Project Report

Paper: Behavioral Clustering of HTTP-Based Malware and Signature Generation Using Malicious Network Traces.
Team: Vineeth CS19B080, Ravi CS19B081

1 Implementation

This project attempts to employ clustering to search for anomalous network behaviour. We have referred to the paper 'Behavioral Clustering of HTTP-Based Malware, and Signature Generation Using Malicious Network Traces' and attempted to adapt it for our dataset.

This paper aims to get higher quality clusters using coarse clustering, followed by fine clustering. This paper performs coarse clustering based on statistical information regarding how the malware uses the web and then performs fine clustering using structural similarity among sequences of HTTP requests.

For our dataset, there are a vast number of almost empty features. As shown in Fig1.1, for most of the continuous features, about 90% of the data points revert to default values like 0, -1, [], etc., to be precise, we got only 17 features which are non-empty.

```
 hostname_caseratio  100.00%
       goal_encoded  100.00%
 filename_caseratio  100.00%
              Src_P  98.99%
             Dest_P  98.99%
            Dest_IP  98.99%
   dns_status_ratio  98.96%
                  ....
hostname_alpharatio  91.89%
urlspecialcharratio  91.88%
   url_query_values  91.88%
    url_query_names  91.88%
    url_path_length  91.88%
                url  91.88%
```

Figure 1.1: Percentage of rows with default values

Furthermore, the required URL field is empty for most of the data. In the paper that we were implementing, they did fine-grained clustering using URL data. For rows where the URL is not available, the corresponding statistics, such as urlalpharatio and url_path_length are also empty. Similar for hostname and SNI statistics. As a result, a significant number of fields have no relevant data.

The non-empty features in the data set given to us correspond to flow statistics. In the paper, they used these in coarse-grained clustering. We did not implement fine-grained clustering because we do not have the required data. To compensate for that, we have experimented with various clustering algorithms. The paper uses single-linkage hierarchical clustering, so apart from that, we also tried K-means clustering and ward-linkage hierarchical clustering.

We need to perform clustering based on the most relevant features in differentiating between various families of malware and non-malware. In order to perform this sort of feature selection, we used chi-squared feature selection. Having chosen these columns, all we need to do is perform clustering based on the best features we found.

Before we cluster the data in an attempt to predict the family of the malware, it might help to first tryout classification using the continuous variables to find out if the features even contain sufficient information to predict whether a behaviour is indicative of malware. We tried using an SVM classifier and a feedforward neural network(FFNN) to do the same. The SVM classifier predicted the malware family with a test accuracy of 57.83%, while the FFNN predicted the malware family with a test accuracy of 64.11%. These accuracies indicate that it is possible to determine the likelihood of a malware family from the continuous variables. However, even using an FFNN, which can find complex non-linear relationships between the features, we obtained an accuracy of only about 64%, which means that clustering will be limited in its ability to separate the various malware families. However, we might have more luck separating the malware logs from normal logs.

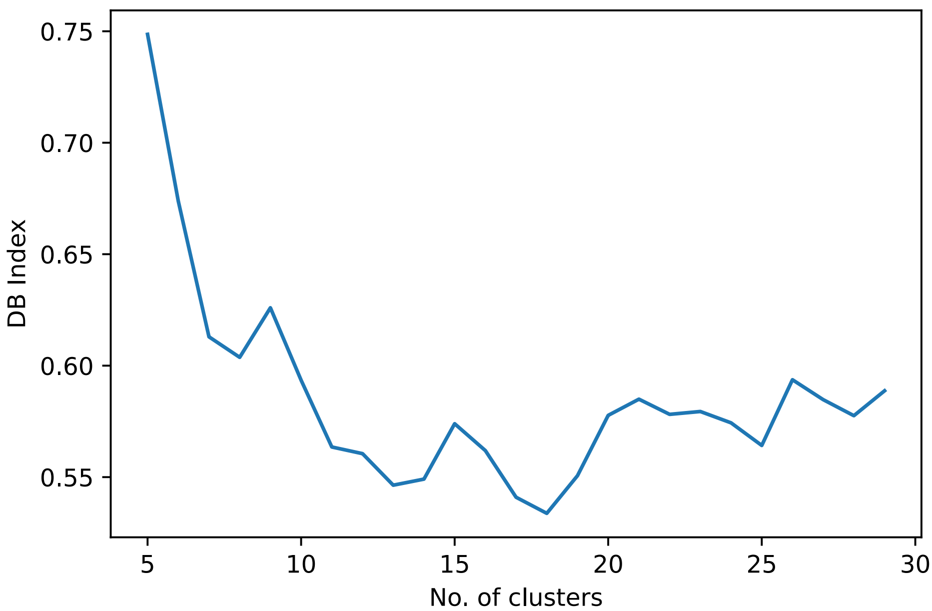
2 Results, Inferences and Analysis

2.1 Single Linkage Hierarchical Clustering & DB Index Validity Analysis

- This is what they did in paper.
- We are using Chi-Squared analysis to do feature selection first, we are taking top F features based on Chi-Squared. After iterating over all possible values of F, we got 6 as the most optimal values. We are taking top 6 features out of the total 17 features that we have. They are:

  ```
  DestIP, Dport, percent_of_established_states,
  inbound_pckts, outbound_pckts, fileno
  ```

- The inclusion of DestIP and Dport makes sense, as malware is likely to use a few malicious IP-Port connections to carry the majority of its traffic. Moreover, the volume of the incoming and outgoing packets will also expose unusual behaviour, as malware might send and receive data in atypical proportions. One surprising inclusion here is the fileno; it isn't immediately obvious that this would help determine whether a specific behaviour is normal or not.
- Now, we will come to the most important hyperparameter. i.e., the number of clusters (K). In the paper, they are using DB Index Cluster validity analysis. They are essentially taking the value of K for which the DB Score is minimum. Here is the graph we got for DB Index:



- The relationship between the DB Index and the number of clusters doesn't seem to be very smooth. As a metric for the purpose of choosing an appropriate number of clusters, DB Index fails here. We shouldn't rely entirely on DB Index Analysis, and should probably define more metrics for an objective and robust analysis of the produced clusters.
- Based on Figure 2.1, at K = 18 we got the lowest DB Index score. Here is the info about the 18 clusters:

```
Cluster # 0 , No of items in cluster: 14628
Malware Percentage: 74.03609515996719
Cluster # 1 , No of items in cluster: 28
Malware Percentage: 7.142857142857143
Cluster # 2 , No of items in cluster: 3
Malware Percentage: 0.0
Cluster # 3 , No of items in cluster: 10774
Malware Percentage: 57.89864488583627
Cluster # 4 , No of items in cluster: 6
Malware Percentage: 66.66666666666667
Cluster # 5 , No of items in cluster: 2
Malware Percentage: 50.0
Cluster # 6 , No of items in cluster: 4
Malware Percentage: 25.0
Cluster # 7 , No of items in cluster: 1
Malware Percentage: 100.0
Cluster # 8 , No of items in cluster: 348
Malware Percentage: 2.586206896551724
Cluster # 9 , No of items in cluster: 1
Malware Percentage: 100.0
Cluster # 10 , No of items in cluster: 7
Malware Percentage: 71.42857142857143
Cluster # 11 , No of items in cluster: 1
Malware Percentage: 100.0
Cluster # 12 , No of items in cluster: 1
Malware Percentage: 0.0
Cluster # 13 , No of items in cluster: 1
Malware Percentage: 100.0
Cluster # 14 , No of items in cluster: 3
Malware Percentage: 33.333333333333336
Cluster # 15 , No of items in cluster: 1
Malware Percentage: 100.0
Cluster # 16 , No of items in cluster: 1
Malware Percentage: 100.0
Cluster # 17 , No of items in cluster: 3
Malware Percentage: 0.0
ACCURACY (MALWARE vs. NORMAL):  67.64421028164104
```

- Out of the 18 clusters only the cluster 0, 3, 8 have a significant number of data points; the rest are almost empty. Figure 2.2, 2.3 show the split of different families/goals in those three clusters.

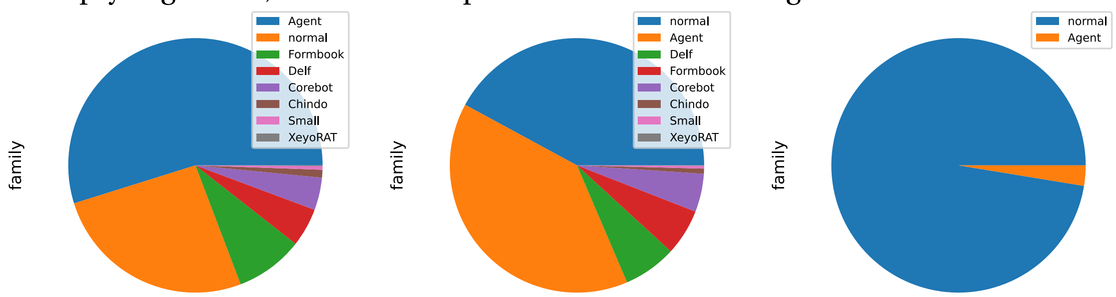


Figure 2.1: Share of different families in cluster 0, 3, 8.

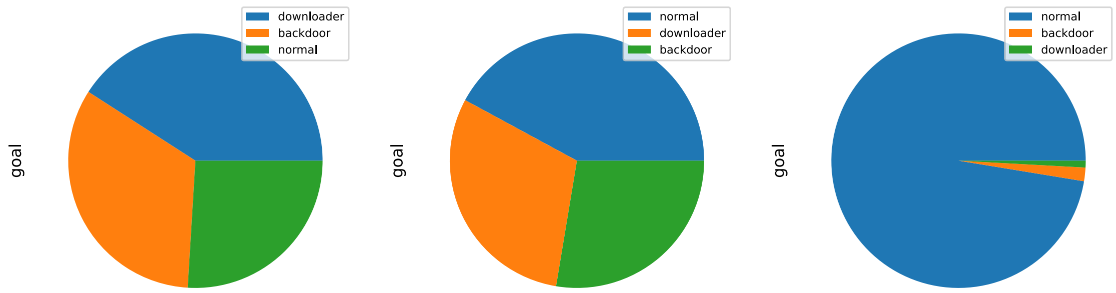


Figure 2.2: Share of different goals in cluster 0, 3, 8.

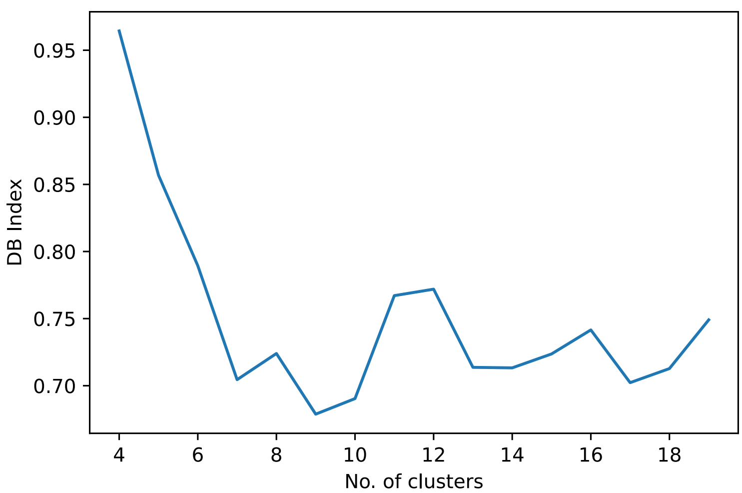
- Due to the lack of features other than flow statistics, we cannot do fine-grained clustering. Hence, separating the malware into families is not easy, which is evident from the pie diagrams. There is no dominant family/goal in most of the pie charts.
- Single linkage hierarchical clustering seems to be prone to giving vast importance to outlier points. This is motivation to try other versions of hierarchical clustering that prioritize denser clusters.

2.2 Ward Linkage Hierarchical Clustering

- Here, the Chi-Squared feature selection did not impact the accuracy. It gave around 1% improvement in accuracy. However, it improved the time of clustering as the number of features decreased.
- We are taking the top 6 features out of the total 17 features that we have. They are:

```
DestIP, Dport, percent_of_established_states,
inbound_pckts, outbound_pckts, fileno
```

- Instead of using DB Index cluster validity analysis, we went through several values of K and checked which values of k were good. Here, we present the results of 2 of those experiments for comparison: number of clusters - 4, 8. We do not need to use DB Index cluster validity analysis for our data set because we already know the labels of the malware, and it will be like a supervised algorithm. Even though DB Index does not know about the malware labels, it still managed to predict the value of K very closely. It gave K = 9 but using the labels given in the dataset, we got K = 8.



- **Number of Clusters = 4**

```
Cluster # 0 , No of items in cluster: 11583
Malware Percentage: 85.66001899335232
Cluster # 1 , No of items in cluster: 8765
Malware Percentage: 56.71420422133485
Cluster # 2 , No of items in cluster: 2360
Malware Percentage: 54.19491525423729
Cluster # 3 , No of items in cluster: 3105
Malware Percentage: 29.790660225442835
ACCURACY (MALWARE vs. NORMAL):  71.09595940030218
```

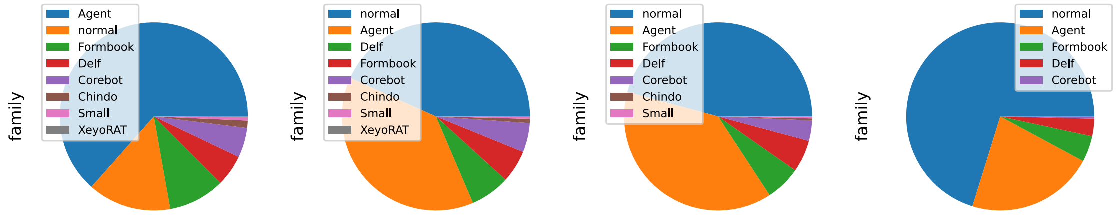


Figure 2.3: Share of different families in each of the 4 clusters.

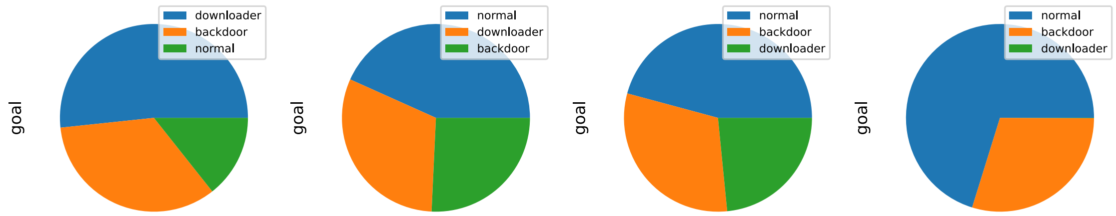


Figure 2.4: Share of different goals in each of the 4 clusters.

- **Number of Clusters = 8**

```
Cluster # 0 , No of items in cluster: 7023
Malware Percentage: 81.19037448383881
Cluster # 1 , No of items in cluster: 3105
Malware Percentage: 29.790660225442835
Cluster # 2 , No of items in cluster: 2360
Malware Percentage: 54.19491525423729
Cluster # 3 , No of items in cluster: 1972
Malware Percentage: 83.21501014198783
Cluster # 4 , No of items in cluster: 2588
Malware Percentage: 99.65224111282843
Cluster # 5 , No of items in cluster: 3233
Malware Percentage: 69.06897618311166
Cluster # 6 , No of items in cluster: 350
Malware Percentage: 3.142857142857143
Cluster # 7 , No of items in cluster: 5182
Malware Percentage: 52.624469316866076
ACCURACY (MALWARE vs. NORMAL):  72.36663696586992
```

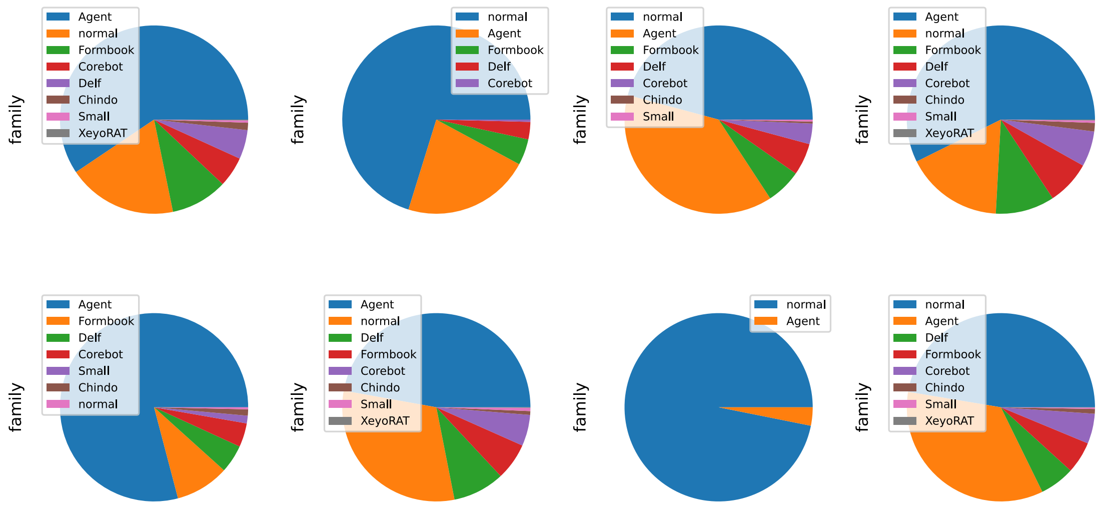


Figure 2.5: Share of different families in each of the 8 clusters.

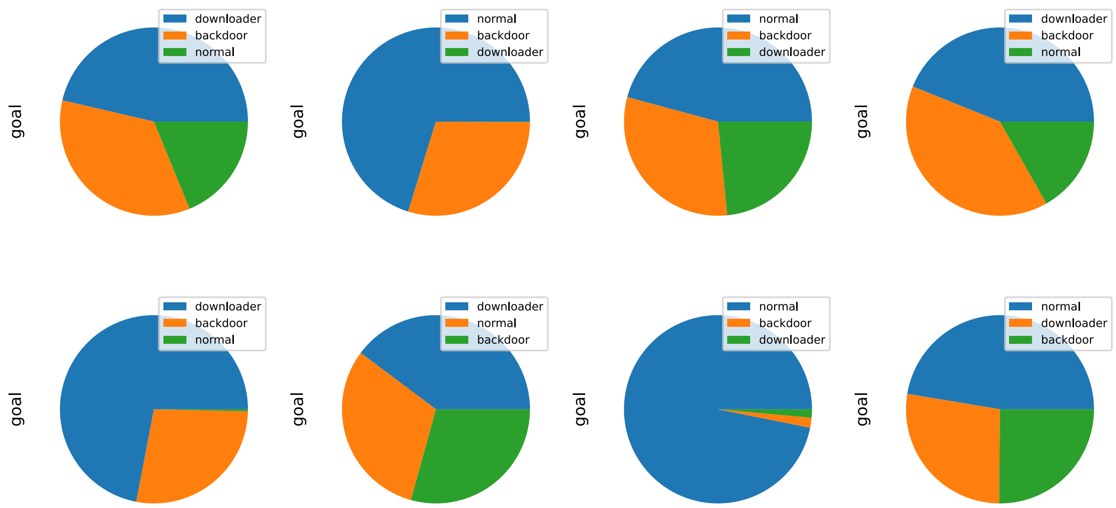


Figure 2.6: Share of different goals in each of the 8 clusters.

- There is not much improvement in the overall accuracy when we increase the number of clusters. However, k = 8 is better because of the better individual cluster performance. There are 6 clusters out of 8 in which the malware percentage is either too high or too low compared to the other two, consider cluster 4, in which we have the Malware percentage as 99.6%, so if something falls in cluster 4, we can be very sure that it is malware.

- In both of them, we cannot classify the malware as different families/goals due to the lack of features, this can be seen from the pie charts above.

2.3 K - Means

The first hyperparameter to set is the number of clusters. This changes for different clustering methods. For K-means clustering using all the continuous features, we can use an elbow plot, of the total sum of squared distances of the data points from the cluster centers against the number of clusters to choose an appropriate number of clusters. In this elbow plot, Fig 2.7, we observe that at 4 clusters itself, our improvised loss metric stops decreasing so rapidly. Hence, for k-means clustering, we set the num_clusters hyperparameter as 4.

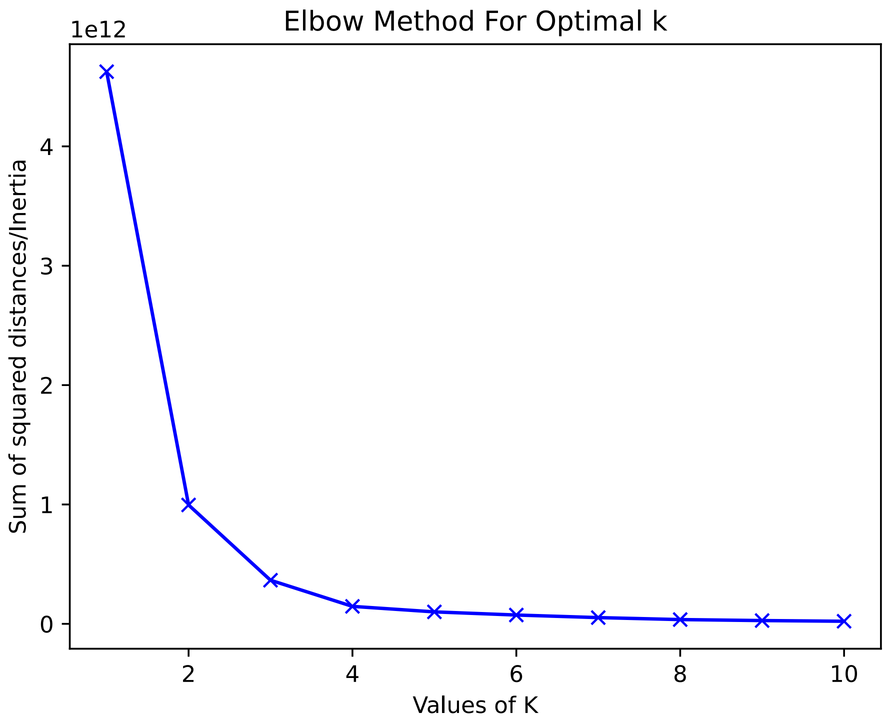


Figure 2.7: Elbow method for optimal K

```
Cluster # 0, No. items = 272 [Malware Percentage: 65.80882352941177]
Percentage of each goal of malware in this cluster:
backdoor      60.6618%
normal        34.1912%
downloader    05.1471%
Cluster # 1, No. items = 23126 [Malware Percentage: 64.1139842601401]
Percentage of each goal of malware in this cluster:
normal        35.8860%
downloader    34.4634%
backdoor      29.6506%
Cluster # 2, No. items = 1153 [Malware Percentage: 79.96530789245446]
Percentage of each goal of malware in this cluster:
backdoor      42.3244%
downloader    37.6409%
normal        20.0347%
Cluster # 3, No. items = 1262 [Malware Percentage: 92.63074484944532]
Percentage of each goal of malware in this cluster:
downloader    66.3233%
backdoor      26.3074%
normal        07.3693%
```

Figure 2.8: Cluster composition after performing k-means clustering

This is not optimal - almost all data points have gone to a single cluster. This might result from including so many unnecessary columns that the algorithm factors into its decision. These results were obtained using all features. Let's try doing features selection using Chi-Squared analysis.

```
Cluster # 0, No. items = 23192 [Malware Percentage: 63.905657123145915]
Percentage of each goal of malware in this cluster:
normal        36.0943%
downloader    34.3351%
backdoor      29.5705%
Cluster # 1, No. items = 275 [Malware Percentage: 66.18181818181819]
Percentage of each goal of malware in this cluster:
backdoor      59.6364%
normal        33.8182%
downloader    06.5455%
Cluster # 2, No. items = 1106 [Malware Percentage: 82.27848101265823]
Percentage of each goal of malware in this cluster:
backdoor      43.4901%
downloader    38.7884%
normal        17.7215%
Cluster # 3, No. items = 1240 [Malware Percentage: 95.48387096774194]
Percentage of each goal of malware in this cluster:
downloader    68.1452%
backdoor      27.3387%
normal        04.5161%
```

Figure 2.9: Cluster composition after performing k-means clustering with chi-squared feature selection

Even after doing Chi-Squared feature selection, it didn't improve the clustering accuracy much in K-Means, although it should be noted that in all but the largest cluster, the malware percentage increased slightly. In the largest cluster, the malware percentage almost remained the same.