**Evaluation of scalable fair clustering machine learning methods for threat hunting in cyber-physical systems**

Dilip Sahoo [[1]](#footnote-1), Aaruni Upadhyay [[2]](#footnote-2)

**Abstract**

With a rapid increase in the automation of industrial control systems, it has become vital to defend them against cyberattacks. Clustering is a widely used, unsupervised machine learning technique to detect malware from behavior data of control systems. Clustering algorithms can be susceptible to amplifying biases that may be present in the input datasets. Recent works in fair clustering attempt to solve this problem by making them balanced with respect to certain sensitive attributes.

The fair k-median clustering is a newly developed technique that allows the assignment of input points to clusters such that the number of each type of point is balanced as per the fairness criteria. In this experiment, we have selected a recent work that implements a fair and scalable k-median clustering algorithm with near-linear runtime. We test our system on 4 new datasets belonging to IoT and Water distribution systems and evaluate the performance and accuracy of our results.

**Keywords:** Fair Clustering, K-Median, Fairness, machine learning, cyber-physical systems, Internet of Things

# Introduction

The safety of our society and infrastructure depends on keeping our mission-critical systems such as Water distribution safe from cyber-attacks [1]. Many such systems work in tandem with the Internet of Things (IoT) systems and other cyber-physical systems that are susceptible to attacks by hostile nations and other non-state actors [2][3]. Machine learning is increasingly being used in designing systems that can detect such attacks through clustering which is an unsupervised machine learning technique [4][5].

The behavior of machine learning systems is dependent on the training data which may contain biases which may in return, result in the bias being reflected in the outcome [6]. This problem was highlighted by Chierichetti in [7] where they argue that the biases may still indirectly appear in results even if unprotected attributes (such as a person’s height) are used for making decisions instead of protected ones such as race and gender. This could happen because of the hidden correlations that may exist between protected and unprotected attributes, for example, average height (unprotected) is related to gender (protected) and can be exploited as a proxy for discrimination.

The established approach followed by the machine learning researchers to solve this problem can be traced back to the US Supreme Court case Griggs v. Duke Power Co. [8] that resulted in the emergence of the concept of adverse impact. Adverse impact occurs when a practice negatively and disproportionately affects a protected group regardless if it was indirectly or unintentionally. The “80% rule” was adopted by the researchers as a generally accepted way to measure adverse impacts which states that an adverse impact has occurred if “the selection rate for a certain group is less than 80 percent of that of the group with the highest selection rate” [9].

Chierichetti applied this notion of fairness to clustering by introducing the use of fairlets that groups together the datapoints while preserving the fairness objective. These fairlets are then combined to form clusters by using existing k-median algorithms. This way, fair clustering reduces biases by placing constraints on the clusters so that the probability of a class of input data points being present in a cluster, is strictly greater than zero. However fair clustering achieved using this method has a super-quadratic runtime. The paper we are basing our research on [10] presents a new implementation of this fair clustering method that runs in near-linear time and therefore offers performance that scales with the input size.

To formally outline the problem, we must first define fair clustering and we will use the same definition as our base paper. Consider *n* number of points *P* from the training dataset such that each point belongs to one of two types: *T1* and *T2*. In a practical application, these classes can correspond to any legally protected attribute such as gender where *T1: Male* and *T2: Female*. Let’s define the Balance of a subset *S* such that *S ⊆ P,* assuming ST1 and ST2 represent subsets of *T1* and *T2* in the set *S*.

*Balance(S) =*

If we assume *T1 < T2*, then the clustering of *P* performed over *(T1, T2)* would be defined as fair if for all clusters *C*:

*Balance(C)* ≥

A formal definition of k-median fair clustering can now be stated as the division of input point set *P* into *k* clusters such that the sum of distances of each point *p ∈ P* to the center of their cluster is minimum AND all clusters have a balance of at least .

Our contribution through this research is to run the k-median fair clustering implementation of original authors [10] on 4 new Cybersecurity related datasets and evaluate the performance and accuracy of our results. We demonstrate that our algorithm runtime is near-linear which is the same as expected in the original paper and so we show that the algorithm is scalable also for much bigger datasets like SWaT. The datasets used in our experiment are referenced in [11][12][13].

Section 2 of this paper contains a Literature Review of the recent work done in fair clustering. Section 3 details the methodology of our experiment and is followed by a discussion of our Experimentation and Results in Section 4. A comparison of our findings with that of the base paper is presented in Section 5. Our concluding statements and avenue for future work are presented in Section 6.

# Related Work

Our dependence on critical systems like electricity and water distribution systems makes them a very lucrative target for our adversaries. Similarly, IoT networks that are frequently integrated with such systems are a frequent target for attack and are also used as a vector for further malware spread and launching DDoS attacks [14]. Machine learning researchers have been actively exploring ways to detect attacks on these crucial systems by utilizing machine learning techniques. [15] and [16] are an example of the use of machine learning techniques in the detection of threats in IoT and Water distribution systems respectively.

Due to an ever increasing adoption of IoT systems, malware detection in these devices has become a topic of great interest among cybersecurity researchers [17] [18][19]. Authors in [20] highlight the new paradign of edge computing in IoT networks and demonstrate the use of fuzzy and fast fuzzy pattern tree methods for detecting IoT malware. Authors in [21] present an interesting approach to detect the presence of ransomware in IoT networks by monitoring the power consumption patterns of IoT devices. Their machine learning based approach was successfully able to classify ransomware from non-malicious applications and produced a better accuracy and precision rate than K-Nearest Neighbors, Support Vector Machine, Neural Network and Random Forest methods. [22] presents a approach for detecting intrusion in IoT networks based on two-layer dimension reduction and two-tier classification module to detect User-To-Root (U2R) and Remote-To-Local (R2L) attacks. The authors use the NSL-KDD dataset and demonstrate that their approach performs better than earlier models designed to detect R2L and U2R attacks. Attackers often employ the use of code level polymorphism to evade any opcode based malware detection algorithms. Authors in [23] demonstrate the use of sequential pattern mining approach to select best features to train KNN, SVM, AdaBoost and other machine learning models and are able to detect IoT malware with polymorphed code to escape detection.

One pitfall of using machine learning can be the appearance of bias in the output if we are not careful. The authors in [24] present the case of a medical center that used an algorithm, that was used to screen patients in an intensive care program, to be racially biased against black patients. The algorithm was found to be functioning correctly, but the bias was inadvertently introduced because it wrongly established that black patients are healthier because they spend less on healthcare. Bias in real-world computing applications can have serious ethical implications as highlighted in [25]. Authors argue that any attempt at fairness, even if it is not 100% effective, should be incorporated in our algorithms instead of waiting for a perfect fair algorithm to emerge. The goal of the fairness algorithms should not be to have a perfect solution to the fairness problem but instead to maximize the common good by achieving whatever fairness is attainable today.

The notion of fair clustering was first introduced in [7] who articulated the implementation of fair clustering for both k-center and k-median objectives. They introduced the idea of division of pointset into smaller minimal subsets (called *fairlets*) that fulfill the fairness criteria while meeting the clustering objective. They used 3 datasets (*Diabetes, Bank, and Census*) to evaluate their algorithm and compare the performance and fairness of their approach to the classical k-center and k-median algorithms. The results successfully demonstrated that traditional k-center and k-median algorithms produced unfair clusters as compared to their fair algorithms. However, their fair algorithm was computationally harder than the traditional algorithms.

Several researchers have done subsequent work based on the original work in [7]. Authors in [26] study the problem of low-cost fair clustering where the data points can belong to multiple protected classes. Their implementation allows for placing upper and lower bounds for any class in a cluster while maintaining fairness for data points that may even span multiple protected classes. [27] looks at the effects of using *fairlet* based approach in fair clustering and raises the important concern of scalability of that algorithm. They propose the use of the concept of *coresets* that is tailored for use in fair clustering problem to provide their own algorithms for fair k-means clustering as an improvement on algorithms in [7]. They empirically demonstrate how *coresets* enable fair clustering algorithms and improve output quality by using better albeit slower algorithms.

Several works have also emerged highlighting the usage of fair algorithms in real applications. [28] uses the fair design for preserving privacy by adding a constraint so that a cluster will be formed only if a lower bound on the number of points is achieved to preserve anonymity. Authors in [29] apply the notion of fairness to address the allocation problem where we want to distribute a limited resource to be distributed across different clusters without bias. For example, the allocation of housing loans based on creditworthiness across groups without a prejudice based on race. Authors in [30] propose fairness preserving algorithms that can produce a summary of texts (e.g. blog posts) in a way so that it fairly represents the opinion of different social groups.

# Methodology

This section describes the detailed steps taken to evaluate the fair clustering Machine Learning (ML) model using four different datasets. Each of the four datasets contained data labeled as ‘normal/good ware’ and ‘attack/malware’ which were collected from the respective testbeds under normal and attack scenarios. The raw datasets were first pre-processed and then feature selection and extraction were applied to reduce overall dimensionality. Finally, the scalable fair clustering algorithm [10] was adopted as the ML classifier for the experiment and the model performance was measured based on cost, runtime, and accuracy. *Figure 1* illustrates the experiment workflow structure.

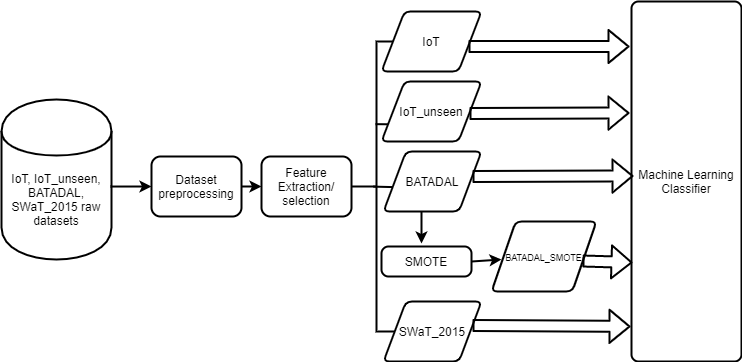


Figure 1: Experiment workflow structure

## Datasets Preprocessing

The preprocessing of each dataset was done in separate ways based on the type and nature of the raw data samples. The detailed preprocessing approach for them is described below.

### IoT dataset

The IoT dataset [13] had a total of 614 text files in the dataset among which 362 files were labeled as ‘good ware’ and 252 as ‘malware’. All the text files contained the opcode data collected from the testbed. These individual text files were processed using Term Frequency- Inverse Document Frequency (TF-IDF) and a TF-IDF score was assigned to each of the opcodes. After, assigning the TF-IDF score the opcodes were treated as feature columns for the modified dataset with the column header as the opcode names and rows as the TF-IDF score assigned to them. Each row created corresponded to one of the text files in the raw data samples and finally, we obtained a CSV file having a total of 236 feature columns and 512 rows after removing anomalies. In the obtained CSV file, there were 271 ‘good ware’ and 281 ‘malware’ data samples and were labeled as 0 and 1, respectively. Hence, we got our final CSV file with all data samples in a numerical format.

### IoT\_unseen dataset

The IoT\_unseen dataset [13] had a similar raw data format as the IoT dataset described in the previous section, except there were differences in the number and type of data samples. This dataset was used mainly for the ML model accuracy evaluation purpose. It is worth mentioning that this dataset was not used in all stages of experiments performed for the main IoT dataset and the rest of the datasets because this dataset is equivalent to the IoT dataset in nature. This dataset had 51 opcode-based text files and all of them were labeled as ‘malware’. A similar TF-IDF based processing was done again to obtain a CSV file with 51 rows against each of the raw text files and 666 feature columns.

### BATADAL

It is an industrial control system (ICS) dataset [12] collected during ‘Normal’ and ‘Attack’ scenarios from different sensors of the testbed. The dataset contains 12446 ‘Normal’ and 492 ‘Attack’ data samples. During data preprocessing the .npy file was converted to CSV using ‘numpy’ library (numpy.org) and all features were converted to numerical data.

### SWaT\_2015

SWaT\_2015 data was collected from the testbed of a six-stage secure water treatment system under ‘Normal’ and ‘Attack’ scenarios [11]. The dataset contains 1387095 ‘Normal’ data samples and 54621 ‘Attack’ samples. During preprocessing of this dataset, a similar approach as ‘BATADAL’ was used and the final CSV file was obtained.

## Feature Selection and Feature Extraction

For feature selection and extraction, we implemented SelectKBest with Chi2 and ExtraTreeClassifier feature scoring method from scikit-learn (scikit-learn.org). The fair clustering ML model [10] adopted for our experiment requires at least one feature column as the sensitive attribute with categorical data. Hence, One of the feature columns, from every four datasets, was decided to be taken as a sensitive attribute and all the values present under the sensitive attribute were converted to categorical data i.e either 1 or 0. It is worth noting that the balance parameters derived from the ratio of categorical values(1 or 0) in the sensitive attribute are used to create the target fairlet decomposition clusters. *Table 1* shows the sensitive attribute name and number of other features selected for the ML model accuracy evaluation for each dataset during the experiment.

3.2.1 Upsampling using Synthetic Minority Over-sampling Technique (SMOTE)

In the ‘BATADAL’ dataset, we observed a huge variance in the number of data points between ‘Normal’ and ‘Attack’ families. To ensure that our fair clustering ML model is not biased towards the majority class, we used SMOTE technique to balance the majority and minority class. The new dataset after upsampling held 12446 data points for each of the ‘Normal’ and ‘Attack’ class. The new SMOTE enhanced BATADAL dataset is referred to as ‘BATADAL\_SMOTE’. Accuracy evaluation for the ‘BATADAL\_SMOTE’ dataset was done separately to compare it with the accuracy of the original BATADAL dataset during the experiments.

|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** | **Total Number of Data Points** | **Sensitive Attribute** | **No. of selected features** |
| IoT | 512 | stmgeia | 12 |
| IoT\_unseen | 51 | bnd | 665 |
| BATADAL | 12938 | P\_J422\_code | 9 |
| BATADAL\_SMOTE | 24892 | P\_J422\_code | 9 |
| SWaT\_2015 | 1441715 | UV401 | 5 |

Table 1: Sensitive attribute and number of selected features for the datasets

## Fair clustering ML model implementation

In this phase, the fair clustering ML model was fed with the extracted features of all four datasets. All the experiments were conducted in a Windows 10 virtual machine environment with 2.21 GHz 64-bit intel i7 processor and 4GB RAM. Jupyter Notebook was used with python 3.6.5 and MATLAB engine.

# Experiments and Results

This section describes the details of three individual experiments conducted and highlights the results. The evaluation measures used to assess the results are described first in section 4.1 and then the details of conducted experiments and their results are discussed in sections 4.2,4.3 and 4.4. The first experiment has been conducted to evaluate the performance of the ML model based on the overall cost and runtime. In the second experiment, the model accuracy is evaluated for every dataset. A separate experiment is done to test the effectiveness of the fair clustering algorithm by comparing its results with the results of a relevant normal clustering algorithm described in section 4.4.

## Evaluation measures

In our first experiment described in section 4.2, evaluation of the near-linear behavior and performance of the fair clustering algorithm is done by measuring the fairlet decomposition cost, fair clustering cost, and runtime. The fairlet decomposition cost denotes the distance between the points and their cluster centroids. Fair clustering cost is the total algorithm cost and the runtime states the time taken in seconds to run the clustering on a certain number of data points.

For the second and third experiments, we used commonly used machine learning matrices which are discussed below. A Confusion matrix represents the summary of all the predicted results of an algorithm in terms of the number of True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN). Indicators used to assess the results of the fair clustering algorithm in our experiment are derived from the confusion matrix. Calculation of ‘Precision’ and ‘Recall’ is based on the binary label i.e ‘Normal’ and ‘Attack’. ‘F1-score’ combines precision and recall and to provide the clustering performance. ‘Accuracy’ denotes how accurately the clustering algorithm detects the binary classes i.e ‘Normal’ or ‘Attack’.

**TP:** Normal observation is predicted as normal

**TN:** Attack observation is predicated as an attack

**FP:** Attack observation is predicted as normal

**FN:** Normal observation is predicated as an attack

**Accuracy** =

**Precision** =

**Recall** =

**F1-score** =

4.1.1 Accuracy evaluation for two clusters and Multicluster clustering

To determine the performance of the fair clustering algorithm output results in terms of ‘Accuracy’ using two clusters(k=2), it is assumed that all data points predicted under one cluster(cluster-1) are ‘Normal’ and the other cluster (cluster-2) are of ‘Attack’ type. Then, it is calculated how many data points in cluster-1 are indeed ‘Normal’ and cluster-2 are indeed ‘Attack’ from the known labels. This resulted in the confusion matrix construction where the diagonal elements represented TPs and TNs respectively along with the off-diagonal elements as FPs and FNs.

In contrast to clustering with k=2, we have used a different approach for multiple clustering (k=n, where the value of n is 4,6,10,15 and 20). In the case of multiple clustering, we assume each of the data points in a cluster as ‘Normal’ if the actual labels for most of the data points are of ‘Normal’ type. Similarly, in a cluster, if the actual labels of majority data points are of ‘Attack’ type then every data point belongs to that cluster is assumed as ‘Attack’ and confusion matrix was constructed. In the multicluster clustering, the final accuracy is the average accuracy of each cluster.

## Experiment-I and results

In Experiment-I, all datasets described in section 3 except ‘IoT\_unseen’ and ‘BATADAL\_SMOTE’ were fed into the fair clustering algorithm. The input dimension and balance parameters for the dataset during the experiment are mentioned in *Table 2*. We did not use all features extracted for Experiment-II accuracy evaluation in this phase of the experiment. The performance of the algorithm was measured in terms of cost and runtime with a different number of data samples and dimensions for all datasets. The results in *Table 2* shows that the algorithm performance varies depending on the overall dimension and size of the dataset. It can be noticed that the IoT dataset having a high dimension with only 500 data samples took 20 seconds to complete the whole clustering process. In contrast, the SWaT\_2015 dataset with 40 times more data samples than the IoT dataset took little more than twice the time needed for IoT.

In addition to the performance evaluation of the fair clustering algorithm, we also checked the scalability of the fair clustering algorithm in terms of runtime for a range of data points. All three datasets were fed into the fair clustering algorithm with a different number of data samples more than once and the time taken to complete the process for each sample was recorded. We then plotted the graphs between the number of data samples Vs runtime to investigate whether the algorithm scales in linear time. *Figure 2*, *Figure 3*, and *Figure 4* show the graphs between the number of data points Vs runtime for the three datasets. The results show that the plotted graphs for SWaT\_2015 and BATADAL datasets are almost a straight line with a slight deviation for the IoT dataset.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Dimension** | **Balance** | **No of Clusters k=20** | | | |
| **Fairlet Decomposition Cost** | **Fair Clustering Cost** | **Fairlet Decomposition time (in sec)** | **Total time (in sec)** |
| IoT (500 sample) | 11 | 0.5 | 50.8 | 56.4 | 0.29 | 20 |
| BATADAL (1000 sample) | 6 | 0.33 | 142 | 205 | 8.4 | 37 |
| SWaT\_2015 (20000 sample) | 2 | 0.02 | 17809 | 39209 | 10 | 41 |

Table 2: Performance evaluation of each of the datasets for a specific number of data samples in terms of runtime and cost

Figure 2: No of datapoints Vs Runtime (in sec) graph for IOT dataset

Figure 3: No of datapoints Vs Runtime (in sec) graph for BATADAL dataset

Figure 4: No of datapoints Vs Runtime (in sec) graph for SWaT\_2015 dataset

## Experiment-II and results

In experiment-II, all five datasets described in *Table 1* are fed into the fair clustering algorithm and the desired numbers of clusters were obtained as output. We used the elbow method from scikit-learn to predict the optimum number of cluster(k) for each dataset.

*Figure 5*, *Figure 6*, *Figure 7*, and *Figure 8* illustrate the optimum cluster value(k) for IoT, IoT\_unseen, BATADAL, and SWaT\_2015 datasets respectively using the elbow method.

A close up of a logo

Description automatically generated

Figure 5: Elbow method showing optimum k for IoT dataset

A close up of a logo

Description automatically generated

Figure 6: Elbow method showing optimum k for IoT\_unseen dataset

A close up of a logo

Description automatically generated

Figure 7: Elbow method showing optimum k for BATADAL dataset

A close up of a mans face

Description automatically generated

Figure 8: Elbow method showing optimum k for SWaT\_2015 dataset

After having an idea about the optimum cluster value for each dataset, we decided to run each dataset with different values of ‘k’ i.e k=2,4,6,10,15 and 20. The performance of fair clustering algorithm is measured in terms of ‘Accuracy’, ‘Precision’, ‘Recall’, and ‘F1-score’ based on the method discussed in section 4.1.1. The results obtained from the ML model for each dataset are shown in *Table 3*. IoT dataset achieved the highest accuracy of 99% with two clusters (k=2) and the accuracy remained slightly less with other values of cluster numbers(k). IoT\_unseen obtained an accuracy of 80% for ‘k-value’ as 2 and it could obtain 100% accuracy when ‘k-value’ was 4,6,10,15 and 20. BATADAL and BATADAL\_SMOTE dataset shown a significant difference in their result. BATADAL dataset achieved maximum accuracy of 97% compared to the maximum accuracy of 67% for the BATADAL\_SMOTE dataset. The reason behind such difference in results of BATADAL and BATADAL\_SMOTE is attributed to the BATADAL dataset biased with majority class. The SWaT\_2015 dataset reported a maximum accuracy of 97% with ‘k-value’ as 6. Overall, the fair clustering algorithm predicted the best results for IoT and IoT\_unseen datasets among other datasets.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Data Set** | **No of Clusters(k)** | **Accuracy (%)** | **Precision (%)** | **Recall (%)** | **F1-score (%)** |
| IoT | 2 | 99 | 99 | 99 | 99 |
| 4 | 97 | 97 | 97 | 97 |
| 6 | 97 | 97 | 97 | 97 |
| 10 | 98 | 98 | 98 | 98 |
| 15 | 97 | 98 | 97 | 98 |
| 20 | 99 | 99 | 99 | 99 |
| IoT\_unseen | 2 | 80 | 100 | 80 | 89 |
| 4 | 100 | 100 | 100 | 100 |
| 6 | 100 | 100 | 100 | 100 |
| 10 | 100 | 100 | 100 | 100 |
| 15 | 100 | 100 | 100 | 100 |
| 20 | 100 | 100 | 100 | 100 |
| BATADAL | 2 | 81 | 93 | 81 | 86 |
| 4 | 96 | 97 | 96 | 95 |
| 6 | 97 | 97 | 97 | 95 |
| 10 | 97 | 96 | 97 | 96 |
| 15 | 97 | 97 | 97 | 96 |
| 20 | 97 | 97 | 97 | 96 |
| BATADAL\_SMOTE | 2 | 59 | 64 | 59 | 57 |
| 4 | 62 | 65 | 62 | 61 |
| 6 | 62 | 62 | 62 | 62 |
| 10 | 66 | 69 | 66 | 65 |
| 15 | 66 | 69 | 66 | 66 |
| 20 | 67 | 67 | 67 | 66 |
| SWaT\_2015 | 2 | 74 | 86 | 74 | 79 |
| 4 | 93 | 93 | 93 | 93 |
| 6 | 97 | 96 | 97 | 96 |
| 10 | 95 | 95 | 95 | 95 |
| 15 | 86 | 85 | 86 | 84 |
| 20 | 96 | 96 | 96 | 96 |

Table 3: Results obtained from fair clustering algorithm

## Experiment-III and results

Experiment-III is conducted to evaluate the suggested fair clustering algorithm performance in terms of ‘Accuracy’ by comparing it with the equivalent normal clustering algorithm. The exact datasets taken for Experiment-II(with fair clustering step) were fed into ‘kmedoids’ clustering algorithm [31] from MATLAB engine without combining the fairlet decomposition step. The ‘Accuracy’ for all datasets without fair clustering is recorded and then compared with the respective results from Experiment-II. *Table 4* shows the comparison of the results between fair clustering and normal clustering with ‘k-value’ as 2.

|  |  |  |
| --- | --- | --- |
| **Data Set** | **Accuracy (%) with k=2** | |
| **Fair\_clustering** | **Without Fair\_clustering** |
| IoT | 99 | 92 |
| IoT\_unseen | 80 | 61 |
| BATADAL | 81 | 82 |
| BATADAL\_SMOTE | 59 | 57 |
| SWaT\_2015 | 74 | 74 |

Table 4: Results comparison between fair clustering and normal clustering

# Results Comparison

The research done in [10] has evaluated the performance of the suggested fair clustering algorithm in terms of cost and runtime. However, the authors did not provide the details of the accuracy they achieved. The results of our experiment show that the fair clustering algorithm achieved better accuracy than the normal clustering algorithm for every dataset except BATADAL and SWaT\_2015 where both performed almost equally.

# Conclusion and Future Work

We successfully implemented a scalable fair clustering machine learning algorithm and evaluated the model with different cyber-physical datasets. Our results show that the model scales linearly with the number of input data points even for a large dataset like SWaT\_2015. The unsupervised fair clustering algorithm was run with different cluster values (k= 2,4,6,10,15 and 20) as well as different input balance parameters (mentioned in *Table 2*) which provided us with a comprehensive view of our results. Our model achieved an accuracy of 99% with a precision of 99% and recall of 99% for IoT dataset which was best among all other datasets. While evaluating the IoT\_unseen dataset, the model accomplished classification with 100% accuracy. Although we tried to address the majority class biasing issue in BATADAL dataset by creating synthetic data samples, the accuracy of BATADAL\_SMOTE remained relatively lower at 67%. Overall, all our datasets exhibited high accuracy of above 95% except BATADAL\_SMOTE.

For future work, the fair clustering model implemented in this paper could be tested on other different cyber-physical systems. Due to the resource limitation of our research environment, we could only feed up to 20000 samples for accuracy evaluation, but this limit can be increased to improve the model prediction capabilities. While up-sampling datasets with artificial data provided a good method for balancing the BATADAL dataset, it is not as effective as original data collected from a real-world scenario. We expect that datasets collected from real-world systems with higher data quality will improve the resulting outcome of our model.

**References**

[1] S Watson, A Dehghantanha (2016), "Digital Forensics: The Missing Piece of the Internet of Things Promise", (Elsevier) Journal of Computer Fraud & Security, DOI: <https://DOI.org/10.1016/S1361-3723(15)30045-2>

[2] PN Bahrami, A Dehghantanha, T Dargahi, R Parizi, KKR Choo, HHS Javadi (2019), “A Cyber Kill Chain based Analysis of Advanced Persistent Threat Actors: Analogy of Tactics, Techniques and Procedures”, Journal of Information Processing Systems (JIPS), DOI:[10.3745/JIPS.03.0126](https://doi.org/10.3745/JIPS.03.0126)

[3] M Conti, A Dehghantanha, K Franke, S Watson (2017), “Internet of Things Security and Forensics: Challenges and Opportunities”, (Elsevier) Future Generation Computer Systems Journal, DOI: <https://DOI.org/10.1016/j.future.2017.07.060>

[4] C. Wang, Z. Qin, J. Zhang and H. Yin, "A malware variants detection methodology with an opcode based feature method and a fast density based clustering algorithm," 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), Changsha, 2016, pp. 481-487.

[5] R. A. Awad and K. D. Sayre, "Automatic clustering of malware variants," 2016 IEEE Conference on Intelligence and Security Informatics (ISI), Tucson, AZ, 2016, pp. 298-303.

[6] H HaddadPajouh, R Khayami, A Dehghantanha, KKR Choo, R Parizi (2020), “AI4SAFE-IoT: an AI-powered secure architecture for edge layer of Internet of things”, (Springer) Journal of Neural Computing and Applications, <https://link.springer.com/article/10.1007/s00521-020-04772-3>

[7] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. 2017. Fair clustering through fairlets. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS’17). Curran Associates Inc., Red Hook, NY, USA, 5036–5044.

[8] Burger, Warren Earl, and Supreme Court Of The United States. U.S. Reports: Griggs v. Duke Power Co., 401 U.S. 424. 1970. Periodical. https://www.loc.gov/item/usrep401424/

[9] Mondragon, N. (2019, October 17). What is Adverse Impact? And Why Measuring It Matters. Retrieved April 20, 2020, from https://www.hirevue.com/blog/what-is-adverse-impact-and-why-measuring-it-matters

[10] Backurs, A., Indyk, P., Onak, K., Schieber, B., Vakilian, A. and Wagner, T., 2019. Scalable fair clustering. arXiv preprint arXiv:1902.03519.

[11] Goh, Jonathan & Adepu, Sridhar & Junejo, Khurum & Mathur, Aditya. (2016). A Dataset to Support Research in the Design of Secure Water Treatment Systems.

[12] The BATtle of the Attack Detection ALgorithms (BATADAL) Dataset. Retrieved April 23, 2020, from https://www.batadal.net/

[13] Cyber Science Lab - Malware Datasets. Retrieved from https://cybersciencelab.org/

[14] M. Roopak, G. Y. Tian and J. Chambers, "An Intrusion Detection System Against DDoS Attacks in IoT Networks," 2020 10th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2020, pp. 0562-0567.

[15] M. Roopak, G. Yun Tian and J. Chambers, "Deep Learning Models for Cyber Security in IoT Networks," 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2019, pp. 0452-0457.

[16] P. Nader, P. Honeine and P. Beauseroy, "Detection of cyberattacks in a water distribution system using machine learning techniques," 2016 Sixth International Conference on Digital Information Processing and Communications (ICDIPC), Beirut, 2016, pp. 25-30.

[17] S. M. Tahsien, H. Karimipour, P. Spachos, “Machine Learning Based Solutions for Security of Internet of Things (IoT): A Survey”, Journal of Network and Computer Applications- Elsevier (IF: 5.3), vol. 161, pp. 1-18, May 2020. <https://doi.org/10.1016/j.jnca.2020.102630>

[18] H HaddadPajouh, A Dehghantanha, R Khayami, KKR Choo (2018), “A Deep Recurrent Neural Network Based Approach for Internet of Things Malware Threat Hunting”, (Elsevier) Journal of Future Generation Computer Systems, DOI: <https://DOI.org/10.1016/j.future.2018.03.007>

[19] A Azmoudeh, A Dehghantanha and KKR Choo (2018), “Robust Malware Detection for Internet Of (Battlefield) Things Devices Using Deep Eigenspace Learning”, IEEE Transactions on Sustainable Computing, DOI: 10.1109/TSUSC.2018.2809665

[20] E Modiri, A Azmoodeh, A Dehghantanha, DE Newton, R Parizi, H Karimipour (2019), “Fuzzy Pattern Tree for Edge Malware Detection and Categorization in IoT”, (Elsevier) Journal of System Architecture, DOI: <https://doi.org/10.1016/j.sysarc.2019.01.017>

[21] A Azmoodeh, A Dehghantanha, M Conti, KKR Choo (2018), “Detecting Crypto-Ransomware in IoT Networks Based On Energy Consumption Footprint”, (Springer) Journal of Ambient Intelligence and Humanized Computing, DOI: https://DOI.org/10.1007/s12652-017-0558-5

[22] H HaddadPajouh, R Javidan, R Khaymi, A Dehghantanha and KKR Choo (2016), “A Two-layer Dimension Reduction and Two-tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks”, IEEE Transactions on Emerging Topics in Computing; DOI: 10.1109/TETC.2016.2633228

[23] H Darabian, A Dehghantanha, S Hashemi, S Homayoun, and KKR Choo (2019), “An Opcode based Technique for Polymorphic Internet of Things Malware Detection”, (Wiley) Concurrency and Computation: Practice and Experience, DOI: https://DOI.org/10.1002/cpe.5173

[24] Obermeyer, Z., Powers, B., Vogeli, C., & Mullainathan, S. (2019, October 25). Dissecting racial bias in an algorithm used to manage the health of populations. Retrieved April 24, 2020, from https://science.sciencemag.org/content/366/6464/447

[25] J. Sylvester and E. Raff, "What About Applied Fairness?", Machine Learning: The Debates (ML-D) organized as part of the Federated AI Meeting (FAIM 2018), 2018, [online] Available: http://arxiv.org/abs/1806.05250.

[26] Bera, S., Chakrabarty, D., Flores, N., & Negahbani, M. (2019). Fair algorithms for clustering. In Advances in Neural Information Processing Systems (pp. 4955-4966).

[27] Schmidt, M., Schwiegelshohn, C., & Sohler, C. (2018). Fair coresets and streaming algorithms for fair k-means clustering. *arXiv preprint arXiv:1812.10854*.

[28] Rösner, C., & Schmidt, M. (2018). Privacy preserving clustering with constraints. arXiv preprint arXiv:1802.02497.

[29] Hadi Elzayn, Shahin Jabbari, Christopher Jung, Michael Kearns, Seth Neel, Aaron Roth, and Zachary Schutzman. 2019. Fair Algorithms for Learning in Allocation Problems. In FAT\* ’19: Conference on Fairness, Accountability, and Transparency (FAT\* ’19), January 29–31, 2019, Atlanta, GA, USA. ACM, New York, NY, USA, 11 pages. https://doi.org/ 10.1145/3287560.3287571

[30] Abhisek Dash, Anurag Shandilya, Arindam Biswas, Kripabandhu Ghosh, Saptarshi Ghosh, and Abhijnan Chakraborty. 2019. Summarizing User-generated Textual Content: Motivation and Methods for Fairness in Algorithmic Summaries. Proc. ACM Hum.-Comput. Interact. 3, CSCW, Article 172 (November 2019), 28 pages. DOI:https://doi.org/10.1145/3359274

[31] k-medoids clustering algorithm. (n.d.). Retrieved April 24, 2020, from https://www.mathworks.com/help/stats/kmedoids.html

1. Dilip Sahoo is with Cyber Science Lab, University of Guelph, Ontario, Canada, e-mail: [dilip@cybersciencelab.org](mailto:dilip@cybersciencelab.org) [↑](#footnote-ref-1)
2. Aaruni Upadhyay is pursuing MCTI, University of Guelph, Ontario, Canada, e-mail: [aupadhya@uoguelph.ca](mailto:aupadhya@uoguelph.ca)

   Corresponding author: Aaruni Upadhyay(e-mail: aupadhya@uoguelph.ca) [↑](#footnote-ref-2)