Investigation of Domain Name System Attack Clustering using Semi-Supervised Learning with Swarm Intelligence Algorithms

Hussain Alibrahim
Department of Computer Science
North Dakota State University
Fargo, USA
hussain.alibrahim@ndsu.edu

Simone A. Ludwig
Department of Computer Science
North Dakota State University
Fargo, USA
simone.ludwig@ndsu.edu

***Abstract*—Domain Name System (DNS) is the Internet's system for converting alphabetic names into numeric IP addresses. It is one of the early and vulnerable network protocols, which has several security loopholes that have been exploited repeatedly over the years. The clustering task for the automatic recognition of these attacks uses machine learning approaches based on semi-supervised learning. A family of bio-inspired algorithms, well known as Swarm Intelligence (SI) methods, have recently emerged to meet the requirements for the clustering task and have been successfully applied to various real-world clustering problems. In this paper, Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), and Kmeans, which is one of the most popular cluster algorithms, have been applied. Furthermore, hybrid algorithms consisting of Kmeans and PSO, and Kmeans and ABC have been proposed for the clustering process. The Canadian Institute for Cybersecurity (CIC) data set has been used for this investigation. In addition, different measures of clustering performance have been used to compare the different algorithms.**

***Index Terms*—Swarm Intelligence, Particle Swarm Optimization, Artificial Bee Colony, Kmeans, Semi-supervised Learning**

I. Introduction

Domain Name System (DNS) is a hierarchical and decentralized naming system for computers, services, and other resources connected to the Internet or a private network. It connects numerous devices with domain names assigned to each of the involving entities. DNS converts easy readable and memorized domain names to numerical IP addresses for finding and identifying computer services and devices using the underlying network protocols. DNS has played a major role for the internet since 1985. The original security design of DNS was adequate to cover the need of the internet at that time. However, because of the widespread use of the internet these days this approach has become a vulnerable network protocol [1]. Nowadays, cyber-attacks are considered a new remote weapon [2] targeting critical infrastructure such as presidential campaigns [3], nuclear programs [4], government personnel data [5], and software providers [6]. It is essential to discriminate between hazard and normal traffic while using the internet.

Securing the DNS system from any unauthorized access is critically important for the operation of private networks and the Internet. To overcome some of the DNS vulnerabilities related to privacy and data manipulation, protocol RFC8484 was introduced to improve privacy and to combat spy and man-in-the-middle attacks by encrypting DNS requests and sending them via a secret tunnel so that data cannot be attacked during the delivery.

An Intrusion Detection System (IDS) [7] plays an important role in supervising the traffic of internet-connected devices and in the discovery of attacks for DNS over HTTPS (DoH) traffic in a network topology. Intrusion detection was described as "the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network" [8]. IDS is the most serious protection tool against sophisticated and ever-growing network attacks. Different IDS systems have been developed to differentiate malicious from normal traffic [9]. Machine learning algorithms [7] have been employed for attack detection such as naive Bayes [8], neural networks [10], support vector machine [11], principal component analysis [12], and random forest [13].

Different machine learning models can be used for determining any suspect traffic over the network. For the purpose of this paper three of these models are given below [14]:

- **Supervised Learning:** This model is known as learning from examples. Since it is used to receive data (features) with their correct label (target) as the training set and thus a trained model can respond more accurately by comparing its output results with the true labels provided. Supervised learning algorithms are used for predicting unseen data based on historical data it was trained on. For example, it can be applied to determine the car type sedan, truck, van, etc. if given enough data, also it can help in recommending flights for a user by using the user's history. The supervised learning paradigm is mostly used on classification problems or regression problems. The main idea for supervised learning is that an algorithm receives the data with their truth label as can be seen in Figure 1(a) and it will predict the class 1 or 0. The main advantages of this approach is that the results are more accurate since the input data is well known and the user is able to determine the number of classes. The disadvantages however are that the modeling can be a complex process with large training times. Examples of common supervised learning algorithms are decision trees, support vector classifier (linear SVC), neural networks, support vector regression (SVR), and gradient boosting.

- **Unsupervised Learning:** This paradigm is used when data has no labels and tries to recognize unidentified existing patterns from the data in order to derive rules, or in other words the modeling tries to find hidden structure in the unlabeled data. Unsupervised learning is used in solving clustering problems or identifying principal components. Figure 1(b) shows the model with the categories of data that are unknown and unsupervised learning tries to cluster the data based on features and the structure without knowing any category or label.
  Opposite to the supervised learning model, the main advantages of unsupervised learning are, taking place in real time, less complex since there is no need to figure out the relation between labels and features as is the case for supervised learning, and it is easier to get data that is unlabeled. However, this model is less accurate, and the results cannot be verified or compared with the ground truth. Common algorithms in unsupervised learning are kmeans, neural networks / deep learning, principal component analysis, singular value decomposition, and independent component analysis.
- **Semi Supervised Learning:** Semi supervised learning algorithms provide a technique that harnesses the power of both - supervised learning and unsupervised learning. There are two ways the target labels are treated; either they are provided for all the instances or no labels are provided. The semi supervised model is used when data has only few labels provided for some of the instances and tries to predict an unlabeled data target. The prediction process uses the supervised learning model in order to learn from labeled instances and by recognizing patterns in others, which is unsupervised learning. Semi supervised learning can be used in classification, regression, clustering, and prediction. Figure 1(c) shows how data was labeled and how the model tries to classify them. The advantage of this model is that it can solve real time problems and usually achieves higher accuracy than unsupervised learning. Furthermore, the results can be compared with the available truth values. The disadvantage is the model can become complex because of the two stages involved that deal with labeled and unlabeled data.

In this paper, a systematic approach is proposed to evaluate the capabilities of five machine learning algorithms to be investigated for analyzing, testing, and evaluating clustering

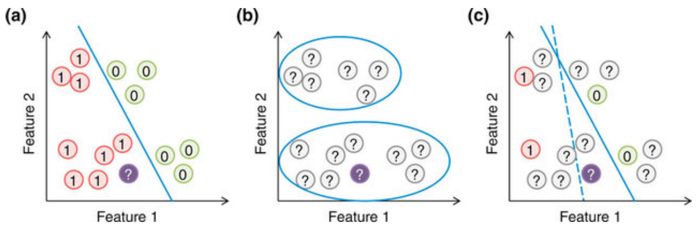


Fig. 1: Learning Algorithms [20]

applied to SSL (semi supervised learning) problems. Algorithms used are K-mean, Particle Swarm Optimization (PSO), and Artificial Bee Colony (ABC). In addition, two hybrid versions of Kmeans with PSO and Kmeans with ABC are proposed. The data set investigated is the Canadian Institute for Cybersecurity (CIC) data that are used for the clustering task.

The remaining sections of this paper are organized as follow: the next section (Section II) describes related work about swarm intelligence algorithms applied to clustering. Details of the algorithms used in this paper are outlined in Section III followed by the experimental setup in Section IV. The results are presented in Section V, and the conclusion is given in Section VI.

II. Related Work

In [15], the power Kmeans algorithms is proposed. It is a generalization of the Lloyd's algorithm that makes it more robust to initialization, increases its performance in multiple dimension spaces, and keeps its speed and simplicity. Power Kmeans embeds the Kmeans problem in a continuum of better-behaved problems. These smoothed intermediate problems have 'flatter' objective functions, which are used to guide the clustering toward the global minimum of the Kmeans objective. Furthermore, it guarantees a decrease in the Kmeans objective at each step. This algorithm was tested on the Down syndrome data set and outperformed self-organizing maps and Kmeans++.

In [16], Kmeans was used to cluster incomplete data. Other than the existing methods that are used to approximate the missing value and then fill in data and finally perform the clustering, the Kmeans filling method was proposed to dynamically optimize the missing values. The new algorithm shows very good performance over seven benchmark data sets and is more robust across a wide range of incomplete approaches for filling missing data. This is the underlying strength of dynamical filling for incomplete data clustering.

In [17], a novel clustering algorithm based on density peaks clustering (DPC) and particle swarm optimization (PSO) is presented. The main objective of this algorithm is to overcome the DPC issues, which are related to mitigating the impact of the selection parameter on the clustering results. One of the steps was to use PSO in the clustering analysis based on the density and distance of the centroid(s). PSO is used because of its simple concept and strong global search ability, which can find the optimal solution in relatively few iterations. New algorithms were compared with six state-of-the-art algorithms over nine data sets and outperformed the others in most of the scenarios.

In [18], a history-driven artificial bee colony (Hd-ABC) approach is proposed to improve the ABC's clustering performance by applying a memory mechanism. Since the fitness evaluation is a costly and time consuming process in clustering, a binary space partitioning (BSP) tree is used to save valuable information of the evaluated solutions. Using the memory mechanism can help to approximate the fitness landscape before the actual fitness computation. With this the number of fitness evaluations is significantly decreased and the optimization process uses the approximate fitness value of solutions instead of calculating the precise fitness values. Moreover, a new local search mechanism is introduced to improve the exploitation ability as well as the convergence speed of the ABC in the onlooker bee phase, which is guided by the anisotropic search (GAS) strategy. The Hd-ABC algorithm is compared with multiple algorithms such as Kmeans, ABC, ACO and with multiple variants of ABC over a different data sets. The results show that the proposed algorithm outperforms the other methods that only use global or local search processes.

In terms of related work that deals with the same data set used in this paper, [19] presents a systematic two-layer approach for detecting DNS over HTTPS (DoH) traffic and distinguishing Benign-DoH traffic from Malicious-DoH traffic using six machine learning algorithms. The machine learning algorithms used were decision tree, extra trees, gradient boosting, XGBoost, light gradient boosting machine (LGBM), and random forest. These algorithms are presented for differentiating DoH traffic from non-DoH traffic in Layer 1, and characterizing Benign-DoH from Malicious-DoH traffic in Layer 2. The evaluation matrices used are accuracy, precision, recall, and F-score, confusion matrices, ROC curves, and feature importance. The results show that the LGBM and XGBoost algorithms outperform the other algorithms in almost all the classification measures reaching the maximum accuracy of 100 in the classification tasks of Layers 1 and 2. The LGBM algorithm only misclassified one DoH traffic test as non-DoH out of 4,000 test in the data set. Feature importance found by LGBM were SourceIP and DestinationIP.

III. Methodology

A. Kmeans

The Kmeans algorithm, as described in [21], is a clustering algorithm that is widely used for clustering large sets of data. It was proposed by MacQueen in 1967 and is one of the most simple, non-supervised learning algorithms. The method classifies the data into K different clusters, where K is predefined before the method is started. The results of the clustering are generated clusters that are dense and independent. The algorithm basically assigns each data object to the nearest centroid in an iterative process until the local minimum is reached. The euclidean distance is generally used to determine the distance between each data object and the cluster centers. The euclidean distance for data object x with N features from centroid y that has the same features can be defined as in Equation 1.

$$d(x,y)=\sqrt{\sum_{i=0}^{N}(x_i-y_i)^2} \tag{1}$$

The detailed process of Kmeans is given in Algorithm 1.

Algorithm 1: Kmeans Algorithm

```
input  : Number of Clusters ($K$)
         Unlabeled Data ($n$)
Output: $K$ clusters contain all elements ($n$)
1  Randomly select $K$ data objects from n
2  Calculate the clusters centers $C_j$ using Eq. (1)
3  while stopping criteria has not been met do
4      for each data object $i$ in $n$ do
5          Calculate the distances between $n_i$ and all $C_j$
           Assign $n_i$ to the nearest cluster
6      end
7      for each clusters in $C_j$ do
8          Update clusters centers $C_j$
9      end
10 end
```

B. Particle Swarm Optimization

The PSO algorithm [22] is one of the well-known optimization methods used for optimization, which finds important features in its feature space providing a local search and global search option. The population consists of particles, which move randomly in the search space and which are optimized via an iterative process to find the best possible solution. This procedure continues until an appropriate convergence is obtained.

This algorithm uses a reasonable number of particles n to solve the specific optimization problem for any number of dimensions c. Each particle in the swarm has a position and velocity with respect to each dimension, and these are randomly initialized at the beginning of the algorithm. The position X of each particle i can be represented in c dimensions as given in Equation (2).

$$X_i=[x_{i_1},x_{i_2},x_{i_3},...,x_{i_c}] \tag{2}$$

Particle velocity v is represented as in Equation (3).

$$v_i=[v_{i_1},v_{i_2},v_{i_3},...,v_{i_c}] \tag{3}$$

As stated earlier, PSO searches for the local best $Pbest_i(t)$, which represents the best location for a particle over the run time t, while $Gbest_i(t)$ is the global best of all particles over the run time. The update equation for the velocity is as in Equation (4). Predefined values in the velocity equation are w that represent the inertia weight, which determines the contribution rate of a particle's previous velocity to its velocity at the current time step [23], c_1 and c_2 are acceleration coefficients together with the random vectors u_1 and u_2, which control the stochastic influence of the cognitive and social components and influence the overall velocity of a particle.

$$\upsilon_i(t+1) = w \times \upsilon_i(t) + c_1 \times u_1 \times (Pbest_i(t) - X_i(t)) + c_2 \times u_2 \times (Gbest_i(t) - X_i(t)) \tag{4}$$

The second important equation is the position update, which depends on the previous location and velocity. The equation is given in Equation (5).

$$X_i(t+1) = X_i(t) + \upsilon_i(t+1) \tag{5}$$

The PSO algorithm is used to find the minimum or maximum values (optimal solution) for a given fitness function $\psi(t)$, which is shown in Algorithm 2.

Algorithm 2: PSO Algorithm

```
1  for each particle do
2      initialize particle position and velocity
3  end
4  while maximal number of iterations is not reached do
5      for each particle do
6          Calculate fitness value $\psi(t)$
7          if $\psi(t)$ is better than $\psi(b_i(t))$ then
8              $\psi(b_i(t))$ = $\psi((t))$
9          end
10         if $\psi(b_i(t))$ is better than $\psi(g(t))$ then
11             $\psi(g(t))$ = $\psi(b_i(t))$
12         end
13         Update particle velocity using Eq. (4)
14         Update particle position using Eq. (5)
15     end
16 end
```

C. Artificial Bee Colony

This algorithm is inspired from bee colony behavior and has been simulated for scientific and engineering tasks by Karaboga in 2005. In a real bee colony the food source and its quality are most important for bees. Applied to optimization problems, this bee behavior is translated into an algorithm that assumes the food source as a possible solution, and the quality of the food is represented as the fitness value in order to determine the optimal solution. Artificial Bee Colony in machine learning use three types of bees, that are employed bees, onlooker bees, and scout bees. There are some conditions in the algorithm implementation such as the number of employed bees and onlooker bees is the same, the bee type can be changed from one to another, and the bee initialization is random.

The algorithm consists of four phases:

1) Initialization Phase: In this phase, the initial population of all bees is initialized randomly. The control parameters are configured as per problem type and each solution or bee X_m and its vectors X_{mi} is initialized as well. m represents the population size or the number of bees, while i is the number of vectors or the problem dimension. In addition, the initial fitness $\psi(t)$ of the population is calculated.

2) Employed Bees Phase: In this phase, each employed bee begins from its allocated location $\vec{x_m}$ assigned in the initial phase, and starts searching for new food sources $\vec{y_m}$ having more honey in nearby sources. Nearby sources can be determined using a random vector X_{ki} selected from vectors in X_{mi}, and random number ϕ_{mi} in range $[-1, 1]$, the new source equation is as in Equation (6).

$$y_{mi} = x_{mi} + \phi_{mi}(x_{mi} - x_{ki}) \tag{6}$$

After the new source is found (possible solution), the bees compare the nectar amount of a food source that corresponds to the quality (fitness) of the associated solution using Equation (7).

$$f_i(t_i) = \frac{1}{1+\psi_i} \tag{7}$$

This phase is concluded after the employed bees return to the beehive with the location's of highest fitness (nectar) only.

3) Onlooker Bees Phase: After all employed bees complete the search process, they share the nectar information of the food sources and their position information with the onlooker bees. The onlooker bees evaluate the nectar information and choose a food source with a probability related to its nectar amount. As is the case for the employed bee, onlooker bees memorize the new position and forget the old one if the new position has a higher nectar amount. An artificial onlooker bee chooses a food source depending on a probability value p_i that is associated with that food source. This is calculated by Equation (8):

$$p_i = \frac{f_i(t_i)}{\sum_{n=1}^{SN} f_i(t_n)} \tag{8}$$

4) Scout Bees Phase: The second class of unemployed bees whose food sources are created randomly are called scouts. Scouts are used in two scenarios in ABC: (1) at the time of creation of the initial population, (2) when there exists an employed bee whose solutions cannot be improved through a predetermined number of trials, as specified by the user of the ABC algorithm. Algorithm 3 outlines the ABC algorithm.

D. Hybrid Algorithms

Swarm intelligence (SI) have proven to give more accurate results than Kmeans, however it comes at the cost of a higher running time as outlined in [7]. Swarm algorithms performance for clustering problems can be improved by seeding the initial swarm with the result of the Kmeans algorithm. The hybrid algorithm starts with Kmeans until it converges or reaches the

Algorithm 3: ABC Algorithm

```
1  Generate Initial population $x_i$, $i = 1, \dots, SN$
2  Evaluate the fitness($\psi(t)$) of population
3  while maximal number of iterations is not reached do
4      for each employed bee do
5          Produce new Solution $y_{mi}$ using Eq. (6)
6          Calculate fitness value $\psi(t)$
7          Apply greedy Selection process
8      end
9      Calculate probability value using Eq. (8)
10     for each Onlooker bee do
11         Select solution depend on $pi$
12         Produce new Solution $y_{mi}$ using Eq. (6)
13         Calculate fitness value $\psi(t)$
14         Apply greedy Selection process
15     end
16     if there is abandoned solution then
17         Replace it with new solution from Scout bee
18     end
19 end
20 return best solution in Onlooker bee
```

stopping criterion such as the number of iterations, then the best solution which are the centroids for clustering.

SI has been employed successfully for solving a variety of optimization problems including many multifaceted problems, where other popular methods like steepest descent, gradient descent, conjugate gradient, Newton method and others do not give satisfactory results [24].

In context of PSO based classification data set X with C classes and D attributes, this problems can be applied as searching for the optimal positions for C centroids of the data clusters in a D-dimensional space with the labeled samples. Each i^{th} element in X can be represented with respect to all centroid values such as:

$$i = \{p_i^1, ..., p_i^C, v_i^1, ..., v_i^C\} \tag{9}$$

(j^{th}) centroid position for (i) element represented using all vectors (N) as:

$$p_i^j = \{p_{1,i}^j, ..., p_{N,i}^j\} \tag{10}$$

Also, the velocity of the (J^{th}) centroid represented with respect to all dimensions is:

$$v_i^j = \{v_{1,i}^j, ..., v_{N,i}^j\} \tag{11}$$

Based on this encoding [29] each individual element has $K \times C \times N$ components.

The fitness function plays an important role in PSO. A good fitness function can quickly find the optimization positions of the particles. The classical PSO classification method is computed as the sum of the Euclidean distances between all the training samples and the class centroids encoded in the particle they belong to. Then, the sum is divided by N, which is the total number of training samples. The fitness of the i^{th} particle is defined as:

$$\psi(p_i) = \frac{1}{N} \times \sum_{j=1}^{N} d(x_j, P_{CL(j),i}) \tag{12}$$

In Equation (12), $d()$ is the Euclidean distances as in Equation (1), x_j is the training sample, and $P_{CL(j),i}$ is the class centroid of x_j.

The fitness function in Equation (12) is designed for classification since it does not utilize unlabeled data in any way, for this, [28] modified the common fitness function to use unlabeled data as well. In SSL, unlabeled data represents high percentage of the data set, the new fitness function that is used in this paper is:

$$\psi(p_i) = \beta\left[(\frac{1}{l} \times \sum_{j=1}^{l} d(x_j, P_{CL(j),i})\right] + \\ (1-\beta) \times \left[\frac{1}{u} \times \sum_{x=1}^{u} \min(d(x_k, P_{1,i}), ..., d(x_k, P_{1,c}))\right]$$

In the modified function, β is the ratio between label l and unlabeled u data, its value ranging between $[0, 1]$, the minimum function chooses the nearest centroid for unlabeled elements from all centroids. Please note that when β is 1, which means all data is labeled and this function is the normal PSO equation for classification, on the other hand if β is 0 its the one applied to clustering problems.

IV. Experimental Measures and Data Set

In this section, the evaluation metric is defined, then a brief introduction of the data set used for the experiments is given, and finally the result of the experiments are presented.

A. Evaluation Metric

The following are the evaluation measures used to evaluate the algorithms for the clustering problem, and more details are available in [25].

1) Loss Value: The loss value of a clustering algorithm is based on the true label. For each instant of the data set, the predicted cluster is compared with the original one based on the truth label, and returns one if it is the same or zero otherwise. The loss value can be computed as:

$$Loss = \frac{wrong\ prediction}{dataset\ length} \tag{13}$$

For this measure, zero is the optimal result and one is the worst result.

2) Adjusted Mutual Info Score: The adjusted mutal info score is an adjusted version of the Mutual Information (MI) score measure that is calculated between two variables and measures the reduction in uncertainty for one variable given a known value of the other variable [30] as "a quantity called mutual information measures the amount of information one can obtain from one random variable given another".

	Best Value	Kmeans	PSO	PSOH	ABC	ABCH
Loss score	0	0.26	0.38	0.23	0.23	**0.17**
Adjusted mutual info score	1	0.23	0.40	**0.65**	0.38	0.63
Homogeneity Score	1	0.24	0.41	0.42	0.39	**0.46**
Completeness score	1	0.23	0.39	0.59	0.42	**0.67**
V-measure score	1	0.23	0.40	0.56	0.39	**0.63**
Davies bouldin score	0	**0.46**	0.83	0.61	0.76	0.54
Sum squared error	0	51.0%	51.0%	57.0%	65.0%	**81.0%**
Quantization error	0	**78.5%**	45.0%	53.0%	52.0%	71.0%
Execution Time (minutes)	0	**24**	87	112	125	143

TABLE I: Results of Experiments

3) Sum Squared Error: This is the sum of the squared differences between each instance x_i and its group's centroid c_i. Identical results are zero but this is usually not achievable on real world problems. This value has been converted to percentage since each algorithm will start randomly, therefor SSE values are different. The number presented is the reduced percentage from the first value; this means if the value is 100% this means that SSE is 0, which is the best result.

$$SSE = \sum_{i=1}^{n} (x_i - c_i)^2 \tag{14}$$

4) Quantization Error: The quantization error is the mean of distance between the absolute value of the centroid and each cluster points, the identical value is zero.The values are converted to percentage as well as for SSE.

$$QE = \frac{\sum_{i=1}^{n} |(x_i - c_i)|}{n} \tag{15}$$

5) Homogeneity Score: The homogeneity of a cluster is achieved if each cluster has the instance that belongs to it as given in the true table. Different labels or permutations will not affect the results. The maximum value that can be achieved is one, which means perfect homogeneous labeling whereas zero is the worst value.

6) Completeness Score: A clustering result satisfies completeness if each cluster contains data points that are members of one class as is given in the true table. A value of one means perfect and complete labeling.

Homogeneity and completeness are not the same and measure a different relation. For example, if the truth table has one cluster only while the algorithm returns more than one, it is considered as perfect homogeneity but not complete since not all instances of the same class are in one cluster. On the other hand, if the true label table has more than one cluster but the algorithm returns one cluster it is perfect completeness since all instances of the same class are in one cluster.

7) V-measure Score: The V-measure is an entropy-based measure, which explicitly measures how successful the criteria of homogeneity and completeness have been satisfied. The V-measure is computed as the harmonic mean of the distinct homogeneity and completeness scores [26].

$$v = \frac{1 + \beta \times homogeneity \times completeness}{\beta \times homogeneity + completeness} \tag{16}$$

The main advantages of the V-measure are its bounded scores between 0 and 1, where 1 is the best value, as well as its intuitive interpretation for homogeneity and completeness.

8) Calinski Harabasz Score: The index is the ratio of the sum of between-clusters dispersion and of inter-cluster dispersion for all clusters (where dispersion is defined as the sum of distances squared). This index is higher when clusters are dense and well separated.

9) Davies Bouldin Score: This index signifies the average 'similarity' between clusters, where the similarity is a measure that compares the distance between clusters with the size of the clusters themselves. Zero is the lowest possible score. Values closer to zero indicate a better partition.

B. Data Set

The CIRA-CIC-DoHBrw-2020 [27] data set is collected using a two-layered approach to capture benign and malicious DoH traffic together with non-DoH traffic. The first layer classifies the traffic as DoH and non-DoH using the statistical features model. At the second layer, DoH traffic is assigned to the DoH type, benign or malicious using a time-series classifier. This data set contains more than 1 million records with 28 statistical features. The experiments applied in this paper was to cluster traffic as DoH or not only.

V. Results and Discussion

This section presents the results of the experiments that compare the different algorithms using the DoHBrw data set. The different algorithms are Kmeans, ABC, PSO, and the hybrid approaches consisting of ABC and KMeans abbreviated as ABCH, and the hybrid of PSO with Kmeans abbreviated as PSOH.

The results of PSO and ABC were very similar, however, ABC performed slightly better based on some measures. In terms of the running time, as shown in Table I, the hybrid algorithms consume more time than the SI algorithms; PSOH's time was faster than ABCH's, and the running time of PSO was less than ABC. However, as expected Kmeans was the fastest algorithm completed the process in 24 minutes.

As shown in Figure 2, Hybrid ABC was the best algorithm in terms of accuracy, its loss value is only 17%, which means few data points were clustered incorrectly. Kmeans being the basic algorithm of clustering scored 26%, the swarm algorithms scores were 38% for PSO and 23% for ABC. For the hybrid algorithms, the loss value decreased by 15% for PSO and 6% for ABC, which is quite an improvement.

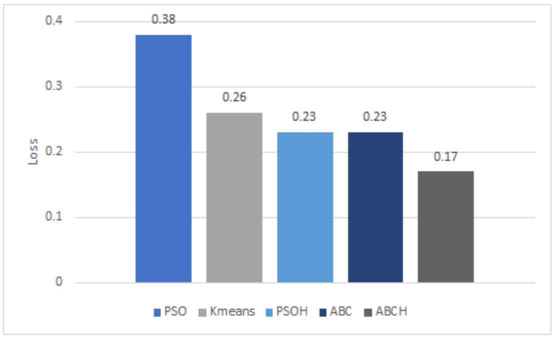


Fig. 2: Loss Value

Figure 3 presents the AMI scores that measure the reduction in uncertainty. The results show PSOH with the highest value of 0.65, followed by ABCH with a score of 0.63. The SI algorithms performed better than Kmeans by around 0.15.

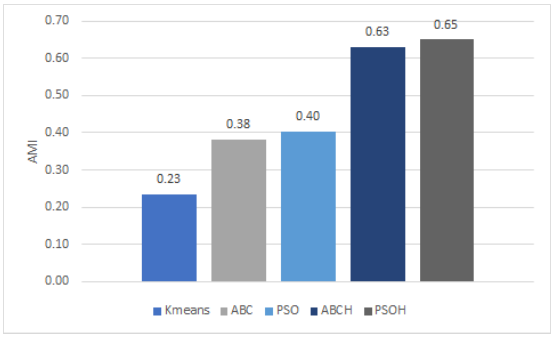


Fig. 3: AMI

Figure 4 shows the results of homogeneity, completeness, and the V-measure. ABCH performed better than the others with values of 0.46, 0.67 and 0.63, respectively, and PSOH with similar values 0.42, 0.59 and 0.56. PSO has better values than ABC, and both results are much better compared to Kmeans only achieving 0.23, 0.23 and 0.24. The detailed results can be seen in Table I.

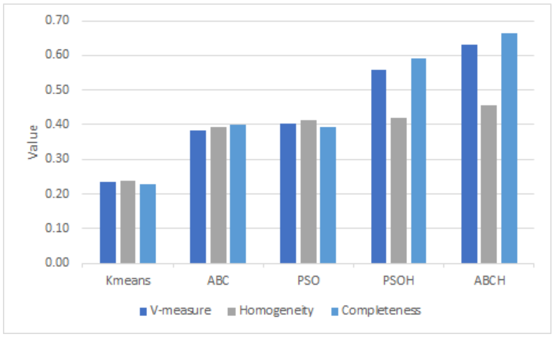


Fig. 4: V-Measure, Homogeneity and Completeness

The Davies bouldin score is used to measure the clusters' similarity for each algorithm. Perfect clustering returns identical clusters. As shown in Table I, the clusters produced by Kmeans were the most different with 0.46, while the SI algorithms did not perform well as did the hybrid ones were the average result was around 0.55.

Figure 5 shows how the SSE changes during the clustering process. SSE measures the square distance between the elements in the cluster and its centroid. Since this number can get large it is converted to percentage, which means 100% in the first iteration, and the worst distance with all elements shows the difference to the optimal solution values. Almost all results were in same range with a small improvement for PSOH where the distance was reduced by 6% more than PSO. ABC's value was reduced more but still within the range of around 65%, while hybrid ABC was able to reduce the distance by more than 80%.

Figure 5a shows Kmeans reduced the distance faster than the others, ABC shown in Figure 5d with ABCH 5e the distance fluctuated until the value reached the minimum for SSE. ABC and its hybrid version as shown in the figure show consistent improvement until the minimum is reached in later iterations.

TABLE II: Classification Results of Supervised Learning Algorithms

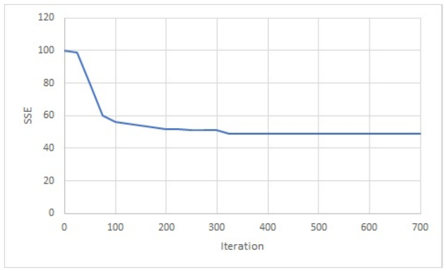
	Accuracy	Precision	Recall	AUC
Decision Tree	99.8%	99.8%	99.8%	99.8%
Extra Tree	99.5%	99.4%	99.6%	99.9%
Gradient Boosting	99.9%	99.8%	100.0%	100.0%
LGBM	100.0%	99.9%	100.0%	100.0%
XGBoost	100.0%	99.9%	100.0%	100.0%
Random Fores	99.8%	99.7%	99.9%	100.0%

Even though this paper deals with semi supervised clustering, for completion we are listing supervised clustering results for the data set investigated as well. Table II summarizes the results from [19] that used the same data set however using supervised learning algorithms. The achieved ML classification results are impressive, however, these results are based on a testing data size of 4,000 records only when the data set contains more than 1 million records. LGBM and XGBoost outperform the another algorithms with an accuracy of 100%, precision of 99.9%, recall and AUC scores of 100%, respectively. Comparing the accuracy with semi supervised learning achieved around 83%. It is normal that classification using supervised learning achieves better results than semi supervised learning approaches.

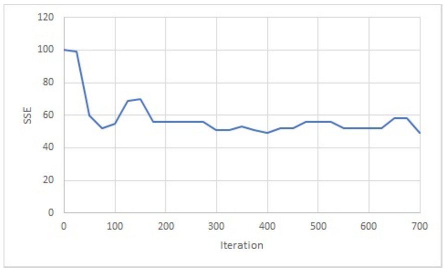
Supervised learning algorithms use usually a very large amount of the data for training, and then performing testing on other part of data, which was also done on the data set that is investigated in this paper. Semi supervised learning algorithms that were used in this paper only uses a small amount of labeled data and extracted the data structure from the unlabeled data to perform the classification or clustering task and thus the results are very impressive. Thus, if only a small amount of data is available semi supervised learning algorithms do very well.

VI. Conclusion

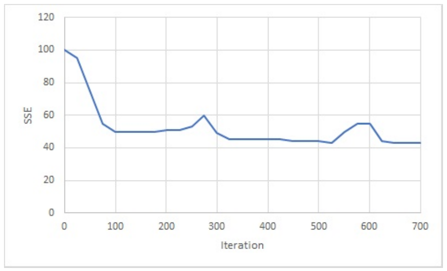
The main objective of this paper was to use swarm intelligence algorithms and two hybrid versions applied to the



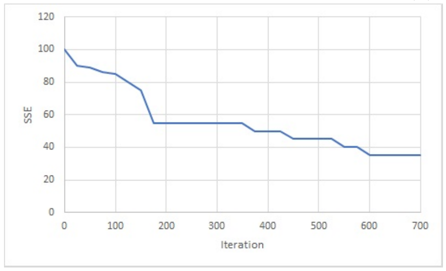
(a) Kmeans



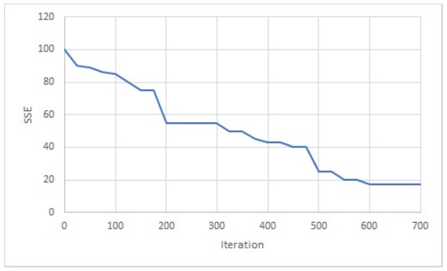
(b) PSO



(c) PSOH



(d) ABC



(e) ABCH

Fig. 5: SSE

clustering problem in semi supervised learning. The algorithms were compared with the well-known clustering algorithm kmeans. To investigate the algorithms a large data set was used with more than 1 million records and 28 features. The data set chosen was a cybersecurity data set that classifies domain name server traffic distinguishing whether https is used or not. Different evaluation measures were used to compare the algorithms. The results of the experiments are that swarm intelligence algorithms by itself are able to solve the clustering problems achieving very good results. Furthermore, the combination of SI algorithms with standard clustering Kmeans can result in significant improvements. However, the increase in clustering accuracy comes at the cost of a higher execution time.

References

[1] N. Chatzis, Motivation for behaviour-based DNS security: A taxonomy of DNS-related internet threats, in SECUREWARE, Valencia, Spain, 2007, pp. 36-41.
[2] A. Ahmim, N. G. Zine, A new hierarchical intrusion detection system based on a binary tree of classifiers, Information & Computer Security, Mar. 2015.
[3] S. Detrow, Obama on Russian Hacking: We Need to Take Action. And We Will, NPR News, 2016.
[4] R. Langner, Stuxnet: Dissecting a cyberwarfare weapon, IEEE Security & Privacy, 2011, pp. 49-51.
[5] I. Bouteraa, M. Derdour, A. Ahmim, Intrusion Detection using Data Mining: A contemporary comparative study, 3rd International Conference on Pattern Analysis and Intelligent Systems (PAIS), IEEE, 2018, pp. 1-8.
[6] H. J. Liao, C. H. R. Lin, Y. C. Lin, K. Y. Tung, Intrusion detection system: A comprehensive review, Journal of Network and Computer Applications, 2013, vol.36(1), pp. 16-24.
[7] C. F. Tsai, Y. F. Hsu, C. Y. Lin, W. Y. Lin, Intrusion detection by machine learning: A review, expert systems with applications,2009, vol. 36(10), pp. 11994-12000.
[8] W. Li, Q. Li, Using naive Bayes with AdaBoost to enhance network anomaly intrusion detection, Third International Conference on Intelligent Networks and Intelligent Systems, IEEE, 2010, pp. 486-489.
[9] F. Ertam, L. F. Kilincer, O. Yaman, Intrusion detection in computer networks via machine learning algorithms, International Artificial Intelligence and Data Processing Symposium (IDAP), IEEE, 2017, pp. 1-4.
[10] S. K. Gautam, H. Om, Computational neural network regression model for Host based Intrusion Detection System, Perspectives in Science, 2016, 8, pp. 93-95.
[11] J. Jha, L. Ragha, Intrusion detection system using support vector machine, International Journal of Applied Information Systems (IJAIS), 2013, vol. 3, pp. 25-30.
[12] G. Liu, Z. Yi, S. Yang, A hierarchical intrusion detection model based on the PCA neural networks, Neurocomputing, 2007, vol. 70 (7-9), pp. 1561-1568.
[13] J. Zhang, M. Zulkernine, A. Haque, Random-forests-based network intrusion detection systems, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2008, vol. 38(5), pp. 649-659.
[14] J. Alzubi, A. Nayyar, A. Kumar, Machine Learning from Theory to Algorithms: An Overview, Journal of Physics: Conference Series, IOP Publishing, 11/2018, vol.1142, pp. 12012.
[15] J. Xu, K. Lange, Power k-Means Clustering, Proceedings of the 36th International Conference on Machine Learning, PMLR, 2019,vol. 97, pp. 6921-6931.
[16] S. Wang, K-Means Clustering With Incomplete Data, in IEEE Access, 2019, vol. 7, pp. 69162-69171, doi: 10.1109/ACCESS.2019.2910287.
[17] J. Cai, H. Wei, H. Yang, X. Zhao, A Novel Clustering Algorithm Based on DPC and PSO, in IEEE Access, 2020, vol. 8, pp. 88200-88214, doi: 10.1109/ACCESS.2020.2992903.
[18] F. Zabihi, B. Nasiri, A Novel History-driven Artificial Bee Colony Algorithm for Data Clustering, Applied Soft Computing, 2018, vol. 71, pp. 226-241, ISSN 1568-4946.
[19] B. Yaser, R. Steve, Detecting Malicious DNS over HTTPS Traffic in Domain Name System using Machine Learning Classifiers, in Journal of Computer Sciences and Applications, 2020, vol.8, pp. 46-55.
[20] Y. Kevin, C. Chao, G. Mark, Machine learning and genome annotation: A match meant to be?, in Genome biology, 2013.
[21] S. Na, L. Xumin, G. Yong, Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm, 2010 Third International Symposium on Intelligent Information Technology and Security Informatics, Jinggangshan, 2010, pp. 63-67, doi: 10.1109/IITSI.2010.74.
[22] M. Fateme, H. Abdorrahman, A novel hybrid wrapper-filter approach based on genetic algorithm, particle swarm optimization for feature subset selection, in Journal of Ambient Intelligence and Humanized Computing, 2020.
[23] D. Kumar, K. K. Mishra,Co-variance guided Artificial Bee Colony, Applied Soft Computing, 2018, vol. 70, pp. 86-107, ISSN 1568-4946.

[24] J. Gourhari, M. Arka, P. Sudip, S. Shamik, C. Pratim, Modified Particle Swarm Optimization Algorithms for the Generation of Stable Structures of Carbon Clusters JFrontiers in Chemistry, 2019, vol. 7, pp. 485, ISSN 2296-2646.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 2011, vol. 12, pp. 2825-2830.

[26] A. Rosenberg, J. Hirschberg, V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure, 2007, pp. 410-420.

[27] M. MontazeriShatoori, L. Davidson, G. Kaur, A. Habibi Lashkari, Detection of DoH Tunnels using Time-series Classification of Encrypted Traffic, The 5th IEEE Cyber Science and Technology Congress, Calgary, Canada, August 2020.

[28] V. A. Jour, Z. Xiangrong, J. Licheng, P. Anand, Y. Yongfu, W. Zhengli, A. Qiang, Semisupervised Particle Swarm Optimization for Classification, 2014.

[29] F. Ivanoe, D. Cioppa Antonio, T. Ernesto, Evaluation of Particle Swarm Optimization Effectiveness in Classification, 2005, pp. 164-171.

[30] H. I. Witten, E. Frank, M. A. Hall, C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques (Morgan Kaufmann Series in Data Management Systems) 4th Edition, 2016.