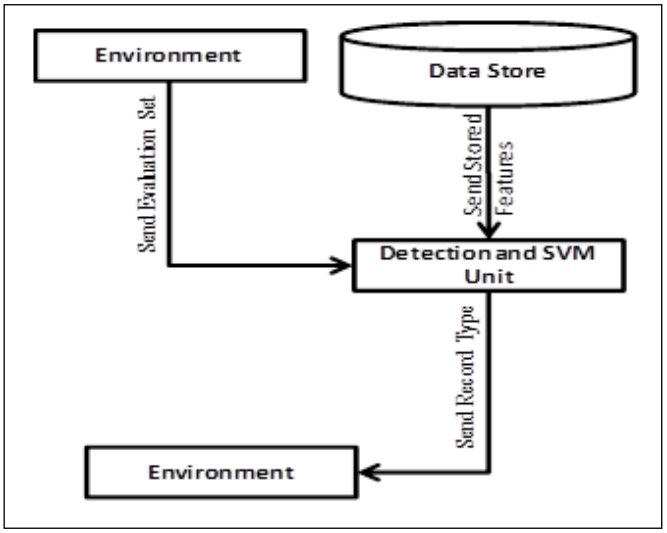

Figure 5. Detection Framework

Algorithm: Detection algorithm

Input: *- Result of KMeans, KMediods and Hopfield neural network*
- Environment records (Real or evaluation records)

Output: *Types of inserted records*

```
1.  Input N records from environment
2.  Construct M classifiers
3.  For i=1 to N Do
4.    For j=1 to M
5.      Distance= Simi.(classifier_j, Record_i )
6.      If j= i Then
7.          MinDistance=Distance
8.          RecordType=ClassifierType
9.      Else
10.         If  MinDistance > Distance Then
11.           MinDistance=Distance
12.           RecordType=ClassifierType
13.         End IF
14.     End If
15.   Next j
16.   Return record type to environment
17. Next i
```

Figure 6. Detection algorithm

$$Accuracy = (TP + TN) / (TP + FN + FP + TN)$$
$$TPR = TP / (TP + FN)$$
$$TNR = TN / (FP + TN)$$
$$FNR = FN / (TP + FN)$$
$$FPR = FP / (FP + TN)$$

- FP: total number of false positive
- TP: total number of true positive
- FN: total number of false negative
- TN: total number of true negative

Figure 7. Accuracy, TPR, TNR, FPR and FNR Equations

Figure 7 shows the main equations of accuracy, TNR, TPR, FPR, and FNR.

The evaluating and training data sets store eight features as mentioned earlier. Thus, we took several cases to check the performance of the proposed system, as shown in Table 2. After determining the cases, the evaluator system computes the values of TN, TP, FP, and FN at different cases

according to the result of the proposed system. Different cases are selected to manipulate the system in those cases and determine the least and most significant features in the data set utilized

Table 2: Evaluation Cases

Case	Features	Case	Features
1	F1	7	F1..F7
2	F1..F2	8	F1..F8
3	F1..F3	9	F5
4	F1..F4	10	F5..F6
5	F1..F5	11	F5..F7
6	F1..F6	12	F5..F8

Table 3 shows the accuracy of the methods used at different cases. In terms of accuracy, the k-medoids is the best one due to the random selection of centers and the equivalence between data set records that were represented as centers in the clustering process. Hopfield over k- medoids (HKMedoids) ranked second because the learning machine is constructed by using the SVM of k-medoids. K-means and Hopfield–K-means (HKMeans) get down approximately at the end of the arrangement because they suffer from outlier records.

Table3. Accuracy of used Methods

Features	HKMeans	HKMedoids	KMeans	KMedoids
F1	0.1489	0.1422	0.1822	0.1422
F1..F2	0.1711	0.1422	0.8511	0.1444
F1..F3	0.1711	0.1444	0.84	0.1444
F1..F4	0.2333	0.2111	0.86	0.2111
F5	0.8022	0.8333	0.8444	0.8644
F5..F6	0.8044	0.8333	0.8467	0.8644
F5..F7	0.8333	0.8644	0.8489	0.8644
F5..F8	0.8156	0.8911	0.8533	0.8911
F1..F5	0.8822	0.9	0.88	0.9578
F1..F6	0.8844	0.9	0.8844	0.9578
F1..F7	0.8778	0.9311	0.8867	0.9578
F1..F8	0.8911	0.9578	0.9044	0.9578

The irregularity of k-means appears in Cases 2, 3, and 4 with accuracy values of 0.85 to 0.86 compared with other methods that have values in the range 0.14 to 0.23. Figure 8 shows the accuracy of each method and the manner in which the precedence of k-medoids clearly appears. According to Table 3, Feature F5 provides strong evidence regarding its significance in discovering the type of record.

The TPR value is based on the computation case. Thus, our system provided different values for various methods to detect the bot types of records, as shown in Table 4. The best result for TPR appeared in Case 8, which takes all the features of records. Several cases provided a comparable result to Case 8, such as Cases 5, 6, and 7. The convergence between these cases refers to the utilization of a significant feature in them.

NR computes the number of correctly classified normal records. Table 5 provides the TNR for different methods. This table shows that Cases 3 12 are respectively the most significant cases for k-medoids and Hopfield k-medoids, whereas Cases 9 and 10 are the best cases for k-means and Hopfield k-means. The variation in the results among several methods refers to the selection and computation processes in clustering for k-medoids and k-means.

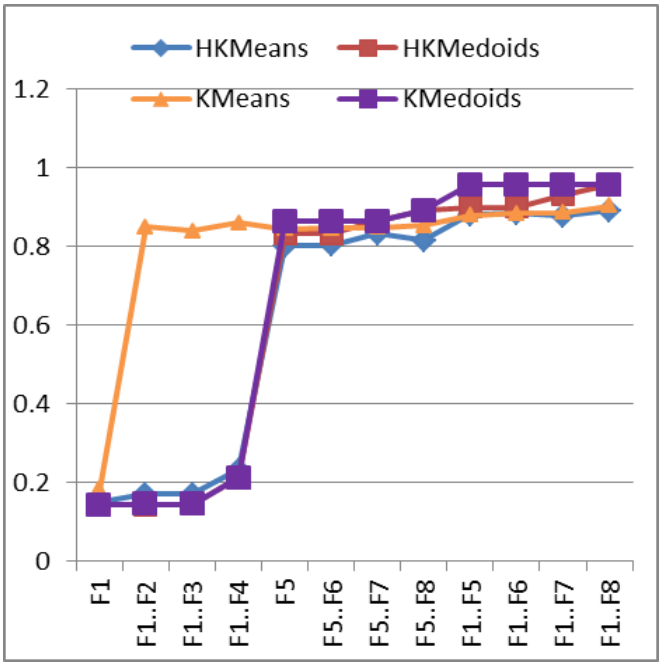

Figure 8. Accuracy of used Methods

Table 4. TPR of used Methods

Case	HKMeans	HKMediods	KMeans	KMediods
F1	0.481	0	0.7692	0
F1..F2	0.679	0	0.9835	0.0026
F1..F3	0.7568	0.0026	0.9835	0.0026
F1..F4	0.9512	0.082	0.9945	0.082
F1..F5	0.9918	0.9317	0.9945	0.9709
F1..F6	0.9918	0.9317	0.9945	0.9709
F1..F7	0.9918	0.9699	0.9945	0.9709
F1..F8	0.9921	0.9709	0.9947	0.9709
F5	0.7876	0.8316	0.8503	0.869
F5..F6	0.7902	0.8316	0.8529	0.869
F5..F7	0.8717	0.869	0.8743	0.869
F5..F8	0.8984	0.8776	0.8987	0.8776

FPR determines the number of normal records classified as bots. The system designer reduces or eliminates FPR because of the accuracy problems encountered by the system itself.

K-medoids and their neural network variations provide a surplus result compared to k-means and their variations, as shown in Table 6.

Furthermore, FNR considers a problem to system security, and then, it computes the number of bot records classified as normal; thus, the constructor or designer attempts to reduce or eliminate this value every time. K-medoids provides a large value in the initial state, which is reduced in the next stage by adding new features for data set records. K-means starts with a small value that is reduced over different cases. K-medoids is more significant than k-means to a number of features; thus, the FNR value is rapidly increasing, as shown in Table 7

Table 5. TNR of used Methods

Case	HKMeans	HKMediods	KMeans	KMediods
F1	0.0782	1	0.0591	1
F1..F2	0.0596	1	0.2907	1
F1..F3	0.0559	0.9697	0.2326	0.9697
F1..F4	0.0734	0.8889	0.2907	0.8889
F1..F5	0.3976	0.7619	0.381	0.8889
F1..F6	0.4096	0.7619	0.4048	0.8889
F1..F7	0.3735	0.7619	0.4167	0.8889
F1..F8	0.3611	0.8889	0.4306	0.8889
F5	0.8906	0.8421	0.8158	0.8421
F5..F6	0.8906	0.8421	0.8158	0.8421
F5..F7	0.6447	0.8421	0.7237	0.8421
F5..F8	0.3333	0.9697	0.5846	0.9697

Table 6. FPR of used Methods

Case	HKMeans	HKMedoids	KMeans	KMedoids
F1	0.9218	0	0.9409	0
F1..F2	0.9404	0	0.7093	0
F1..F3	0.9441	0.0303	0.7674	0.0303
F1..F4	0.9266	0.1111	0.7093	0.1111
F1..F5	0.6024	0.2381	0.619	0.1111
F1..F6	0.5904	0.2381	0.5952	0.1111
F1..F7	0.6265	0.2381	0.5833	0.1111
F1..F8	0.6389	0.1111	0.5694	0.1111
F5	0.1094	0.1579	0.1842	0.1579
F5..F6	0.1094	0.1579	0.1842	0.1579
F5..F7	0.3553	0.1579	0.2763	0.1579
F5..F8	0.6667	0.0303	0.4154	0.0303

Finally, k-medoids and their variations are better than k-means in all checked criteria. K-medoids provides the same chance for all records in the data set to become representative centers, which select the best one between them. By contrast, k-means based on the mean value for the next representative centers; thus, no switching existed between records. The mean computation for the center in k-means may produce an outlier in clusters caused by the k-means clustering process. This work clearly supports the precedence of k-medoids over k-means.

Table 7. FNR of used Methods

Case	HKMeans	HKMedoids	KMeans	KMedoids
F1	0.519	1	0.2308	1
F1..F2	0.321	1	0.0165	0.9974
F1..F3	0.2432	0.9974	0.0165	0.9974
F1..F4	0.0488	0.918	0.0055	0.918
F1..F5	0.0082	0.0683	0.0055	0.0291
F1..F6	0.0082	0.0683	0.0055	0.0291
F1..F7	0.0082	0.0301	0.0055	0.0291
F1..F8	0.0079	0.0291	0.0053	0.0291
F5	0.2124	0.1684	0.1497	0.131
F5..F6	0.2098	0.1684	0.1471	0.131
F5..F7	0.1283	0.131	0.1257	0.131
F5..F8	0.1016	0.1224	0.1013	0.1224

5. Conclusion

Numerous works have been established in this field, but the novelty of this research lies in its combination of clustering (data mining) and neural network (artificial intelligence), which provides a unique system for manipulating such complexity. In this regard, our approach firstly reads the data set records and computes the similarity between them. Second, our approach constructs an SVM classifier using k-means and k-medoids clustering algorithms. Finally, the clustering process results are employed in the Hopfield neural network as a learning machine for detecting different types of bots. The results show that k-medoids and their variations improved more than k-means with different variations. In the future, we plan to utilize other approaches for learning, such as Kohonen, to enhance system accuracy and performance.

6. Acknowledgment

This research is supported by Dept. of Information Technology, Al-huson University College, Al- Balqa Applied University.

References

[1] Abu-Khalaf, M., EE 5322 Neural Networks Notes, Personal Study, arri.uta.edu/acs/abumurad/EE5322/EE5322_NN_notes.pdf , 2004.

[2] Botha, M., et al. The utilization of artificial intelligence in a hybrid intrusion detection system. in Proceedings of the 2002 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology. 2002:

South African Institute for Computer Scientists and Information Technologists, pp. 149-155,2002.

[3] Bivens, A., et al., "Network-based intrusion detection using neural networks", Intelligent Engineering Systems through Artificial Neural Networks, 2002. 12(1): pp. 579-584, 2002.

[4] Vapnik, V.N. and V. Vapnik, Statistical learning theory. Vol. 1. 1998: Wiley New York,1998.

[5] Ilgun, K., R.A. Kemmerer, and PP.A. Porras, "State transition analysis: A rule-based intrusion detection approach", IEEE transactions on software engineering, 1995. 21(3): pp. 181-199, 1995.

[6] Siraj, M.M., M.A. Maarof, and S.Z. Hashim, "Intelligent alert clustering model for network intrusion analysis". Int. J. Advance. Soft Comput. Appl, 2009, 1(1), pp.1-16, 2009.

[7] Kaplantzis, S. and N. Mani. "A study on classification techniques for network intrusion detection". in IASTED Conference on Networks and Communication Systems (NCS 2006), Thailand, http://users.monash.edu.au/~skap3/ncs2006.pdf, 2006.

[8] Lippmann, R., "An introduction to computing with neural nets". IEEE Assp magazine, 1987. 4(2): pp. 4-22,1987.

[9] Rhodes, Brandon Craig, James A. Mahaffey, and James D. Cannady. "Multiple self-organizing maps for intrusion detection." In Proceedings of the 23rd national information systems security conference, pp. 16-19, 2000.

[10] Hopfield, J.J., "Artificial neural networks", IEEE Circuits and Devices Magazine, 1988. 4(5): pp. 3-10, 1988.

[11] Makki, A., and P. Siy. "Hopfield neural networks control for optimal solutions," In Neural Networks, 1992. IJCNN., International Joint Conference on, vol. 4, IEEE, pp. 462-467, 1992.

[12] Portnoy, Leonid, Eleazar Eskin, and Sal Stolfo. "Intrusion detection with unlabeled data using clustering." In In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001, pp.5-8, 2001.

[13] Wei, Xianglin, Jianhua Fan, Ming Chen, Tarem Ahmed, and Al-Sakib Khan Pathan. "SMART: A subspace based malicious peers detection algorithm for P2P systems", International Journal of Communication Networks and Information Security 5, no. 1 (2013): pp.1-9, 2013.

[14] Saad, Sherif, Issa Traore, Ali Ghorbani, Bassam Sayed, David Zhao, Wei Lu, John Felix, and Payman Hakimian. "Detecting P2P botnets through network behavior analysis and machine learning." In *Privacy, Security and Trust (PST), 2011 Ninth Annual International Conference on*, pp. 174-180. IEEE, 2011.

[15] Kheir, N. and C. Wolley, BotSuer: Suing stealthy P2P bots in network traffic through netflow analysis, in Cryptology and Network Security. 2013, Springer. pp. 162-178, 2013.

[16] Jian, K., S. Yuan-Zhang, and J.-Y. Zhang, "Accurate Detection of Peer-to-Peer Botnet using Multi-Stream Fused Scheme", Journal of Networks, 2011. 6(5): pp. 807-814, 2011.

[17] Cholez, T., et al., Detection and mitigation of localized attacks in a widely deployed P2P network. Peer-to-Peer Networking and Applications, 2013. 6(2): pp. 155-174, 2013.

[18] Fan, Y. and N. Xu, "A P2P Botnet Detection Method Used On-line Monitoring and Off-line Detection", International Journal of Security & Its Applications, 2014. 8(3): pp. 87-96, 2014.

[19] Guntuku, S.C., PP. Narang, and C. Hota, "Real-time Peer-to-Peer Botnet Detection Framework based on Bayesian Regularized Neural Network". arXiv preprint arXiv:1307.7464, 2013.

[20] Hang, Huy, Xuetao Wei, Michalis Faloutsos, and Tina Eliassi-Rad. "Entelecheia: Detecting p2p botnets in their waiting stage." In IFIP Networking Conference, 2013, IEEE, pp. 1-9, 2013.

[21] Lei, X., X. XiaoLong, and Z. Yue, P2P Botnet Detection Using Min-Vertex Cover. Journal of Networks, 2012. 7(8): pp. 1176-1181. 2012

[22] Narang, Pratik, Vansh Khurana, and Chittaranjan Hota. "Machine-learning approaches for P2P botnet detection using signal-processing techniques." In Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems, ACM, pp. 338-341, 2014.

[23] Qiao, Y., et al., "Detecting P2P bots by mining the regional periodicity", Journal of Zhejiang University SCIENCE C, 2013. 14(9): pp. 682-700, 2013.

[24] García, S., A. Zunino, and M. Campo, "Survey on network-based botnet detection methods", Security and Communication Networks, 2014. 7(5): pp. 878-903. 2014.

[25] Obeidat, A.A., "Analysis the P2P botnet detection methods", IPASJ International Journal of Computer Science (IIJCS), 2016. 4(3), pp. 1-11. 2016.

[26] Tarng, W., et al., "The analysis and identification of P2P botnet's traffic flows", International Journal of Communication Networks and Information Security, 2011. 3(2), pp. 138-148, 2011.

[27] Alauthaman, M., et al., "A P2P Botnet detection scheme based on decision tree and adaptive multilayer neural networks", Neural Computing and Applications, 2016: pp. 1-14, 2016.

[28] Rahbarinia, B., et al., "Peerrush: Mining for unwanted p2p traffic", Journal of Information Security and Applications, 2014. 19(3): pp. 194-208, 2014.

[29] Zhang, J., et al., "Building a scalable system for stealthy p2p-botnet detection". IEEE transactions on information forensics and security, 2014. 9(1): pp. 27-38, 2014.

[30] Zhao, David, and Issa Traore. "P2P botnet detection through malicious fast flux network identification." In P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2012 Seventh International Conference on, IEEE, pp. 170-175, 2012.

[31] Zhao, D., et al., "Botnet detection based on traffic behavior analysis and flow intervals", Computers & Security, 2013. 39: pp. 2-16, 2013.

[32] Lu, W., G. Rammidi, and A.A. Ghorbani, "Clustering botnet communication traffic based on n-gram feature selection". Computer Communications, 2011. 34(3): pp. 502-514, 2011.

[33] Venkatesh, G. Kirubavathi, and R. Anitha Nadarajan. "HTTP Botnet Detection Using Adaptive Learning Rate Multilayer Feed-Forward Neural Network," In WISTP, pp. 38-48, 2012.

[34] Wang, K., et al., "A fuzzy pattern-based filtering algorithm for botnet detection," Computer Networks, 2011. 55(15): pp. 3275-3286, 2011.
[35] Huang, C.-Y., "Effective bot host detection based on network failure models", Computer Networks, 2013. 57(2): pp. 514-525, 2013.
[36] Dhayal, H. and J. Kumar, "Peer-to-Peer Botnet Detection based on Bot Behaviou", International Journal of Advanced Research in Computer Science, 2017. 8(3). pp.172-175, 2017.
[37] Chen, Ruidong, Weina Niu, Xiaosong Zhang, Zhongliu Zhuo, and Fengmao Lv. "An Effective Conversation-Based Botnet Detection Method." Mathematical Problems in Engineering 2017 (2017). https://doi.org/10.1155/2017/4934082
[38] Mohammad Aluthoman, N.A., M AHosssani and Rafe Alsem, "Investigation of P2P Botnet using TCP control packets and data mining techniques", in 7th international conference on software knowledge , Information management and Application,(SKIMA 2013): Chiang Mai , Thailand,pp.1-9, 2013.
[39] Lai, D. and N. Mani, Support vector machines and linear stationary models, conversion report. 2001.
[40] Faraoun, K. and A. Boukelif, Neural networks learning improvement using the K-means clustering algorithm to detect network intrusions. INFOCOMP Journal of Computer Science, 2006. 5(3): pp. 28-36. 2006.