2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Web Bots Detection Using Particle Swarm Optimization Based Clustering

Shafiq Alam, Gillian Dobbie, Yun Sing Koh, Patricia Riddle
Department of Computer Science,
University of Auckland, New Zealand.
sala038@aucklanduni.ac.nz, {gill, pat, ykoh}@cs.auckland.ac.nz

***Abstract*—Optimization based techniques have emerged as important methods to tackle the problems of efficiency and accuracy in data mining. One of the current application areas is outlier detection that has not been fully explored yet but has enormous potential. Web bots are an example of outliers, which can be found in the web usage analysis process. Web bot requests are different from a genuine web user as web bots crawl large numbers of pages in a very short time. If web bots remains undetected they can skew the analysis process which can result in incorrect patterns that can cause wrong decisions. In this paper we use one of the popular Swarm Intelligence (SI) based techniques called Particle Swarm Optimization (PSO) to detect web bots among genuine user requests. We use our Particle Swarm Optimization (PSO) based clustering algorithm, Hierarchical Particle Swarm Optimization based clustering (HPSO-clustering) to cluster the web-usage data and detect the abnormal behaviour caused by the web bots. We present the results of detection which shows that our proposed approach is capable of detecting such abnormal behaviour. We then compare the characteristics of the detected web bots with genuine web-users using cross validation.**

Keywords: Particle Swarm Optimization, Data clustering, web bots, Web usage mining, evolutionary computation

I. Introduction

Outlier detection has become one of the major research areas in knowledge discovery and data mining. Traditional data mining techniques focus on finding knowledge such as frequent patterns, clusters of data, classification and association rules. Outlier detection, however, focuses on finding patterns, which show deviations from the normal data and reveal abnormalities in the data [1]. Outliers are observations that deviate significantly from the normal data and represent a very small fraction of the data [2]. Any noise added to the data which has sufficient deviation from the rest of the data of the repository can skew the analysis process substantially. An example of such observations are the sessions from web bots in web data logs, which sufficiently deviate from normal web user sessions. Because web bots are outliers, they can be excluded from further analysis, as they do not represent genuine web user behaviour. Such outliers may lead to wrong analysis and hence to a wrong prediction which can result in incorrect decisions. The study of outliers can be helpful in modelling the decision making process.

The web usage data is divided into two types, explicitly collected web usage data, and implicitly collected web usage data. In explicit web usage data the data is collected with the consensus of the web user. The web users actively participate in generating the data by specifying their likes and dislikes about specific groups of web resources. From a system perspective, the users are known so we can easily separate them from each other. To collect data implicitly the user does not actively participate in the data gathering process, hence remains anonymous, and it is not easy to separate users from each others. Building applications based on patterns generated from implicitly collected data has two major issues. Firstly, the data may be of poor quality and require sophisticated analysis and preprocessing before it can be used for pattern extraction. Secondly, some of the web crawlers are not detectable in the traditional data cleaning phase as they are not known in advance.

Generally implicit web usage data does not provide any information about the identity of the web user because most of the data is generated by anonymous users. While building any systems based on such data, there is a danger that the noise and the data generated by web bots will skew the preprocessing and pattern extraction process. As far as the web bot requests are concerned there are a number of resources which provide lists of those IP's/hosts which are used by the known crawlers, i.e. http://www.iplists.com/. However, there are two problems with these lists. The first problem is that most of the web bots' lists have not been updated over the last few years and there may be new hosts which are crawling the web that are not included in the lists. The second problem is the number of bad bots which hide their identity. For these reasons, there is a need to develop techniques that detect such requests on the basis of their features rather than relying on the provided list. Outlier detection is one such technique which can tackle this problem.

The contributions of this paper include (1) Modelling web usage data for web bot detection, (2) detection of web bots from implicit web usage data, and (3) assessing the use of HPSO-clustering for web bot detection on real world data. Like all other web usage data, the web usage data, which will be used for our analysis, has web bots, without treating them would skew the clustering process. We evaluated the accuracy of the detection method based on the difference of detected outliers' behaviour compared to the genuine users, and traditional precision and recall measure after injecting web bots into the data. The rest of the paper is organised as follow. Section II outlines the related work, Section III presents the

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

proposed outlier detection method, and Section IV presents the experimental details and their results. Section V discusses the outcomes of the experimentation while the last section concludes the paper with a summary and recommendations.

II. Related Work

There are a number of supervised, semi-supervised, and unsupervised outlier detection techniques that can be used for the detection of abnormal observations in web usage data. There are no particular techniques that have been designed specifically for web bot detection. Traditional statistical techniques are univariate, looking at the distribution of the data for a particular attribute and examine the degree of legitimacy for an attribute value to be an outlier. Distance based techniques are multivariate, and find anomalies using clustering or density based approaches.

a) **Statistical outlier detection methods::** The statistical outlier detection process targets the distribution of the data, the distribution of parameters (whether they are known or unknown), the number of expected outliers, and the types of expected outliers. Boxplot plot and histogram methods fall into univariate outlier detection methods while some common statistical measures such as mean, standard deviation, and variance can also help find outliers in the data. Distribution based outlier detection techniques includes Control Chart Technique (CCT) and Linear Regression Technique (LRT). Statistical tests such as Grubb's test, and Dixon's test can also be applied to isolate suspected observations from the genuine data. The main problems with these approaches include considering only univariate data and its dependence on distribution. Such techniques are used in [2] and [3] where they give a definition to an outlier observation.

b) **Density based outlier detection::** In Density based outlier detection methods, whether an observation is an outlier is based on its neighbourhood. An observation is considered an outlier if it does not have many data points around it. In [1] an observation is considered an outlier based on a factor called Local Outlier Factor (LOF). This factor looks purely at the local neighbourhood of the data point and implements a fuzzy approach while assigning an observation a degree of outlierness, rather than using a rule of logic. Due to the local outlier factor, this technique is only capable of detecting single point outliers and can't detect a cluster of outliers [4]. A similar approach is proposed in [5], where an outlier is detected based on the density of the neighbourhood. However, instead of calculating the LOF for all outliers, they compute it for the top K-outliers only. The authors in [6] suggested the use of frequent pattern mining based outlier detection. The approach uses a parameter called Frequent Pattern Outlier Factor (FPOF).

c) **Distance based outlier detection::** Distance-based outlier detection methods have been effective in detecting individual as well as clusters of outliers. One of the most commonly used distance based techniques is [7], which considers a data point as an outlier based on a threshold distance from the rest of the data. It can be applied to univariate data as well as multivariate data. A distance based outlier detection mechanism is described in [8], where a subset of the data set is used as an outlier detection solving set. Another similar approach is applied to detect an outlier in [9] based on its k-nearest neighbours and ranking it on the same distance measure. In [10] random order with simple pruning rules isolate the outlier from the genuine data. The overall performance of this algorithm is good. A number of hybrid distance based approaches have also been introduced which combine different measure such as distances and densities, and partition the data into different isolated groups.

d) **Clustering based outlier detection::** Some data mining techniques are used for dual purposes. For instance clustering based techniques can be used to detect a cluster of outliers in the data if the outliers have similar characteristics. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [11], Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) [12], Robust clustering algorithm for categorical attributes (ROCK) [13], and Clustering Large Application based on RANdomised Search (CLARANS) [14] are some of the clustering algorithms that can be used for detecting outliers in the data. ROCK uses links to isolate outliers from genuine numerical, categorical and Boolean attributes. CLARANS combines PAM [15] and CLARA [16], and uses a random search for finding different clusters using k-medoids. The k-medoids based methods are very robust when detecting the existence of outliers. BIRCH extracts nested clustering structures by concentrating on densely populated portions and captures the natural closeness of data while separating sparse data as outliers. Some other distance based approaches including [17], [18], [4], LDBSCAN [19], and OPTICS [20] have also used clustering approaches to partition the data and isolate outliers from the data.

Clustering techniques such as ROCK, BIRCH, DBSCAN, and CLARNS are efficient, but are not optimised for outlier detection, so outliers are always regarded as noise and a bi-product. The use of such techniques for outlier detection is not efficient [18].

e) **Machine learning and computational intelligence based outlier detection::** Some of the recent outlier detection techniques take their inspiration from machine learning and computational intelligence, which provide a way to improve the efficiency of the detection process by involving a self learning mechanism and optimization.

In [21] the authors propose a particle swarm optimization based clustering approach for outlier detection called Hierarchical Particle Swarm Optimization Based Clustering (HPSO-clustering). It performs clustering in a hierarchical manner as well as detecting outliers by merging different particles during successive generations of the swarm evolution. The results showed an improvement on the state of the art outlier detection methods. FindOut described in [22] uses wavelet transformation to isolate clusters from the data and then find outliers. An evolutionary search technique was described in [23] on high dimensional data and detected suspected outliers. A replicator neural network based outlier detection method was proposed in [24], which classifies the data and detects the outliers. Another recent work, which uses PSO for the first time is proposed in [25] where Euclidean distance is used as a separation measure. The work highlights the efficiency and performance of the proposed technique, which is better than LOF based methods.

There is no general method to evaluate outlier detection methods without having a ground truth, however a method called rare class detection is used to evaluate the performance of outlier detection algorithms. In rare class detection methods an imbalance class of data is formed by manipulating the original data and an algorithm is trained and tested on that data [26].

The next section highlights our proposed approach and the evolution criteria we used to measure the performance of the approach. The proposed HPSO-Clustering Based Outlier Detection mechanism is capable of doing both clustering and outlier detection at the same time.

III. The Proposed Outlier Detection Technique

Hierarchical Particle Swarm Optimization based clustering has been successfully used for clustering as shown in [27], [28], and [21]. Improved accuracy, efficiency, and simplicity of these techniques make it extendable and adoptable for most data mining applications. Our proposed HPSO-clustering approach performs clustering in a hierarchical agglomerative manner and provides a clustering solution by partitioning the data into tiny clusters and then merging the clusters to form a hierarchy of clusters. Overall performance of this approach is better than hierarchical agglomerative clustering in terms of accuracy and efficiency. Our proposed technique has two modules, a clustering module and an outlier detection module. The clustering module performs clustering in a hierarchical agglomerative manner while the outlier module works simultaneously to label the suspected outliers and remove them from the succeeding hierarchy of clusters. Below we briefly discuss how HPSO-clustering works, followed by the way it detects outliers in the data. Additional details of the technique and experimentation on benchmark UCI machine learning datasets are given in [27] and [21].

HPSO-Clustering: HPSO-clustering partitions the data into small clusters and then merges these small clusters based on the learning of the particles of the swarm. The technique exploits the learning components of Particle Swarm Optimization (PSO) to initially partition the data into clusters as well as merge the smaller clusters into larger clusters. It uses cognitive and self organising components of the swarm to move the centroids of the clusters to better positions. One particle represents one centroid of a cluster. Equation 1 shows how the particle moves from one position to a better position within the cluster.

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

where $X_i(t+1)$ is the new position of the particle, $X_i(t)$ is the previous position of the particle, and $Vel_i(t+1)$ is the new velocity that is obtained by the cognitive and self organising components of the swarm. Equation 2 calculates the velocity of the particle for equation 1.

$$Vel_i(t+1) = \omega \times Vel_i(t) + q_1 r_1(pBest_i(t) - X_i(t)) + q_2 r_2(Y_i(t) - X_i(t)) \tag{2}$$

where $Vel_i(t)$ represent the current velocity, $Vel_i(t+1)$ is the new velocity, $pBest_i(t) - X_i(t)$ is the cognitive learning component, while $Y_i(t) - X_i(t)$ is the self-organising component of the swarm. The **cognitive** component is the learning of the particle from its own experience. The particle records its best ever position in a temporary variable called $pBest$, which gets updated if the particle finds a better position. The self-organising component of the swarm represents the current configuration of the data points in the clusters. This component helps the particle remain inspired by the configuration of his corresponding clusters.

In each generation of a swarm, the particles learn from their experience, move to a better position and merge based on the population of clusters. Lower populated clusters are merged to the nearest higher populated clusters. The swarm then evolves to the next generation and the same process of refining the position of the centroids is performed until the required hierarchy of clusters is achieved. Algorithm 1 shows the clustering module of the HPSO-clustering based outlier detection algorithm.

HPSO-Clustering and outlier detection: The outlier detection module triggers before entering into the next generation of the swarm. Before merging a less populated cluster, a distance threshold is used to identify whether a particular cluster is a group of outliers or a genuine cluster. The distance threshold depends on the application as well as the configuration of data. We relate this distance measure to average intra-cluster distance and maximum intra-cluster distance. Equation 3 shows how the threshold distance based on average intra-cluster distance is calculated for a particular particle.

$$ThreshDist(X_i) = \frac{D_t}{k} \times \sqrt{\sum_{j=1}^{k}(Y_j - X_i)^2} \tag{3}$$

where Y_j represents a data vector and k is the number of total data associated with a particular particle X_i. Equation 4 shows how the threshold distance based on maximum intra-cluster distance is calculated for a particular particle.

$$ThreshDist(X_i) = D_t \times argMax_{i=0}^{n} \left\{ \sqrt{\sum_{j=1}^{k}(Y_j - X_i)^2} \right\} \tag{4}$$

Algorithm 1 presents the pseudocode of HPSO based outlier detection. The initial swarm is spread in the input data space and different parameters such as minimum and maximum velocities, D_t, value of cognitive component, and position of the swarm are initialized. The algorithm starts with clustering of data into different groups using the HPSO-clustering approach.

At the end of the first generation, the intra-cluster distance achieved by each cluster is calculated and used for finding the $ThreshDist(X_i)$ mentioned in line 17 of the algorithm. In each cluster if a data element is at a distance greater than $ThreshDist(X_i)$ it is marked as an outlier. This process finds each single outlier in a particular generation. The outlier can then be ranked on the basis of its distance from the centroid or its appearance as an outlier in the subsequent generations. The more frequently an observation appears in the outlier list, the higher its rank gets in the ranking list. To be marked as an outlier the observation must have a considerable distance from the centre of the cluster. The further the data element is from the centroid, the stronger is the chance of the data element being an outlier. In our experiment we do not rank outliers but generate all potential outliers.

There are some cases where there are clusters of outliers in the dataset. Such outlier clusters are not detected as a whole but in the consumption process, when consumed by a stronger particle, it becomes a part of that stronger particle. Obviously the centroid of the stronger particle will evolve to the most dense areas in the cluster and so the outliers will fall at a considerable distance i.e. more than $ThreshDist(X_i)$. In this way the clusters of outliers are also detected through merging and self organization of particles in successive generations.

IV. WEB BOT DETECTION

Experimental Setup: For our experiments we set the following parameters. The number of particles i.e Swarm Size =49, iterations per generation = 100 and number of generations = 50. All other HPSO-clustering parameters were set to default values.

We performed four different experiments: **(1)** detect outliers based on average intra-cluster distance, **(2)** detect outliers based on maximum intra-cluster distance, **(3)** detect outliers using the intersection of maximum and average intra-cluster distance, and lastly, **(4)** identify the synthetically injected web bots and evaluate using traditional precision and recall measures.

For our first experiment we chose different values of threshold distance D_t as a function of average intra-cluster distance as shown in equation 3, and record the number of outliers found in the log. Figure 1 shows the distribution of outliers at different values of the average intra-cluster distance. At larger values of D_t fewer web bots have been detected and vice versa. The right most part represents the large number of outliers found at smaller values of D_t i.e. $D_t = 8$.

Figure 2 shows the frequency of these web bots. The more frequently an observation is labelled as an outlier the higher is the chance that the observation is different from the rest of the data, hence a web bot. A threshold value can be set to separate the most prominent outliers from the less prominent ones.

For our second experiment we used maximum intra-cluster distance as a threshold distance D_t as shown in equation 4, and extracted the suspected web bots. Figure 3 shows the distribution of outliers at different values of the maximum intra-cluster distance. At smaller values of D_t fewer web bots have been detected and vice versa.

Algorithm 1 HPSO-Clustering based outlier detection

Input: Data file
Output: Outliers based on D_t
Parameters: Swarm Size S, V_{Max}, V_{Min}, ω,q_1,q_2, Distance D_t, and number of records N
Method:

```
1:  INITIALIZE $S$,$V_{Max}$, $V_{Min}$, $\omega$,$q_1$,$q_2$,$N$, and $D_t$
2:  for Each Particle $X$ do
3:      INITIALIZE $X_i$
4:  end for
5:  while (STOPPING CRITERIA (false)) do
6:      for each generation do
7:          for each iteration do
8:              for Each Particle $X$ do
9:                  ASSIGN won data vectors to Particles
10:                 CALCULATE $pBest$ for each particle
11:                 if $pBest(t+1)$ is better then $pBest(t)$ then
12:                     $pBest(t) \leftarrow pBest(t+1)$
13:                 end if
14:                 CALCULATE Velocity $V_i(t)$
15:                 UPDATE Position $X_i(t)$
16:                 CALCULATE intra-cluster distance
17:                 CALCULATE $ThreshDist(X_i)$ for current Particle
18:                 if $intraclusterdistance > ThreshDist(X_i)$ then
19:                     FLAG data as potential outlier
20:                     ADD to the list of outlier
21:                     INCREMENT Outlier count
22:                 end if
23:             end for
24:             LIST all outliers
25:             RANK outliers
26:         end for
27:         CALCULATE swarm strength
28:         FIND the weakest Particle
29:         FIND the nearest strong Particle
30:         MERGE both Particle
31:         UPDATE $X_i(t)$
32:         DELETE weaker particle
33:     end for
34: end while
```

Figure 4 reports the frequency distribution which shows that there are a number of bots which have been detected only once while the minority of the bots have been detected frequently. Such bots are highly suspected web bots.

Table I shows the average and standard deviation of the three attributes, amount of downloaded data during a session, number of pages browsed, and time (in minutes) for each session. The results show that bots discovered based on larger values of maximum intra-cluster distance (highly confirmed

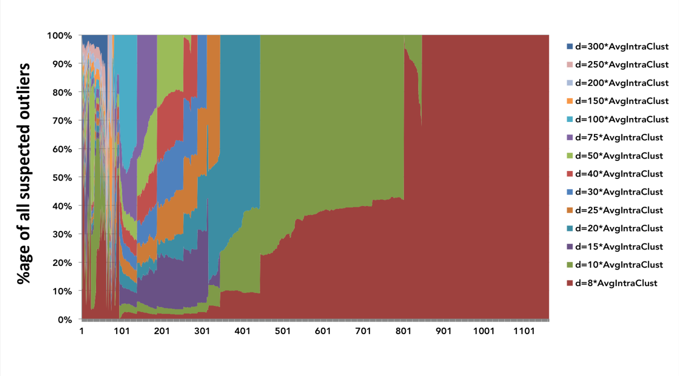

Fig. 1. Density of suspected outliers using Average Intra-cluster distance at different value of D_t

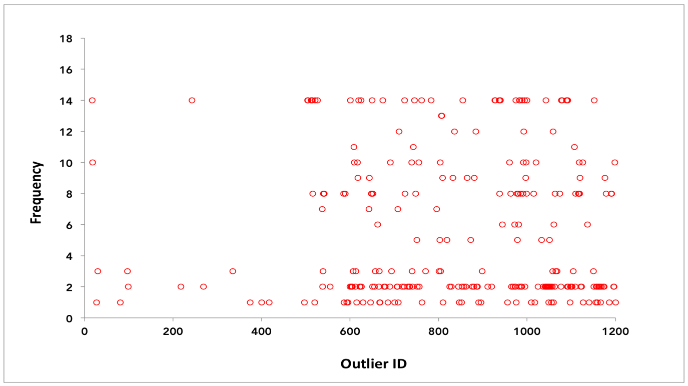

Fig. 2. Frequency of the detected bots based on avg. intra-cluster distance

web bots), download more data than those bots which are discovered based on lower values of max intra-cluster distance. We observe a similar relationship in the average number of pages browsed during a session. Web bots discovered having larger values of D_t often browse more pages as compared to the web bots which are less suspicious. The last column shows the amount of time that has been incurred during each session by web bots. The results show that the confirmed bots take less time to browse a large number of pages. This means the bots have very fast inter page transitions which is a typical behaviour for web bots.

TABLE I
AVERAGE AND STD. DEV. OF DIFFERENT ATTRIBUTES VALUES FOR THE DETECTED WEB BOTS BASED ON MAX. INTRA-CLUSTER MEASURE

Dt	**Data Download**		**Pages browsed**		**Time per session**	
	Avg.	**StdDev.**	**Avg.**	**StdDev.**	**Avg.**	**StdDev.**
0.0001	4562939.24	21137281.59	245	1122	45.97	17.99
0.0003	20816311.64	36492151.61	465	1974	46.36	17.18
0.0005	12112349.71	45205821.38	620	2447	45.18	17.58
0.0007	12741171.99	46572822.81	649	2523	45.11	17.67
0.0009	13188894.66	47743044.07	674	2586	44.98	17.61
0.001	13681160.45	101632846.29	692	5598	44.52	19.71
0.003	25299697.15	82690783.20	1569	4626	41.51	17.71
0.005	35543601.72	91630147.22	1821	5139	39.78	19.25
0.007	37782139.12	97883748.75	1986	5499	39.90	19.44
0.009	36537475.73	100666470.17	1792	5543	38.73	19.74
0.01	37248603.64	48899644.58	1826	2650	39.17	17.85
0.03	54063456.47	148188323.22	2980	8863	38.60	18.24
0.05	69038997.40	145603181.22	3538	8218	34.75	21.39
0.07	92737810.08	173092047.28	4906	9948	32.54	22.70
0.09	104376580.56	196071848.73	5588	11501	23.89	22.00

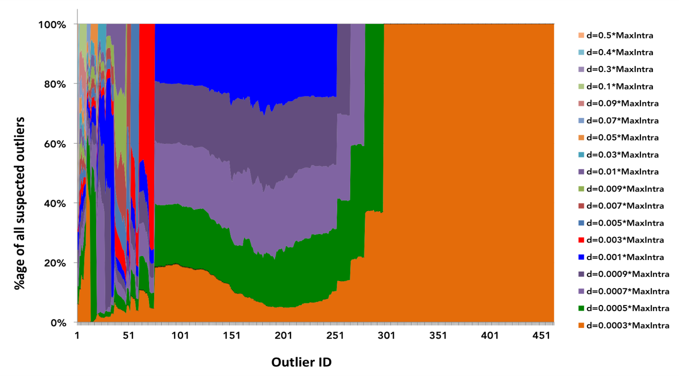

Fig. 3. Density of suspected outlier using Maximum Intra-cluster distance at different values of D_t

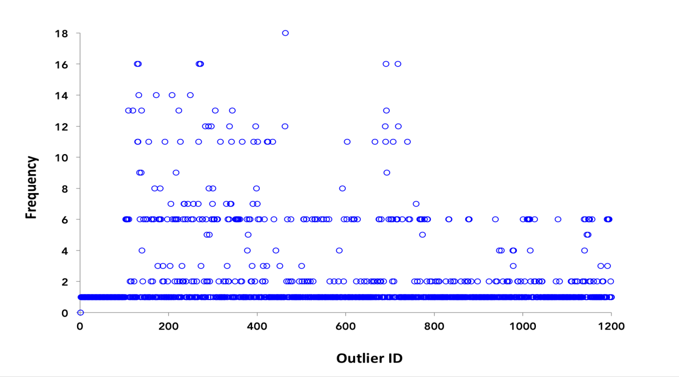

Fig. 4. Frequency of the detected bots based on max. intra-cluster distance

Figure 5 shows the relationship between legitimate users and the web bots. There is a clear difference in behaviour between legitimate users and web bots when it comes to data download and number of pages browsed. The time of a session for both kinds of users is similar, however, if we look at the relationships in terms of *"time-to-number of pages browsed"*, there is a clear difference between genuine users and web bots. We deduce that a web user visits a large number of pages in a very short time. In our case, at the threshold value of 0.09 of max-intra cluster distance, on average the web bots have crawled about 5588 pages. Such users are confirmed bots as genuine web users could not view so many pages in such a short time. Table I verifies this deduction.

Table III shows the comparison results of all the attributes of genuine and suspected web bots at different values of the max-intra cluster distance. The average downloaded data of genuine users goes up when we remove fewer web bots, and vice versa, which means that genuine web users data is sensitive to these web bots and it will eventually skew the analysis if web bots are not removed. A similar trend can be seen in the average pages browsed and time per session.

The next experiment shows the results of overlap when the intersection of avg. and max. intra-cluster distance was used to detect web bots. The overlap increases when the number of suspected web bots decreases because of the smaller

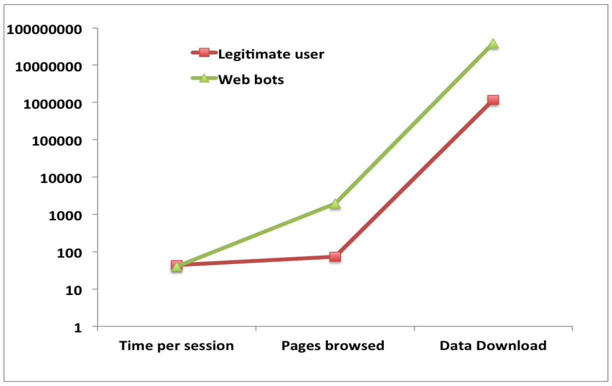

Fig. 5. Log Scale: Average of the three attributes for sessions legitimate users and web bots

TABLE II
COMPARISON OF GENUINE WEB USER AND DETECTED WEB BOTS BASED ON MAX. INTRA-CLUSTER MEASURE

Dt	Data Download		Pages browsed		Time per session	
	Legitimate	Web bots	Legitimate	Web bots	Legitimate	Web bots
0.0001	940448.61	4562939.24	62	245	45.97	44.87
0.0003	694654.85	20816311.64	70	465	46.36	44.58
0.0005	1121798.99	12112349.71	72	620	45.18	44.95
0.0007	1124596.59	12741171.99	72	649	45.11	44.99
0.0009	1124529.77	13188894.66	72	674	44.98	44.99
0.001	1124823.04	13681160.45	72	692	44.52	45.03
0.003	2694461.17	25299697.15	75	1569	41.51	44.59
0.005	1203242.02	35543601.72	76	1821	39.78	44.62
0.007	1215930.48	37782139.12	76	1986	39.90	44.63
0.009	1232103.77	36537475.73	77	1792	38.73	44.64
0.01	1232355.87	37248603.64	77	1826	39.17	44.64
0.03	1258824.53	54063456.47	78	2980	38.60	44.67
0.05	1259857.56	69038997.40	79	3538	34.75	44.69
0.07	1272779.00	92737810.08	79	4906	32.54	44.69
0.09	1293250.24	104376580.56	80	5588	23.89	44.70

number of true negatives. Approximately 100% overlap was achieved when less than 100 web bots were labelled as suspected. The value of D_t based on average intra-cluster distance and maximum intra-cluster distance for this detection was $200 * AvgIntraClust$, and $0.003 * MaxIntraClust$ respectively.

TABLE III
COMPARISON OF GENUINE WEB USER AND DETECTED WEB BOTS BASED ON MAX. INTRA-CLUSTER MEASURE

S. No.	Dt (avg)	Dt (max)	Number of Web bots	Overlap
1	d=30 × AvgIntraClust	d=0.0005 × MaxIntra	311	83.60%
2	d=40 × AvgIntraClust	d=0.0007 × MaxIntra	297	89.22%
3	d=50 × AvgIntraClust	d=0.0009 × MaxIntra	272	97.42%
4	d=200 × AvgIntraClust	d=0.003 × MaxIntra	75	98.66%

Our last experiment uses the traditional precision measure as shown in equation 5 to evaluate our proposed approach based on the ground truth. We inject different numbers of web bots into our legitimate data and try to detect those as outliers. The generated outliers were similar in behaviour to the web bots that we have suspected in the original web-log data.

$$precision = \left(\frac{TP}{TP + FP}\right) \tag{5}$$

We chose different numbers of web bots and assess the performance of our techniques. Table IV shows the precision for different numbers of injected web bots. We synthetically injected 50 to 800 web bots and found the precision of detection at value of $D_t = 10$. The overall precision is around 70% to 97% which is reasonable for such kinds of data. However, if we increase the number of web bots or we decrease the value of D_t too much the false positive ratio goes up, which affects the recall value. We conclude that with a value of D_t around 10, we get the best precision and false positive rate as shown in table IV.

TABLE IV
PRECISION OF THE DETECTED OUTLIERS BASED ON AVERAGE INTRA-CLUSTER DISTANCE FOR DIFFERENT D_T

S. No.	Number of web bots	D_t =10
1	50	53.85
2	100	71.94
3	200	81.32
4	300	87.98
5	400	92.50
6	500	93.46
7	600	93.17
8	700	92.72
9	800	97.09

V. DISCUSSIONS AND FUTURE WORK

Discussion: In the experimental section we tested the approach in two different ways, firstly, without using any ground truth, and secondly, using injected profiles that mimic web bots' behaviour.

In the first part three different measures have been used to predict an observation as a web bot. We used average intra-cluster distance, maximum intra-cluster distance, and the overlap or intersection of these two measures. By using the average and maximum intra-cluster distance measure we have shown that we can detect web bots that are significantly different from legitimate web user. One such example is that web bots that we detect browse many pages in less than one second or browse huge numbers of pages in a few minutes. We also cross check with the maximum intra-cluster distance, and with the intersection of average and maximum intra-cluster distance, and verified that the suspected web bots are real web bots. The measure $ThreshDist$ that we use has a significant impact on the number of web bots. Larger values of this measure generates fewer web bots and vice versa. Web bots generated on larger values on $ThreshDist$ have a higher tendency of being confirmed outliers than those generated at lower values of $ThreshDist$.

In the second part of the experiment we injected different numbers of web bots and used our technique to predict those as web bots. The synthetically generated web bots had similar attributes as the suspected web bots in the original data. We used the precision measure to assess the performance of our technique. The overall precision that we achieved is around 98% at $D_t = 10$ for different numbers of web bots. However, when we injected around 10% outliers, the false positive rate was high.

Future work: We have discussed one of the unexplored areas of PSO based outlier detection (web bot detection) but there are still issues related to extendability, performance, and generalization which would benefit from further investigation.

Some of the future research includes the generalization of these techniques, automating the parameter selection process, and application of HPSO-clustering based outlier detection method in different domains other than with web usage data. We would like to investigate how the selection of outlier threshold distance D_t can be automated and bound to a single value or a close range of values to be used as a universal parameter for all the datasets. Scaling the outlier detection to high dimensional data is another future research direction.

As an application of this technique we are currently looking at how the approach can be extended to tackle heterogeneity of data which incorporates numeric as well as sequential attributes of web usage data. This will include extending our existing work in [30] [31].

VI. Conclusion

Web usage data is sparse, noisy and contains outliers which cause deficiencies and inaccuracies in the output patterns. In this paper we investigated the detection of outliers in web usage data. Web bots are one of the outliers that crawl the web pages and are significantly different from genuine web users. Detecting outliers is not a trivial job as they are not all known in advance, some forge their identities, while some resemble genuine users in some of the attributes while differing in others.

Recently optimization based techniques have emerged as an important method for Knowledge Discovery and Data mining (KDD). One of the current application areas is outlier detection which has not been explored much yet but has enormous potential. In this paper we analysed web usage with outlier detection. We focused on web bots as an outlier in web usage data. The detection of web bots as outliers is very important because it can skew the entire pattern extraction process.

We introduced a Particle Swarm Optimization (PSO) based clustering technique called Hierarchical Particle Swarm Optimization based clustering (HPSO-clustering) to detect web bots in the web usage data. Using HPSO-clustering, we cluster the data and based on the avg. intra-cluster distance, max intra-cluster distance, and overlap of these distances we identified the suspected outliers.

The results show that the proposed approach has successfully identified the suspected outliers. The overlap of avg. intra-cluster distance, max intra-cluster distance is close to 100%. We then compare the characteristics of the detected web bots to the genuine web-users for cross validation, which verifies that the detected outliers are web bots and they are significantly different from genuine users.

References

[1] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data, SIGMOD '00*. New York, NY, USA: ACM, 2000, pp. 93–104.

[2] D. M. Hawkins, *Identification of outliers*. Chapman and Hall, 1980.

[3] V. Barnett and T. Lewis, *Outliers in statistical data*. Wiley, 1978.

[4] L. Duan, L. Xu, Y. Liu, and J. Lee, "Cluster-based outlier detection," *Annals of Operations Research*, vol. 168, no. 1, pp. 151–168, Apr. 2009.

[5] W. Jin, A. K. H. Tung, and J. Han, "Mining top-n local outliers in large databases," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '01*. New York, NY, USA: ACM, 2001, pp. 293–298.

[6] Z. He, X. Xu, J. Z. Huang, and S. Deng, "Fp-outlier: Frequent pattern based outlier detection," *Comput. Sci. Inf. Syst.*, vol. 2, no. 1, pp. 103–118, 2005.

[7] E. M. Knorr and R. T. Ng, "Algorithms for mining distance-based outliers in large datasets," in *Proceedings of the 24rd International Conference on Very Large Data Bases, VLDB '98*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 392–403.

[8] F. Angiulli, S. Basta, and C. Pizzuti, "Distance-based detection and prediction of outliers," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, pp. 145–160, 2006.

[9] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *SIGMOD '00: Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. ACM, 2000, pp. 427–438.

[10] S. D. Bay and M. Schwabacher, "Mining distance-based outliers in near linear time with randomization and a simple pruning rule," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '03*. New York, NY, USA: ACM, 2003, pp. 29–38.

[11] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *KDD*, vol. 96, 1996, pp. 226–231.

[12] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," in *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996*, H. V. Jagadish and I. S. Mumick, Eds. ACM Press, 1996, pp. 103–114.

[13] S. Guha, R. Rastogi, and K. Shim, "ROCK: A robust clustering algorithm for categorical attributes," in *Proceedings of the 15th International Conference on Data Engineering, ICDE '99*. Washington, DC, USA: IEEE Computer Society, 1999, pp. 512–521.

[14] R. T. Ng and J. Han, "CLARANS: A method for clustering objects for spatial data mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, pp. 1003–1016, 2002.

[15] L. Kaufman and P. Rousseeuw, *Finding Groups in Data An Introduction to Cluster Analysis*. New York: Wiley Interscience, 1990.

[16] L. Kaufman and P. J. Rousseeuw, *Clustering Large Applications (Program CLARA)*. John Wiley & Sons, Inc., 2008, pp. 126–163.

[17] M. F. Jaing, S. S. Tseng, and C. M. Su, "Two-phase clustering process for outliers detection," *Pattern Recogn. Lett.*, vol. 22, pp. 691–700, May 2001.

[18] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recogn. Lett.*, vol. 24, pp. 1641–1650, June 2003.

[19] L. Duan, L. Xu, F. Guo, J. Lee, and B. Yan, "A local-density based spatial clustering algorithm with noise," *Inf. Syst.*, vol. 32, pp. 978–986, November 2007.

[20] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: ordering points to identify the clustering structure," *SIGMOD Rec.*, vol. 28, pp. 49–60, June 1999.

[21] S. Alam, G. Dobbie, P. Riddle, and M. A. Naeem, "A swarm intelligence based clustering approach for outlier detection," in *IEEE Congress on Evolutionary Computation (CEC), 2010*. IEEE, 2010, pp. 1–7.

[22] D. Yu, G. Sheikholeslami, and A. Zhang, "Findout;: Finding outliers in very large datasets," *Knowledge and Information Systems*, vol. 4, pp. 387–412, 2002.

[23] C. Aggarwal and S. Yu, "An effective and efficient algorithm for high-dimensional outlier detection," *The VLDB Journal*, vol. 14, pp. 211–221, April 2005.

[24] G. Williams, R. Baxter, H. He, S. Hawkins, and L. Gu, “A comparative study of rnn for outlier detection in data mining,” in *Proceedings of the 2002 IEEE International Conference on Data Mining, ICDM ’02*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 709–712.

[25] A. W. Mohemmed, M. Zhang, and W. N. Browne, “Particle swarm optimisation for outlier detection,” in *GECCO*, 2010, pp. 83–84.

[26] S. Hawkins, H. He, G. J. Williams, and R. A. Baxter, “Outlier detection using replicator neural networks,” in *Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery, DaWaK 2000*. London, UK: Springer-Verlag, 2002, pp. 170–180.

[27] S. Alam, G. Dobbie, P. Riddle, and M. A. Naeem, “Particle swarm optimization based hierarchical agglomerative clustering,” *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2, pp. 64–68, 2010.

[28] S. Alam, G. Dobbie, Y. Koh, and P. Riddle, “Clustering heterogeneous web usage data using hierarchical particle swarm optimization,” in *IEEE Symposium on Swarm Intelligence (SIS), 2013*, 2013, pp. 147–154.

[29] S. Alam, G. Dobbie, and P. Riddle, “Exploiting swarm behaviour of simple agents for clustering web users session data,” in *Data Mining and Multi-agent Integration*. Springer, 2009, pp. 61–75.

[30] S. Alam, “Intelligent web usage clustering based recommender system,” in *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 2011, pp. 367–370.

[31] S. Alam, G. Dobbie, P. Riddle, and Y. S. Koh, “Hierarchical pso clustering based recommender system,” in *2012 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2012, pp. 1–8.