THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

EVALUATION OF UNSUPERVISED FEATURE SELECTION AND CLUSTERING ALGORITHMS FOR NETWORK TRAFFIC ANALYSIS

STEVEN COULTER
Spring 2010

A thesis
submitted in partial fulfillment
of the requirements
for baccalaureate degrees
in Computer Engineering
and Mathematics
with honors in Computer Science

Reviewed and approved[1] by the following:

George Kesidis
Professor of Computer Science and Engineering
Thesis Supervisor

David Miller
Professor of Electrical Engineering
Thesis Supervisor

Lee Coraor
Associate Professor of Computer Science and Engineering
Honors Adviser

[1]Signatures on file in the Schreyer Honors College.

Abstract

Proper identification of network traffic is an essential component of network administration and must be performed in an efficient and accurate way. Due to issues of cost, complexity, inaccuracy, and time it is often times infeasible to place effort into separating network traffic data into different labels and classifications. Instead, methods can be devised to create meaningful groupings of network traffic with no predetermined knowledge of the traffic aside from purely statistical data of the communications sessions themselves.

Clustering is an approach to accomplish this, and it does so by developing a measure of similarity over several of the statistics, or *features*, of each communication session. For the same reasons of cost and complexity, in addition to accuracy, it is desirable to have a way to extract those statistics that are the most meaningful. This paper discusses several means of extracting the most discriminating features from an unlabeled set of data, while measuring how well those features perform on a set of labeled data where a measure of accuracy can be had.

The variance that a feature exhibits over a set of data is a measure that can be exploited to develop a means of extracting meaningful features. These features can be found using either an iterative method, involving steps each with one feature removed, or by developing a more advanced measure of feature similarity. Most importantly, this paper shows that certain features, such as the ratio of maximum to minimum payload

size, are rated as highly discriminating features both across different selection methods and across data gathered from very different locations.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Problem Description

Generating an accurate network traffic profile is useful in a diversity of applications. For the administration of a network, a traffic profile is useful for maintaining the *quality of service*, QoS, as well as content monitoring, network forecasting, and identifying the impact of new applications entering the network. On the security side, traffic analysis with the intent of anomaly detection provides the means of a *Network Intrusion Detection System* (NIDS) including a botnet detection scheme, as described in (1). Recent developments in network applications, both useful and malicious, have increased the challenge of correctly identifying and classifying traffic patterns. Advances in P2P, VoIP, and streaming multimedia applications coupled with their ease of availability and more frequent usage have made it difficult to understand the core network traffic in order to preserve the QoS. For reasons necessary and often intentionally deceitful, many network applications use masquerading schemes, tunneling, and non-standard or dynamic port numbers for purposes of both communication and detection avoidance, demanding a transition from classical port-based analysis to a more sophisticated method. Clustering algorithms, designed to partition data into different groupings based on a measure of flow similarity, provide a means of generating a network traffic profile.

1.2 Netflows

Network flows are a statistical resource useful for classification purposes. A flow is an aggregate record of statistical information about a communication session between end devices. Cisco has developed a methodology of recording and reporting network flows, called *NetFlow*, described in (2). Cisco has allowed third parties, both commercial and free-ware, to take advantage of the NetFlow system, enabling the majority of routers currently in use to be capable of recording NetFlow statistics. The availability of NetFlows makes them both an effective and easy to acquire resource.

A NetFlow is identified by a 5-tuple of packet attributes defined as (*source address, destination address, source port, destination port, layer 3 protocol*). NetFlow capable routers record information about the flow and export the data to a collection server when the flow is deemed finished. The termination of a flow can be determined by a timer measuring either a period of inactivity time, useful for short flows, or activity time for longer flows. The collection server has the ability to merge the flows that have been received due to expiration of the activity timer.

NetFlow devices are commonly placed along central links in order to capture an inclusive sample of the network traffic. For business use, the flows may be captured at the gateway in order to record statistics of all network traffic entering and leaving the building. A sample of the NetFlow features is available in Table 2.1.

1.3 Current Methods of Analysis

As an alternative to the naive and often incorrect port based classification or a detailed per-packet payload analysis that is both computation and time intensive, methods can be applied to the statistical information provided in or computed from network flows. One advantage of flow analysis is that the computational resources, other than the flow gathering mechanisms themselves, may be entirely disjoint from the network in question. The analysis considers several different statistics of the flow and by some certain means combines flows that have similar features. This process may be iterative, either pairing two flows together or adding one flow to an existing group each iteration, and then recomputing the flow similarity or distance measures. Another approach is to calculate a probability distribution for each data class, and determine the chances of a particular flow belonging to a particular grouping. The result of these approaches are distinct groupings, or *clusters*, of flows. If the result of the method can be considered correct, each cluster will uniquely identify a particular application class of network traffic. These methods are referred to as clustering schemes, and the remainder of this section will discuss their use with unlabeled and labeled data sets.

1.3.1 Unsupervised Clustering

Unsupervised procedures are used in data sets where the class labels are not known. An introduction to unsupervised learning techniques is discussed in (3) and is summarized here. Not only does the unsupervised method eliminate the cost of labeling

the data set, it is also somewhat more dynamic than the supervised approach. For example, a supervised method devised and not updated before the advent of P2P applications would not have a proper classification for this traffic, since P2P would not have been in the original label set. The unsupervised method, without its reliance on the labels, would be able to identify this and other new types of traffic, assigning them to distinct clusters. One implication is that the unsupervised method may produce superior results if used for anomaly detection purposes. Though the clustering method is not aware of and does not use the categorizations of the data, the unsupervised approach can still be performed on a labeled data set allowing for a judgment of correctness of the algorithm. As discussed in Chapter 2, several varieties of unsupervised clustering have been developed.

1.3.2 Supervised Classification

Supervised classification relies on a class labeled data set to define the clustering rules. Due to the expense incurred when labeling the entire data set, usually only a small labeled subset of data, referred to as a *training set*, is used for developing the classification criteria. These methods look at the data set, in this case a set of network flows labeled by application class, and attempt to develop a means of computing the probability that a certain flow belongs to a certain class, incorporating preference to those flow features the algorithm determines to be most significant.

In the case of network traffic analysis, labeling the data sets is often a difficult and time consuming task. As discussed, port based classification fails when applications use non-standard port numbers, and payload inspection or hand-labeling data sets is

quite tedious and often incorrect. These methods are also complex in storage: a diverse sample may include several days of sampling which, with captured payloads, becomes very storage intensive. Automated packet payload inspection and classifications do exist, however, making the supervised approach applicable in these cases.

Chapter 2

Unsupervised Clustering

The goal of any clustering algorithm is to partition data as accurately as possible into distinct groupings of data points with similar intra-group characteristics and a degree of inter-group dissimilarity. Every data point used within in the clustering algorithm consists of many different values of *features* of the data. Each resultant grouping of data points of the algorithm is referred to as a *cluster*.

Several different metrics can be used to develop measures of similarity and dissimilarity between the data points. Unsupervised clustering exclusively uses these metrics on the feature values of the data points to form the groupings, having no knowledge on what classifications the points belong to. If the data is labeled, some validation may later be performed on the outcome of the algorithm.

The primary components of a clustering algorithm are outlined as follows:

1. *Feature Selection.* Before clustering can be performed, a subset of distinguishing features must be selected from some group of candidate features. Often times the set of available features is quite large, making a computation involving the entire set infeasible. Features may also be *extracted*, meaning that a new feature is created by mathematically combining several existing features. Proper selection of features is crucial to the performance of the clustering algorithm. Selected features should be easily able to distinguish patterns in the data points. The

features themselves should also have some measure of dissimilarity between them, to reduce computational overhead incurred by using redundant features.

2. *Clustering Method and Metric Selection.* Multiple unsupervised clustering methods exist, ranging from a simple nearest-neighbor method to least squares error based approaches. The nature of a scheme usually goes hand in hand with the metric selection. The metric must be able to compare the distance or similarity of different points withing the data set. A *distance* metric is usually a measure on some continuous data, while *similarity* metrics are useful for qualitative data. A second consideration is the scaling or normalization of the data set. Consideration should be taken that information is not lost under the function of the metric upon the scaled interval, and that the combination of the algorithm and metric is a proper solution for the problem at hand.

3. *Cluster Validation.* Regardless of the choice of metric, any clustering method will be able to divide a data set into different partitions, making it necessary to develop measures of cluster comparison and stopping rules for the algorithm. In order to be considered reliable, these methods should have no preference to the metric or algorithm used, and certainly have no reliance on the data labels. Validation criteria may be placed within the clustering algorithm for useful purposes such as determining a merging criterion between clusters or stopping the algorithm when the best number of clusters has been reached.

4. *Results Interpretation.* Once the clustering algorithm has created the groupings of data points, some meaningful insights should be able to be gathered from the data.

If unsupervised methods were performed on a labeled set, observing the placement of the classifications provides a measure of accuracy on the method. Clusters that are very large or very small may be further explored to discover strengths or weaknesses of the approach used. A visualization of the clustering results may also be graphed or plotted for further understanding.

2.1 Feature Selection and Extraction

Deciding which features, extracted or otherwise, to be used in order to determine the proper dimensionality of the system is the primal problem in any clustering method. Features may either be *selected* or *extracted*. Selection refers to picking a feature from the data provided feature set to be used for analysis. Feature extraction, on the other hand, refers to the creation of new features by combining, arithmetically or otherwise, values of features currently in the data set. For netflow data, an excerpt of the features is listed in 2.1. Some simple examples of features extracted from this excerpt include flow time T, byte rate R_b, packet rate R_p given by the following:

$$T = \text{LAST\_SWITCHED - FIRST\_SWITCHED}$$

$$R_b = \frac{\text{OUT\_BYTES}}{T}$$

$$R_p = \frac{\text{OUT\_PKTS}}{T}$$

Because the clustering algorithm must compute distances between clusters for each feature, reducing the number of features drastically reduces the runtime of the

Table 2.1: Net flow Statistics Excerpt

Feature Name	Description
MIN_PACKET_LENGTH	Min. packet length
MAX_PACKET_LENGTH	Max. packet length
OUT_BYTES	Total bytes in flow
OUT_PKTS	Total packets in flow
FIRST_SWITCHED	Time of first packet
LAST_SWITCHED	Time of last packet

algorithm as the number of features used directly defines the dimensionality of each feature. Although features can be naively selected from the flow statistics, more advanced approaches are often beneficial. The features selected have a large and direct impact on the clustering results. Features must be able to discriminate well amongst the data while maintaining a level of dissimilarity amongst themselves. For these reasons, feature selection has developed into its own class of algorithms. An introduction to feature selection is provided in (6), and the general methods are discussed below.

1. *Wrapper-Based Selection.* A wrapper based feature selection method utilizes a learning mechanism to decide the usefulness of a feature. Wrapper methods usually are computationally intensive, but provide an easy and universal means of feature selection. A wrapper method is used in Chapter 4 to select features.

2. *Nested Subset Methods.* Nested subset models investigate the difference of the value of the objective function resultant from the inclusion or exclusion of a feature during each iteration of a clustering algorithm. Predictions are obtained by measuring finite differences or developing approximations to the objective function.

3. *Direct Objective Optimization.* Direct methods attempt to determine and optimize the objective function of variable selection, with the goal of being a best-fit approximation to the data while reducing the number of features as much as possible. Linear prediction model adding a penalty term to the objective function are useful for this purpose.

2.2 Distance and Similarity Metrics

A distance metric is a function d satisfying the following conditions between two points x and y:

$$d(x, y) \geq 0$$

$$d(x, y) = d(y, x)$$

$$d(x, y) \leq d(x, z) + d(z, y)$$

$$d(x, y) = 0 \text{ iff } x = y$$

Distance functions are used to measure the closeness of continuous features. In the case of qualitative data, a similarity measure is instead used. Like the distance metric, there are some conditions that must be satisfied by the similarity measure:

$$s(x, y) = s(y, x)$$

$$0 \leq s(x, y) \leq 1$$

$$s(x, y)s(y, z) \leq [s(x, y) + s(y, z)]s(x, z)$$

$$s(x, y) = 1 \text{ iff } x = y$$

Some of the most common distance metrics are described in Table 2.2, while many more are defined in (7). On diverse data sets, a standard and perhaps too easy approach is Euclidean distance on the data scaled to the interval $[0, 1]$, however this approach causes a loss of information within the data point, and ultimately the result of the clustering scheme. Certain metrics, like the *Maximal Information Compression Index* discussed in Section 4.7, are sensitive to scaling and limit the loss.

Table 2.2: Distance Metrics

Metric	Common Name	Computation
l_1	Manhattan Distance	$\sum_{i=1}^{n} \lvert x_i - y_i \rvert$
l_2	Euclidean Distance	$\left(\sum_{i=1}^{n} (x_i - y_i)^2 \right)^{1/2}$
l_∞	Supremum	$\max_{i=1}^{n} \lvert x_i - y_i \rvert$

A structure called a *proximity matrix* is used to store the distances between the nodes, where the entry $a_{ij} = d(i, j)$, defined by the metric selected. Because of the symmetry of the measure, only a triangular matrix is required for storage.

2.3 Clustering Algorithms

Many clustering algorithms exist, and each serve different purposes well suited to the field in which they were developed. A discussion on the different clustering approaches is well covered in (7) and (8), and two methods that will be used for experiments in this paper are covered in this section.

2.3.1 Hierarchical Clustering

An *agglomerative*, or bottom-up, Hierarchical Clustering (HC) approach begins with each of the n data points belonging to its own singleton cluster. On each iteration a *merge* of two clusters takes place based on a distance measure between the clusters. Because each cluster contains several sets of data points, some rules must be developed on how inter-cluster distance is measured. These are called *linkage* models, and the most common ones are listed in Table 2.3.

Table 2.3: Linkage Models

Name	Inter-Cluster Distance
Single	Nearest points in different clusters
Complete	Farthest points in different clusters
Average	Midpoints of different clusters

On each iteration, the two nearest clusters according to the linkage method and proximity matrix are combined, The proximity matrix is then recomputed, and the next iteration begins. The algorithm completes when all of the data points belong to a single cluster. Although a top-down, or *divisive*, method beginning from one large cluster containing all points can be used, this is often too computationally intensive to be used in practice. (7) provides a more detailed view of Hierarchical Clustering, while (9) provides an easy to use algorithm with visualizations. A general outline of the Hierarchical Clustering is given as follows:

1. Create n singleton clusters for each of the n data points
2. Compute distances (in the selected metric) between all clusters and store in a proximity matrix
3. Merge the two nearest clusters into a new cluster.
4. Update distances between this newly formed cluster and all other clusters in the proximity matrix.
5. Repeat from Step 2 until all data points are within one cluster,
   or the desired number of clusters has been attained.

Once the clustering method is completed, the result can be viewed as a *dendrogram*. Figure 2.1 displays a sample dendrogram. The horizontal axis depicts the different data points, while the vertical axis measures the relative distance or similarities between each level of the hierarchy. Determining a proper number of clusters, in other

words making cuts at different levels of the dendrogram, is crucial to the validity of the method.

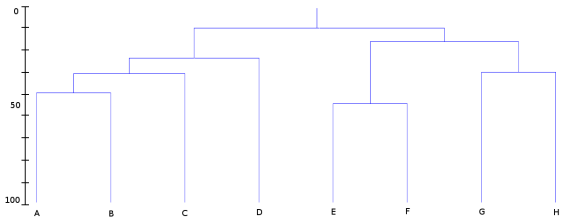


Fig. 2.1: Sample Dendrogram

2.3.2 k-Means Clustering

In contrast to a hierarchical clustering scheme, the k-means algorithm provides a *partitional* clustering outcome, attempting to minimize the sum of squares error. A *prototype* matrix is used to store the centroid of each cluster. During an iteration of the algorithm, objects are assigned to the nearest cluster, and the prototype matrix is recomputed. The algorithm is as follows:

1. Create a prototype matrix M from an initial scheme of k partitions.
2. Assign each data point to the nearest cluster.
3. Recompute the prototype matrix.
4. Repeat from Step 2 until no changes occur in the assignment of data points to their respective clusters.

Though the k-means is an easy algorithm to implement, it does contain some drawbacks. A large question is how to define the initial partitioning of the data set. Though there

is no universal method to determine the initial partitioning scheme, a computationally extensive determination could be made by running the algorithm on many random initial partitioning. Another issue is that during each iteration all data points are assigned to clusters, making the algorithm sensitive to noise. Even if a data point is not near to any cluster, it still must assigned to a cluster, skewing the centroid in the next iteration. Some variation on the parameter k may be useful solution. (7) writes in more detail about k-means clustering.

2.4 Cluster Validation Criteria

2.4.1 Intra-Algorithm Rules

During the merge operation between different clusters in algorithms such as HC, two questions arise:

- Are the clusters similar enough that they should be merged?
- Has the most most meaningful number of clusters been reached?

Within the clustering algorithm, certain validation criteria are able to these answer. It is challenging to develop a universal model to for this purpose, and for that reason a wealth of algorithms have been devised for both of these purposes. (10) describes and compares 30 of these methods, and the top three are discussed below and summarized in Table 2.4,

1. *Calinski and Harabasz*, The C&H method is develped in terms of the n data points and the k clusters created. Two matrices, B and W, consist of the cluster sum of

squares and cross products, respectively. Taking the C&H value over all levels of the hierarchy, the level of maximal value is selected as optimal.

2. *Duda and Hart*, The D&H method is a ratio between a two- and a one-cluster scheme of the sum of squared errors. This method is applied during the merge operation. Two clusters are allowed to be merged only if their D&H value meets a certain threshold.

3. *C-Index*. The C-index is a measure taken on the distances of points within a cluster. Taking the C-index over all levels of the hierarchy, the level with the minimal value is selected as optimal.

Table 2.4: Cluster Validation Criteria

Method	Description	Optimal Result
C & H	$\frac{\mathrm{Tr}(B)(n-k)}{\mathrm{Tr}(W)(k-1)}$	Maximal[1]
Duda & Hart	$Je(2)/Je(1)$	Merge Operation
C-index	$\frac{d_w - \min(d_w)}{\max(d_w) - min(d_w)}$	Minimal[1]

1. Taken over all levels of hierarchy.

2.4.2 Results Comparison

For labeled data, a metric is inherently present on the clustering outcome by observing the amount of same-class flows located within each resultant cluster. The

class purity of the flows can then be taken as some measure of accuracy over the entire scheme by averaging the per-cluster purity over the entire set.

It becomes a challenge to compare the results of different clustering outcomes on unlabeled data as the lack of labels do not allow for a measure of accuracy. However, partitions can still be compared by measuring the movement of the data points between the schemes. The Rand Index, discussed in (11), was originally developed in 1971 and over the years has developed several enhancements, including chance correction. Though extraneous details will not be covered here, the Rand index is a way of comparing two different partitioning schemes of the same data set, based on the relationship of data points and cluster membership. Let A and B represent two different partitioning schemes of the data set. These partitioning schemes would have been the result of some clustering method, and may contain a different number of clusters, with data points belonging to different clusters in each partitioning scheme. The Rand index classifies the movement of data points between the set into four categories. These are labeled as agreements or disagreements, or A and D, respectively.

2.5 Graphical Representation

Once the cluster algorithm has completed, a visualization of the resultant clusters can provide insight, comparison, and validation of the clustering method. A multi-dimensional technique, star coordinates, is described in (12). The basic star coordinate system arranges n equal length vectors meeting at the origin in a circular formation. These vectors are represented as the coordinate axes. Scaling and rotation transformations are possible to modify the axis length, based on their contribution to the data, or adjust the angle between axes, based upon their correlation. A point in this n-dimensional space is mapped into a 2-D Cartesian plane by summing the vectors along each of the n axes, determined by the following formula

$$(x, y)_j = \left(\sum_{i=1}^{n} v_{xi}(d_{ji} - \min_i), \sum_{i=1}^{n} v_{yi}(d_{ji} - \min_i) \right) \tag{2.1}$$

Where $v_i = (v_{xi}, v_{yi}), i = 1, ..., n$ is the Cartesian representation of the ith axis vector, Dj represents the n-dimensional data point, and $\min_i$ the smallest ith coordinate of all points in the set.

$$D_j = (d_{j0}, d_{j1}, ..., d_{jn}),$$

$$\min_i = \min\{d_{ji}, 0 \leq j \leq |D|\}$$

(13) describes the application of star coordinates to the visualization of network traffic analysis. Each discriminator of the flow used is one of the axes of the star plot. Some initial feature is placed on the plane where the positive x axis would lie in the Cartesian system. Size based discriminators are placed in the upper plane, with time based features

comprising the lower plane. Both the scaling and rotation transformations are applied: the axis length is modified to reflect the feature weight, the angle between features is calculated based upon the feature correlation, by use of the correlation coefficient.

Though a star cluster visualization may appear a very good method to witness the results of a clustering outcome, this method does include some drawbacks, particularly in the axis alignment when using a measure of variance, such as the correlation coefficient, to determine the axis angle. It may be possible for features to have a high degree of correlation globally, grouping the axes close together, but the feature may have a high degree of local variance, making it a good discriminator within the clusters. In this scenario, clusters along two close axes may appear as a globular feature but still retain a high level of dissimilarity.

An example of star coordinates for clustering display purposes is shown in Figure 2.2, Figure 2.3, and Figure 2.4. Figure 2.2 details the axis labeling for the particular set of features used in this example, with the time and size constraints in the appropriate half plane. Figure 2.3 shows a 5-cluster sample of the data set displayed on the star map. Here, the clustering is quite obvious, seen by the grouping of the points. Figure 2.4 is the complete clustering output with the axes titles removed. The on-line version of this document contains a colored version of these charts, with each cluster represented by a unique symbol color combination.

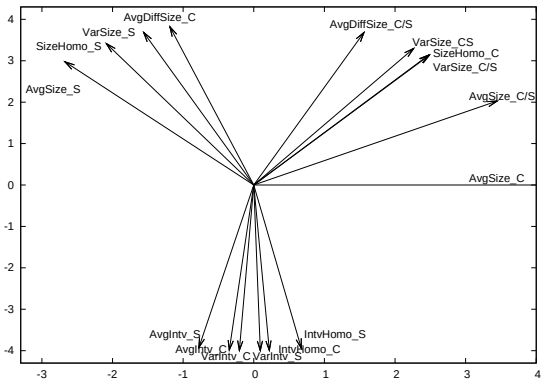


Fig. 2.2: Star Coordinate Axes Labeling

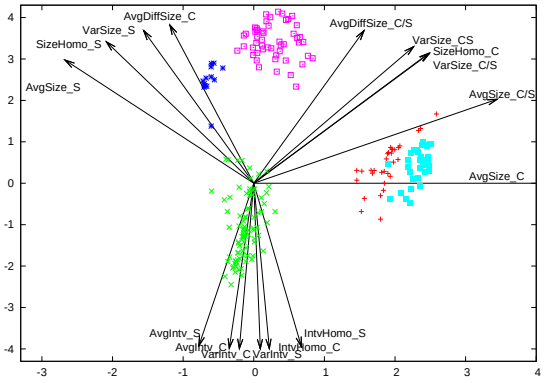


Fig. 2.3: Five Sample Clusters

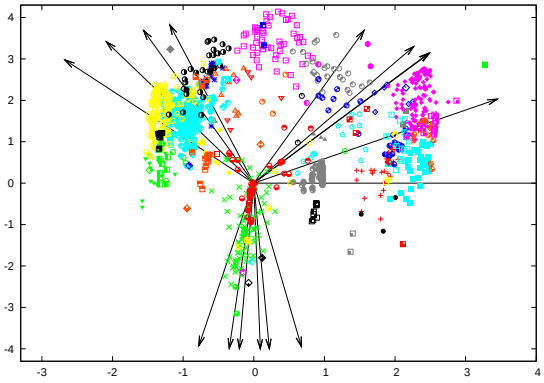


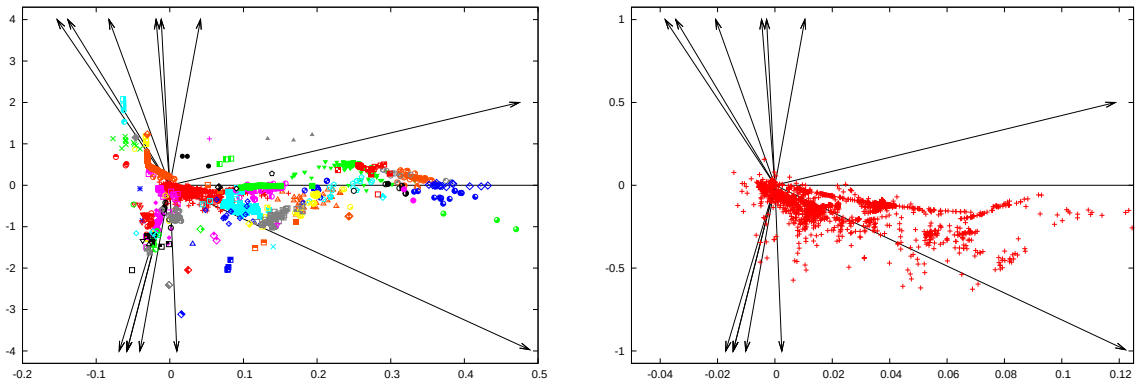
Fig. 2.4: Complete Clustering Output

Chapter 3

Feature Selection: A Bad Example

3.1 The Need for Proper Feature Selection

A simple preliminary experiment can easily show the need for proper feature selection. Performing the HC algorithm at a level of 80 clusters on the Cambridge trace, using the initial feature set of Table 4.4, the resultant scheme is depicted in Figure 3.1a. Upon closer inspection, it can be observed that over half of the data points were grouped within a single cluster, shown in Figure 3.1b. Supervised classification by flow amounts labels this as a web traffic cluster that has a purity of only 28.7%.



(a) Clustering Output (b) 3,392 Flow Cluster

Fig. 3.1: HC Output on Cambridge Data With Initial Feature Set

This "*black hole*" effect is a indicator that the clustering scheme is performing poorly. The data points within this cluster are detailed in Table 3.1. The algorithm has granted cluster membership to large portions of the WWW, FTP-PASV, DBASE and SERVICES classes. The evolution of this cluster throughout the HC process can be witnessed in Figure 3.2. Pausing the algorithm at different levels of N clusters, the images best showing the growth pattern were selected. For each level, the cluster selected to be displayed was the largest and most diverse ancestor of the original *black hole* cluster shown in Figure 3.1b. The n_i value represents the number of flows in each cluster.

In the resultant scheme, the next largest cluster in the scheme consists of 800 flows of MAIL and FTP-DATA in a nearly equal distribution. All other clusters have less than 350 flows and relatively low error rates. This is a preliminary indicator that splitting these large and diverse clusters may lead to a more accurate scheme. For $N \lesssim 80$, the primary testing range of the experiments of this paper, the HC algorithm did no further merges on the black hole cluster until the final few levels.

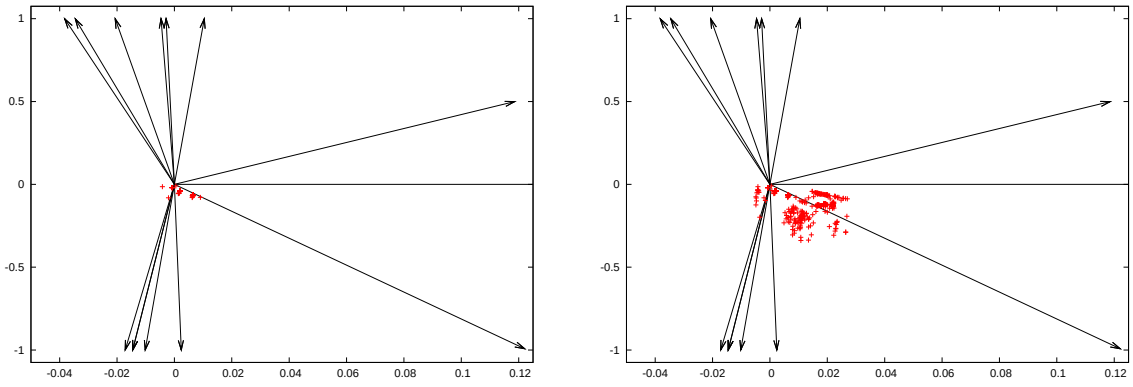
One cause of this effect can be identified as a poorly discriminating, or *noisy*, feature. Because of the nature of HC, the noise may continually draw in new flows to some particular cluster, compounding the error rates in higher levels of the hierarchy. In fact, with the removal of just one noisy feature, AvgIntv_S, a noticeable size reduction can be seen. For this 13-feature scheme at $N = 80$, two large clusters exist. MAIL and INTERACTIVE have been filtered out of these clusters, The WWW, SERVICES and DBASE classes are now mostly unique to one of the clusters, while the P2P and FTP-CONTROL flows have been split between them. This reduced feature set scheme had an

overall weighted purity error rate of 34.7%, compared to the 49.9% of the original. The two new large clusters are detailed in Table 3.2.

Feature reduction can further be exploited to provide even more accuracy to HC clustering results, while keeping near to the same number of clusters. As in the example above, by elimination of noisy features the class divisions between the clusters become more distinct. A similar result can be achieved by selection of only the most discriminating features. The problem that arises here is how to correctly identify either noisy features or well discriminating features from an unlabeled data set. Several approaches to this problem are discussed in Chapter 4.

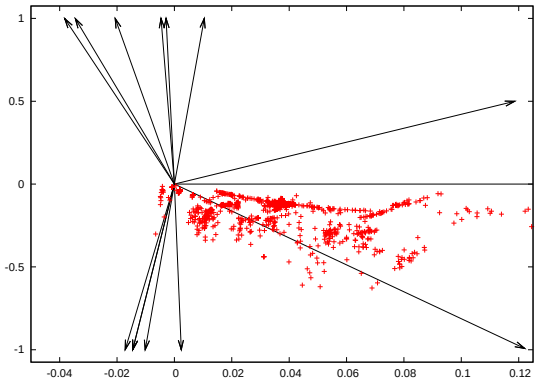
Table 3.1: 3,392 Flow Cluster Makeup

Flows	Class	% of Sampled Flows of Same Class
972	WWW	97.2
595	MAIL	59.5
155	FTP-CONTROL	31.0
342	FTP-PASSV	68.4
91	ATTACK	18.2
279	P2P	55.8
363	DBASE	72.4
0	FTP-DATA	0.00
55	MULTIMEDIA	11.4
500	SERVICES	100.0
39	INTERACTIVE	34.2
2	GAMES	25.0

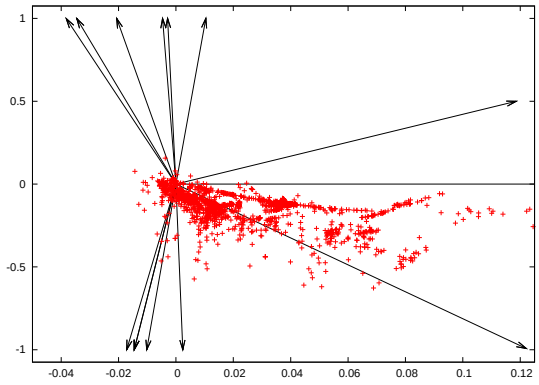


(a) $N = 440, n_i = 570$

(b) $N = 180, n_i = 837$



(c) $N = 120, n_i = 2,267$



(d) $N = 80, n_i = 3,392$

Fig. 3.2: Evolution of Black Hole Cluster

Table 3.2: Large Clusters of Reduced Feature Set

Flows	Class	% of Class
922	WWW	92.2
17	MAIL	1.7
125	FTP-CONTROL	25.0
358	FTP-PASSV	71.6
67	ATTACK	13.4
116	P2P	23.2
1	DBASE	0.0
0	FTP-DATA	0.0
18	MULTIMEDIA	3.
0	SERVICES	0.0
35	INTERACTIVE	3.1
0	GAMES	0.0

(a) 1,659 Cluster

Flows	Class	% of Class
1	WWW	0.1
1	MAIL	0.1
66	FTP-CONTROL	13.2
47	FTP-PASSV	9.4
5	ATTACK	1.0
161	P2P	32.2
359	DBASE	71.8
0	FTP-DATA	0.00
27	MULTIMEDIA	5.6
498	SERVICES	99.6
5	INTERACTIVE	4.4
2	GAMES	25.0

(b) 1,172 Flow Cluster

Chapter 4

Experimentation

4.1 Data Sets

For the purposes of cluster validation a labeled data set was the primary focus of the experimentation. This labeled trace was gathered within an enterprise network and was made available by the University of Cambridge (14). Once the methods were validated on the labeled data set, the same experiments were then performed on an unlabeled backbone trace provided by the Measurement and Analysis on the WIDE Internet group (MAWI).

4.1.1 Cambridge Trace

The primary data set used in this analysis was made available by the University of Cambridge and is available online (14) with several supporting documents. Network traffic, with full payload, was captured by a passive network monitor (15) capable of multi protocol recording without information loss. The monitor was placed within a 1,000 user facility and captured data in a full duplex trace for 24 hours. An iterative content-based classification process (16) was used to label the captured data. After the packets have been aggregated to flows, the process uses nine different identification methods, each applied to certain areas of the flow, and is then processed through several validation mechanisms. The 12 resultant classes of this method are listed in Table 4.1.

Table 4.1: Cambridge Classifications and 6,104 Selected Sample Flows

Classification	Example Protocol	Total Flows	Sampled Flows
FTP-CONTROL	ftp control	3343	500
FTP-PASV	ftp pasv	3383	500
FTP-DATA	bulk ftp data transfer	6326	500
DATABASE	sqlnet, oracle, others	2943	500
INTERACTIVE	telnet, ssh, rlogin	114	114
MAIL	POP, IMAP, SMTP	30326	1000
SERVICES	UDP services	2220	500
WWW	Web traffic	343689	1000
P2P	KaZaA, GnuTella, Bittorrent	2391	500
ATTACK	Botnets, worms, virus attacks	1792	500
GAMES	Online gaming	8	8
MULTIMEDIA	Streaming audio and video	575	482

The authors provide the full data set but due to time and computational constraints only a diversity sample set consisting of 6,104 flows was selected, using a percentage of the most common application classifications and the entirety of the least common classifications. A total of 6,104 flows were selected from the full data set to form the experimental data set. A representative number of flows was selected for each class, and a random selection of flows throughout the data set was used to meed this. The number of sample flows for each classification is listed in Table 4.1 along with some protocols representative of traffic belonging to that classification.

4.1.2 WIDE Trace

The WIDE project, maintained by the MAWI Working Group provided an unlabeled trace that was used for experimentation. The MAWI Group provides several backbone traces online, made available at (17). The trace selected for the purposes of this experiment was a 15 minute capture taken on March 2nd from the WIDE Internet

backbone. This trace contains 9,181,454 packets comprising 538,629 flows. Flows under 100 packets were considered insignificant and removed, resulting in a data set containing 1,845 long flows.

No sophisticated classification algorithms were performed on the WIDE data set, however a naive port-based classification was developed. Although the reliance on port numbers for classification purposes is questionable, some common ports may provide insight on the accuracy and comparisons of different clustering schemes. Table 4.2 lists the port-based classifications used on the WIDE trace. The classes consist of most common ports present in the flows. If a flow had a source or destination port matching distinct classes, it was entered into the class determined by it's destination port.

Table 4.2: WIDE Port-Based Classifications

Name	Flows	Ports
MAIL	91	25,109,110,143,220,465,578,993,995
WEB	1105	80,88,443,8000,8008
7700	207	7700
DNS	75	53
OTHER	367	

4.2 Feature Sets

A previous analysis on the long flows of the WIDE data by (13) included a brief study of feature selection. Ranking features including first and third quartiles were first removed, as well as those features that were recriminates of other features, reducing the set to 42 features. (13) then performed HC Clustering on the remaining features using the correlation coefficient between features as a measure of feature similarity. The

results were compared with the k-means feature similarity measure discussed in (18) and Section 4.7, and a final set of 16 features was selected. With the addition of the byte and packet counts, these features are listed in Table 4.3. The details _C and _S denote the origin as client or server, respectively.

Table 4.3: WIDE Features Set

Feature Name	Feature Details
Total Bytes	-
Total Packets	-
AvgSize	_C, _S, _C/S
VarSize	_C, _S, _C/S
SizeHomo	_C, _S
AvgDiffSize	_C, _S
AvgIntv	_C, _S
VarIntv	_C, _S
IntvHomo	_C, _S

From the 250 candidate features of the Cambridge data set, a subset was selected to match those features of Table 4.3. Due to lack of timing information, the IntvHomo and AvgDiffSize features were excluded from the Cambridge Set. The selected Cambridge features are listed in Table 4.4 along with their common name to those features in the WIDE trace. The Cambridge trace assigns the machines a and b designations rather than client and server.

Table 4.4: Cambridge Features Set

Feature Name	WIDE Naming Convention	Feature Details
actual_data_bytes	Total Bytes	ab, ba
mean_data_IP	AvgSize	ab, ba, a/b
var_data_IP	VarSize	ab, ba, a/b
max_data_IP/min_data_IP	SizeHomo	ab, ba
med_IAT	AvgIntv	ab, ba
var_IAT	VarIntv	ab, ba

A detailed description of the feature details are available in Appendix A, and a descriptive summary of the selected features is listed in 4.5.

Table 4.5: Feature Details

WIDE	Cambridge	Description
PktNo	actual_data_pkts	total number of packets sent
Total Bytes	actual_data_bytes	total number of bytes sent
AvgSize	mean_data_IP	average size of IP payload
VarSize	var_data_IP	variance of IP payload size
AvgDiffSize		average of absolute difference in IP payload size
MaxSize	max_data_IP	maximum size of IP payload
MinSize	min_data_IP	minimum size of IP payload
SizeHomo	*extracted*	the ratio of maximum over minimum packet size
AvgIntv	mean_IAT	average time difference of two consecutive packets
VarIntv	var_IAT	variance of time difference of two consecutive packets
MaxIntv	max_IAT	maximum of time difference of two consecutive packets
MinIntv	min_IAT	minimum of time difference of two consecutive packets
IntvHomo	*extracted*	ratio of maximum over minimum interval time[2]

1. Each feature listed is computed separately for both the sending and receiving host.
2. Over consecutive packets.

4.3 Clustering Method

All flow clustering was performed using agglomerative Hierarchical Clustering with average linkage. Prior to computation, all data was scaled to the interval $[0, 1]$. The merge criterion to join two clusters was selected as the two closest clusters $c_i, c_j \in C$ in Euclidean distance over a dimensionality of F features. The value of feature f in cluster i is denoted c_{if}.

$$\sum_{f=0}^{F}(c_{if} - c_{jf})^2 = \min_{\substack{c_\alpha \in C \\ c_\beta \in C}} \sum_{f=0}^{F}(c_{\alpha f} - c_{\beta f})^2 \tag{4.1}$$

The center of the newly merged cluster was computed as a weighted mean of the centers of the clusters being merged. The computation for the merge is shown in equation 4.2, with n_i representing the number of flows of cluster c_i.

$$c_{kf} = \sum_{f=0}^{F} \Big(n_i c_{if} + n_j c_{jf} \Big) \tag{4.2}$$

Clusters c_i and c_J were removed from C, while c_k was placed into C before the next iteration of the algorithm. When the number of clusters N was within to $20 \leq N \leq 80$ a complete clustering scheme was produced and recorded.

The HC algorithm also included the functionality of calculating the covariances between the features. The covariance between features f_i and f_j was calculated in the standard way,

$$\mathrm{cov}(f_i, f_j) = \frac{1}{n} \sum_{k=1}^{N} (c_{ki} - \mu_i)(c_{kj} - \mu_j) \tag{4.3}$$

4.4 Means of Analysis

The classifications of Tables 4.1 and 4.2 define an inherent measure of the purity of a cluster. Purity was defined as the percentage of flows within the cluster of the most represented class. The classification, $\mathcal{C}_i$, of cluster i was determined by the mode of all classifications of the flows x within the cluster, The purity measure $\mathcal{P}_i$ of cluster c_i is calculated as the number of flows within the cluster of the same class divided by the total number of flows within the cluster.

$$\mathcal{P}_i = \frac{|\{x : x \in c_i \cap \mathcal{C}_i\}|}{|\{x : x \in c_i\}|} \tag{4.4}$$

A weighted average of this measure over all N clusters and n flows produces a purity measure for the entire clustering scheme.

$$\mathcal{P} = \frac{1}{n} \sum_{i=1}^{N} n_i \mathcal{P}_i \tag{4.5}$$

4.5 Supervised Feature Selection Using Cluster Purity

An iterative feature selection wrapper method was developed from the action of the cluster scheme purity measure (4.5) on the Cambridge data. On each iteration of the method, HC was performed a number of times equal to the number of features in some feature set $\mathcal{F}$. For each HC trial, one feature was removed. The feature whose removal led to the highest error rate was selected as a discriminating feature. The method was repeated until no features remained in $\mathcal{F}$. The algorithm is stated as follows:

1. Determine a fixed number of clusters N.
2. Determine a feature set $\mathcal{F} = \{f_i, i = 1, ..., F\}$
3. Perform F trials of the HC algorithm, each with one of the f_i features removed.
4. Record the purity $\mathcal{P}_{f_i}$ of each resultant clustering scheme.
5. Find f_i' such that $\mathcal{P}_{f_i} = \max_{f \in \mathcal{F}}\{\mathcal{P}_f\}$.

   This feature denotes the feature whose removal led to the highest error scheme.
6. Place f_i' in a set of discriminating features $\mathcal{F}'$.
7. Remove f_i' from $\mathcal{F}$ and take $F = F - 1$.
8. Repeat from Step 3 until $\mathcal{F}'$ contains the desired number of features or $F = 0$.

4.6 Iterative Variance Methods for Unsupervised Feature Selection

A variance measure was developed as a means of identifying poorly discriminating features without reliance on flow labels. The variance of each feature was measured locally within each cluster and compared to its global mean within the entire data set. The representative feature of each cluster was the feature that most deviated from the global mean. The least discriminating feature was taken to be the minimum of the representative feature description over all clusters.

The local mean and variance of each feature f within each cluster i was computed in the standard way. Define n_i as the number of flows within cluster i, and $x_{if}^{(k)}$ as the value of feature f of the kth flow within cluster i.

$$\mu_{if} = \frac{1}{n_i} \sum_{k=1}^{n_i} x_{if}^{(k)} \tag{4.6}$$

$$v_{if} = \frac{1}{n_i} \sum_{k=1}^{n_i} (x_{if}^{(k)} - \mu_{if})^2 \tag{4.7}$$

Because the difference of the global and local variance measures were very diverse over the different features, the local variances were scaled to the interval $[0, 1]$.

$$v'_{if} = \frac{v_{if} - \min\limits_{i=1,\ldots,N} \{v_{if}\}}{\max\limits_{i=1,\ldots,N} \{v_{if}\} - \min\limits_{i=1,\ldots,N} \{v_{if}\}} \tag{4.8}$$

The global mean Π_f was taken as as a weighted average of the normalized global variances. In the non-normalized case this is the global mean of the squares of the feature values.

$$\Pi_f = \frac{1}{n}\sum_{i=1}^{N} n_i v'_{if} \tag{4.9}$$

The feature removal criterion is described as removing the feature producing the minimum value over all features of the the feature set of the maximum of the local feature variances of each cluster, or equivalently:

$$\min_{f\in\mathcal{F}} \max_{i=1,\ldots,N} |\Pi_f - v'_{if}| \tag{4.10}$$

Note for a feature set consisting of F features, the unsupervised algorithm performs in $O(F)$ time complexity while the algorithm of Section 4.5 is of order $O(F^2)$. The algorithm is as follows.

1. Determine a fixed number of clusters N.
2. Determine a feature set $\mathcal{F} = \{f_i, i = 1, ..., F\}$
3. Perform one trial of the HC algorithm.
4. Compute local mean (4.6) and normalized local variance (4.8) of all features $f \in \mathcal{F}$.

5. Calculate the global weighted average Π_f of all features $f \in \mathcal{F}$.

6. Select the feature f_i meeting the condition of Equation 4.10.

7. Discard f_i from $\mathcal{F}$ and take $F = F - 1$.

8. Repeat from Step 3 until $\mathcal{F}$ contains the desired number of features or $F = 0$.

During the feature reduction the order of removal of each feature was recorded. The features removed near the end of the algorithm were listed as the most discriminating features.

4.7 kNN Feature Selection Using Feature Similarity

A k-nearest neighbor approach, or kNN, feature reduction method was developed involving a measure of feature similarity, the *Maximal Information Compression Index*, or λ_2. The reasoning for this measure was developed by (18). The λ_2 value is defined as the smallest eigenvalue of the covariance matrix between random variables. (18) describes the λ_2 as both the "minimum amount of information loss" or the "maximum amount of information compression" when reducing dimensionality and claims that its properties of symmetry, sensitivity to scaling, and invariance to rotation make it a convenient norm for feature selection. The measure is defined as follows:

$$2\lambda_2(x,y) = \text{var}(x) + \text{var}(y) - \sqrt{(\text{var}(x) + \text{var}(y))^2 - 4\text{var}(x)\text{var}(y)(1 - \rho(x,y)^2)} \quad (4.11)$$

The λ_2 measure is used as a criterion for feature removal within the context of a k-means clustering algorithm. The authors of (18) use the λ_2 value to induce an error threshold ϵ allowing for multiple feature removal, but due to the relatively small number of features provided with the data sets used in the context of this experiment, the threshold measure was not used. Define r_f^k as $\lambda_2(f, f_k)$, where f_k is the kth nearest neighbor of f in Euclidean distance. The modified kNN algorithm is then written:

1. Determine a feature set $\mathcal{F} = \{f_i, i = 1, ..., F\}$
2. Determine an initial value of $k \leq F - 1$.
3. For each $f \in \mathcal{F}$ compute r_f^k.
4. Find feature f' for which r_f^k is minimum.
5. Remove the kth nearest neighbor f_k of feature f' from $\mathcal{F}$ and take $F = F - 1$.
6. Repeat from Step 3 until $\mathcal{F}$ contains the desired number of features or $F = 0$.

Again, the removal order of the features was recorded, and the features removed near the end of the algorithm were classified as the most discriminating features.

4.8 Brute Force Validation of Feature Selection Methods

To gain a comparative measure and better perspective on the accuracy of the feature selection methods of Sections 4.5, 4.6, and 4.7, a brute force trial was performed on all 2^{14} combinations of the Cambridge features listed in Table 4.4. The accuracy of each of the resultant schemes was computed using the purity measure $\mathcal{P}$ (4.5).

The results of the brute force trials served a dual purpose. Firstly, the results of this section were able to provide information about what the best combinations of the top features selected by each method were. Secondly, the brute force results were able to provide a ranking of the schemes resultant from each of the feature selection methods and allow for a comparison between them.

Chapter 5

Feature Selection and Clustering Results

5.1 Number of Clusters

Proper selection of the number of clusters includes both attempting to equate the number of clusters to the number of classes as well as attempting to maximize the purity, or minimize the error of the resultant scheme. To accomplish this we can construct a function representative of both the error rate and an induced penalty for the number of clusters as follows:

$$f(x) = \alpha(1 - \mathcal{P}_x) + (1 - \alpha)n \tag{5.1}$$

Averaging a random sample of 50 clustering trials, and setting the parameter $\alpha = .9$, the observed behavior is shown in Figure 5.1. The average rating alone shows a leveling-off of the error rate after approximately 70 clusters. For the penalty function, we see the minimum value occurs near a level of 60 clusters.

A more thorough analysis was performed between the range of 20 and 80 clusters. The average results from all 2^{14} feature combinations were compared, and these are shown in Figure 5.2, with a detailed description in Table 5.1. Considering the level error range of the random sampling, the minimal range of the penalty function, and the average error amongst the complete set of trials, a level of $N = 80$ was chosen for the

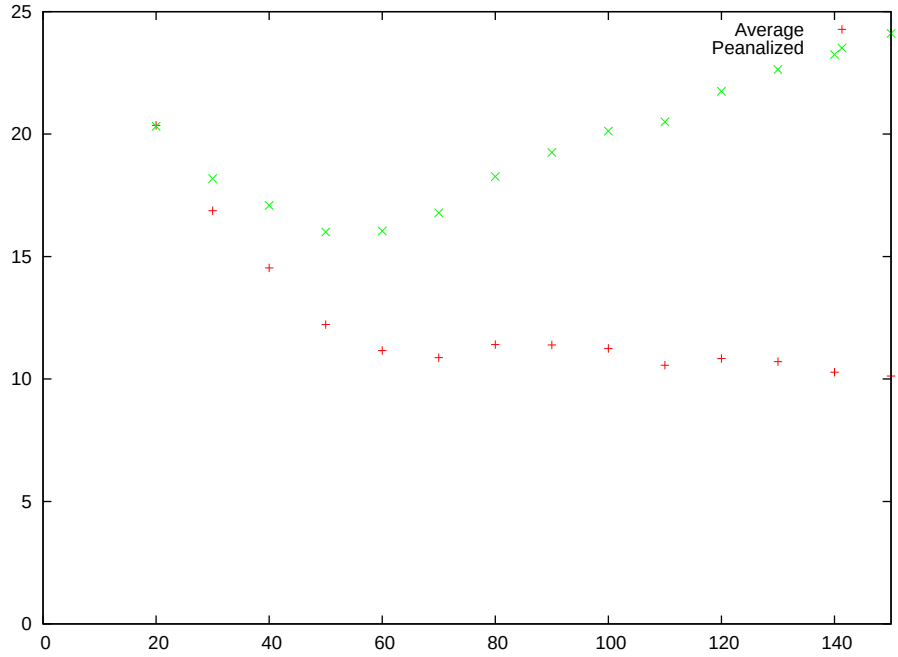


Fig. 5.1: Purity and Penalty v. Number of Clusters

experiments of Section 5.2, though the range $60 \leq N \leq 80$ would provide equivalent results.

5.2 Cambridge Results

5.2.1 Supervised Feature Selection Results

Supervised feature selection on the Cambridge feature set of Table 4.4 by means of the purity measure (4.5) selected size based features as the most discriminating. Recall the the SizeHomo feature is a ratio of the MaxSize to MinSize of the payload size of an IP Packet. Byte and packet counts were ranked as the next highest features. The four time-based features of the Cambridge data set were the last to be removed by the method. Table 5.2 lists the removal order of the features.

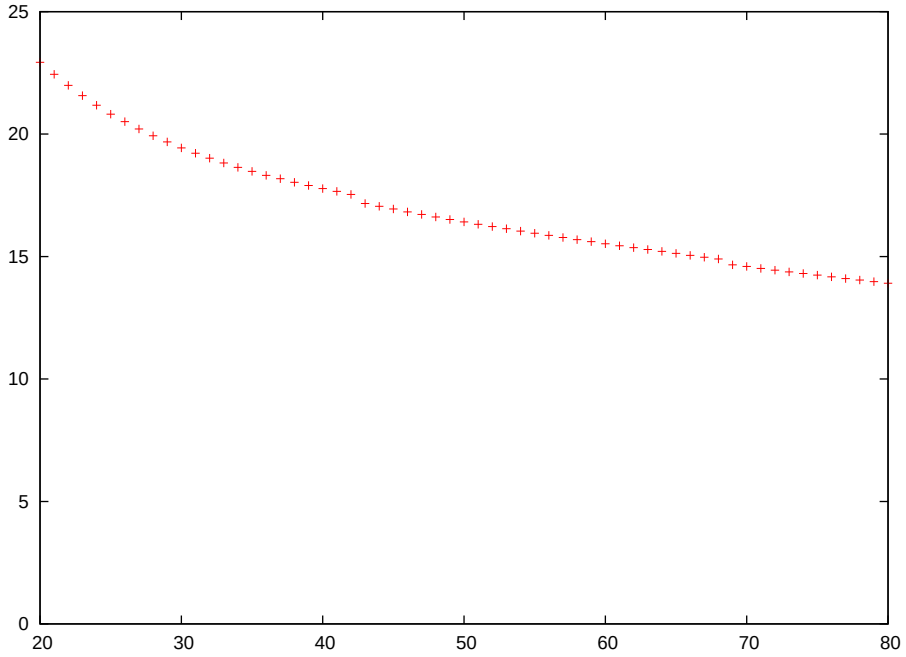


Fig. 5.2: Purity v. Number of Clusters Refined

5.2.2 Variance Method Selection Results

In contrast to selecting the most discriminating features first, the variance method eliminates the least discriminating features first. For this reason, the removal rankings of the tables in this section are in reverse order. The first course of variance method removal did not involve normalization of the features. The removal rankings of Table 5.3a are the results of this method. As the intra-feature variances of the _C and _S descriptors was much less than that of the inter-feature variances of the different feature varieties, the removal scheme was merely an indication of the features with the most diversity. For this reason, normalization was added.

The normalized variance method selection order is listed in Table 5.3b. Similarly to the supervised selection method, the SizeHomo features were picked as predominant

Table 5.1: Purity Statistics of Number of Clusters

n	Minimum	Maximum	Average
20	36.04%	58.65%	17.23%
30	41.33	63.75	18.50
40	44.87	68.79	18.63
50	48.04	71.43	21.33
60	51.04	75.39	21.84
70	53.85	78.88	21.94
80	56.21	80.14	22.12

discriminators. Size-based and time-based features remained in approximately the same ordering, with the somewhat more dramatic movements of the byte and packet counts, as well as the VarSize_C/S feature. These three features were given less discriminating ranks in the unsupervised selection results.

5.2.3 kNN Method Selection Results

The features selected by the kNN method using the λ_2 measure (4.11) are listed in Table 5.4. The features of total bytes, average size, and variant size were removed first. Homogeneous and average size features were given priority as highly discriminating features. For the kNN method, server-side features appeared to be more dominant than their client-side counterparts. The kNN method was the only feature selection method that listed time-based features in the top five discriminators.

5.2.4 Brute Force Top Feature Subsets

The brute force method discussed in Section 4.8 does not select individual features, rather it makes known the feature sets used in the most pure clustering schemes. Because

Table 5.2: Supervised Feature Reduction on Cambridge Feature Set

Order Selected	Feature Name
1	SizeHomo_C
2	SizeHomo_S
3	VarSize_S
4	Total Bytes_S
5	Total Bytes_C
6	AvgSize_C
7	VarSize_C/S
8	AvgSize_C/S
9	VarSize_C
10	AvgSize_S
11	AvgIntv_S
12	VarIntv_C
13	AvgIntv_C
14	VarIntv_S

all of the top five schemes used five or less features, the comparisons in Table 5.5 use only the top five features of the respective method.

Each of the methods was able to correctly identify a superset of one of the four most accurate (according to the purity measure) clustering schemes. The supervised selection identified the three most accurate schemes. Both homogeneous size features were identified in the top five features of all schemes. The unsupervised method was the only method that selected the AvgSize feature, correctly identifying the fourth most accurate scheme.

It is also noted that the third and fourth most accurate schemes are both identified by the variance reduction method while preserving the order of feature selection. The SizeHomo features were the top two discriminators selected by the variance method, and the AvgSize feature was the third.

Table 5.3: Variance Method Reduction on Cambridge Feature Set

Order Removed	Feature Name
14	AvgIntv_C
13	AvgIntv_S
12	SizeHomo_C
11	AvgSize_C/S
10	SizeHomo_S
9	AvgSize_C
8	AvgSize_S
7	VarIntv_C
6	VarIntv_S
5	VarSize_C
4	VarSize_C/S
3	VarSize_S
2	Total Bytes_C
1	Total Bytes_S

(a) Unnormalized Variances

Order Removed	Feature Name
14	SizeHomo_C
13	SizeHomo_S
12	AvgSize_C/S
11	VarSize_C
10	VarSize_S
9	AvgSize_S
8	Total Bytes_S
7	AvgSize_C
6	AvgIntv_S
5	VarIntv_C
4	VarIntv_S
3	AvgIntv_C
2	Total Bytes_C
1	VarSize_C/S

(b) Normalized Variances

5.2.5 Clustering Outcomes

The top four clustering results are detailed in this section. Part (a) of each figure shows a scaled view of the largest cluster of each of the Cambridge classes, 12 in total. Part (b) details the error of the clustering scheme, 1-$\mathcal{P}$, the number of clusters and singleton clusters, and the specific features used in each scheme.

Part (c) shows a breakdown by class of the clusters within the resultant scheme. The middle four ranking columns lists the number of clusters that were decided to belong to their respective class, and the weighted purity measure over all class clusters. The latter of these show what percent of class flows was located within the number of clusters deemed to be of that class. The final four columns detail the largest cluster of each class, and its representative amount of class flows both within the cluster and from the entire class itself. In all four figures, the largest cluster of each class was also the cluster

Table 5.4: kNN Feature Reduction on Cambridge Feature Set

Order Removed	Feature Name
14	VarIntv_C
13	SizeHomo_S
12	AvgSize_S
11	SizeHomo_S
10	AvgIntv_C
9	AvgIntv_S
8	VarIntv_S
7	VarSize_C/S
6	VarSize_C
5	VarSize_S
4	AvgSize_C/S
3	AvgSize_C
2	Total Bytes_S
1	Total Bytes_C

containing the most flows of each class, regardless of the class of the cluster itself with the exception of the GAMES class. In all cases, the cluster containing the most GAMES flows had no more than 2 flows. Recall that of the entire data set, only 8 flows are of the GAMES class.

The most accurate scheme is displayed, using both SizeHomo features along with the total packet count. This outcome is detailed in 5.3. The largest clusters of each class had high purity ratings, and contained on average over half of the class flows. Database traffic seemed to have the poorest classification, averaging about a 50% purity over 6 clusters.

The second scheme of Table 5.4, utilizing byte count instead of packet count had slightly less purity about the class clusters. The number of clusters of each class had some degree of variation, while the largest cluster of each class maintained their proportions to each other.

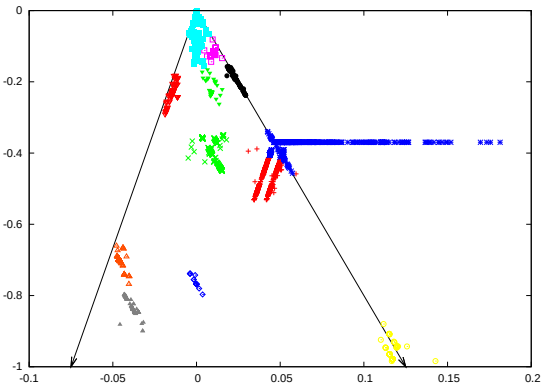
Table 5.5: Features of Top Overall Clustering Schemes

Rank	Used Features	Subset of Top Five Features		
		Supervised	Unsupervised	kNN
1	SizeHomo_C SizeHomo_S Total Bytes_S	•		
2	SizeHomo_C SizeHomo_S Total Bytes_C	•		
3	SizeHomo_C SizeHomo_S	•	•	•
4	SizeHomo_C SizeHomo_S AvgSize_C/S		•	
5	SizeHomo_C SizeHomo_S VarSize_C/S VarIntv_S Total Bytes_S			

Only using the two homogeneous size features, the clustering outcome is detailed in Table 5.5, P2P traffic is the only class to realize an increase across the class purity, and the same amount of class clusters are maintained. The WWW traffic's largest cluster now only contains approximately half of the flows.

Adding the AvgSize feature to the SizeHomo features resulted in the outcome of Table 5.6. This scheme boasted the highest purity of the DBASE traffic. SERVICES traffic was placed into a larger cluster, reducing its purity. The general trend of lowering purity rates over all clusters continued.

Fig. 5.3: First Most Accurate Cambridge Scheme



(a) Largest Cluster of Each Class

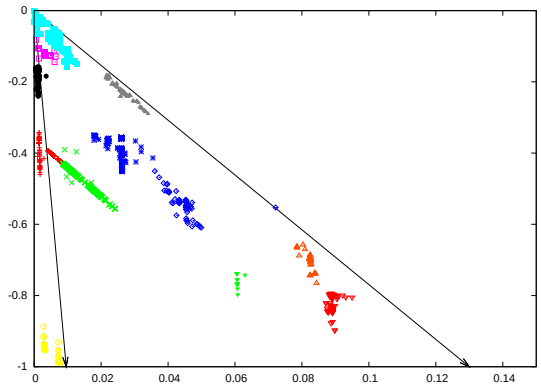
Weighted Error:	19.86%
Clusters:	80
Singletons:	20
Features:	SizeHomo_C
	SizeHomo_S
	Total Bytes_S

(b) Cluster Scheme Details

	Rankings By Class				Largest Cluster of Class			
Class	Clusters	Purity	Flows	% Class	Total	Class	Purity	% Class
WWW	7	93.2	926	92.6	685	654	95.5	65.4
MAIL	3	94.8	868	86.8	823	819	99.5	81.9
FTP-CONTROL	3	73.0	335	67.0	231	217	93.9	43.4
FTP-PASSV	5	67.1	367	73.4	178	165	92.7	33.0
ATTACK	13	80.5	412	82.4	344	321	93.3	64.2
P2P	11	94.9	300	60.0	169	168	99.4	33.6
DBASE	6	50.7	341	68.2	564	242	42.9	48.4
FTP-DATA	6	72.8	500	100.0	659	472	71.6	94.4
MULTIMEDIA	10	93.3	322	66.8	284	273	96.1	56.6
SERVICES	1	78.6	449	89.8	571	449	78.6	89.8
INTERACTIVE	14	84.5	71	62.3	20	20	100.0	17.5
GAMES	1	100.0	1	12.5	1	1	100.0	25.0

(c) Clustering Scheme Breakdown by Class

Fig. 5.4: Second Most Accurate Cambridge Scheme



(a) Largest Cluster of Each Class

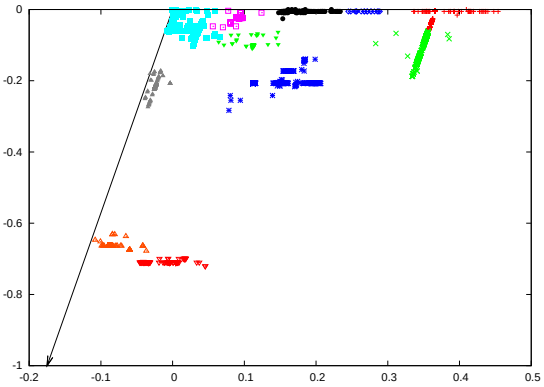
Weighted Error:	20.90%
Clusters:	80
Singletons:	18
Features:	SizeHomo_C
	SizeHomo_S
	Total Bytes_C

(b) Cluster Scheme Details

	Rankings By Class				Largest Cluster of Class			
Class	Clusters	Purity	Flows	% Class	Total	Class	Purity	% Class
WWW	9	91.3	964	96.4	685	654	95.5	65.4
MAIL	2	98.7	764	76.4	763	757	99.2	75.7
FTP-CONTROL	3	74.3	335	67.0	231	217	93.9	43.4
FTP-PASSV	8	67.5	374	74.8	178	165	92.7	33.0
ATTACK	13	79.5	412	82.4	353	321	90.9	64.2
P2P	13	92.4	306	61.2	168	167	99.4	33.4
DBASE	7	50.9	343	68.6	564	242	42.9	48.4
FTP-DATA	1	65.2	500	100.0	767	500	65.2	100.0
MULTIMEDIA	6	93.2	313	64.9	284	273	96.1	56.6
SERVICES	1	78.6	449	89.8	571	449	78.6	89.8
INTERACTIVE	16	94.4	67	58.8	20	20	100.0	17.5
GAMES	1	100.0	1	12.5	1	1	100.0	25.0

(c) Clustering Scheme Breakdown by Class

Fig. 5.5: Third Most Accurate Cambridge Scheme



(a) Largest Cluster of Each Class

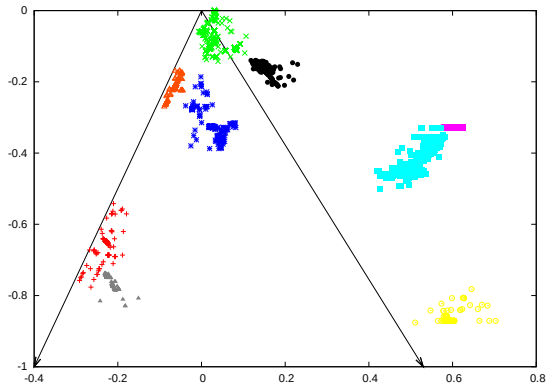
Weighted Error:	21.02%
Clusters:	80
Singletons:	19
Features:	SizeHomo_C
	SizeHomo_S

(b) Cluster Scheme Details

	Rankings By Class				Largest Cluster of Class			
Class	Clusters	Purity	Flows	% Class	Total	Class	Purity	% Class
WWW	11	91.3	965	96.5	472	465	98.5	46.5
MAIL	2	98.7	764	76.4	763	757	99.2	75.7
FTP-CONTROL	3	73.8	335	67.0	231	217	93.9	43.4
FTP-PASSV	6	67.2	370	74.0	178	165	92.7	33.0
ATTACK	14	78.8	412	82.4	353	321	90.9	64.2
P2P	13	95.0	301	60.2	168	167	99.4	33.4
DBASE	6	50.7	341	68.2	564	242	42.9	48.4
FTP-DATA	1	65.2	500	100.0	767	500	65.2	100.0
MULTIMEDIA	7	93.2	313	64.9	284	273	96.1	56.6
SERVICES	1	78.6	449	89.8	571	449	78.6	89.8
INTERACTIVE	15	86.4	70	61.4	20	20	100.0	17.5
GAMES	1	100.0	1	12.5	1	1	100.0	25.0

(c) Clustering Scheme Breakdown by Class

Fig. 5.6: Fourth Most Accurate Cambridge Scheme



(a) Largest Cluster of Each Class

Weighted Error:	22.58%
Clusters:	80
Singletons:	21
Features:	SizeHomo_C
	SizeHomo_S
	AvgSize_C/S

(b) Cluster Scheme Details

	Rankings By Class				Largest Cluster of Class			
Class	Clusters	Purity	Flows	% Class	Total	Class	Purity	% Class
WWW	6	88.1	925	92.5	592	507	85.6	50.7
MAIL	7	88.3	979	97.9	394	385	97.7	38.5
FTP-CONTROL	3	63.8	342	68.4	270	224	83.0	44.8
FTP-PASSV	3	69.0	351	70.2	178	165	92.7	33.0
ATTACK	10	83.9	386	77.2	337	321	95.3	64.2
P2P	13	95.1	251	50.2	169	168	99.4	33.6
DBASE	6	91.8	101	20.2	1135	359	31.6	71.8
FTP-DATA	1	100.0	491	98.2	491	491	100.0	98.2
MULTIMEDIA	13	94.8	328	68.0	284	273	96.1	56.6
SERVICES	1	44.1	500	100.0	1135	500	44.1	100.0
INTERACTIVE	16	76.3	71	62.3	20	20	100.0	17.5
GAMES	1	100.0	1	12.5	14	3	21.4	37.5

(c) Clustering Scheme Breakdown by Class

5.3 WIDE Results

5.3.1 Variance Method Selection Results

The unsupervised variance selection method was then performed on the features of Table 4.3 of the WIDE feature set. A trial with the 18 features led to a diverse result listed in Table 5.6a. Size-based features had predominance as discriminators, but time-based features were more predominant than in the other algorithms. The algorithm gave preference to average-valued features as the top discriminators. With the exception of AvgIntv, the time-based features were again marked as poorly discriminating.

Table 5.6b lists the features selected by the method on a reduced feature set, without the total byte and packet counts. This trial displayed the most preference to client side time-based features. Size features, predominantly SizeHomo, were ranked next most discriminating. The remaining time-based features were amongst the first removed.

The top three features of the supervised reduction on the Cambridge trace were also listed as discriminators on the set with the aggregate counts removed. Timing features were marked more discriminating as compared to the supervised Cambridge results. The AvgSize, AvgIntv, and SizeHomo features were listed as discriminators both in this set and in the unnormalized Cambridge result. There was also a high degree of matching with the kNN removal on the Cambridge data.

A final variance-based feature selection was performed on a larger feature set of the WIDE trace. This set involved all features provided with the trace with ordering and redundant or reciprocal features removed. The full feature set is listed in Appendix A.

Table 5.6: Variance Method Reduction on WIDE Feature Set

Order Removed	Feature Name
18	VarSize_C
17	AvgIntv_C
16	AvgDiffSize_S
15	AvgSize_S
14	AvgSize_C/S
13	SizeHomo_S
12	AvgSize_C
11	AvgIntv_S
10	VarSize_S
9	AvgDiffSize_C
8	IntvHomo_C
7	Total Bytes
6	SizeHomo_C
5	VarSize_C/S
4	Total Packets
3	VarIntv_S
2	IntvHomo_S
1	VarIntv_C

(a) 18 Features Set

Order Removed	Feature Name
16	VarIntv_C
15	AvgIntv_C
14	VarSize_S
13	SizeHomo_C
12	SizeHomo_S
11	AvgDiffSize_C
10	VarSize_C
9	AvgDiffSize_S
8	VarIntv_S
7	AvgIntv_S
6	AvgSize_S
5	VarSize_C/S
4	IntvHomo_S
3	AvgSize_C/S
2	IntvHomo_C
1	AvgSize_C

(b) Byte and Packet Counts Removed

Duration, not represented in any other feature set, was selected as both the most and least discriminating feature. Average and variant size features ranked highly as discriminators. The bulk of the time-based features were placed predominantly in the middle third and above. SizeHomo, along with its components of MaxSize and MinSize were removed near the beginning of the algorithm, marking them as poorly discriminating. The results are shown in Table 5.7.

5.3.2 Variance Method Clustering Outcomes

The display shown for the clustering outcomes of the WIDE trace is similar to that of the Cambridge trace, except for the third section. This section now addresses the cluster containing the most flows of each class, rather than the largest cluster of each

class (in the Cambridge trace these measures were equivalent). Due to the port-based classification listed in Table 4.2, these purity measures should be taken somewhat lightly. The purest clusters of each scheme were around 80% WWW traffic.

Figures B.1, B.2, and B.3 of Appendix B detail the clustering outcomes over top four features selected by the variance method. The 3- and 4- feature schemes both have the black-hole effect, but in the 4-feature scheme there are no more than 54% of each class representative in the large cluster. Reducing to two features breaks up the large cluster, but as a consequence reduces the purity amongst the class clusters.

Over the reduced feature set, the top discriminators of Table 5.6b are detailed in Figures C.1, C.2, and C.3 of Appendix C. Like the results of the non-reduced, the majority of all classes were gathered into WWW clusters. The large cluster showed the highest rate of increase for WWW traffic.

The final three Figures D.1, D.2, and D.3 of Appendix D. detail the clusters obtained from the variance method feature reduction on the full feature set. These results have the first occurrence of the large clusters being of a class other than WWW. For the large cluster of the 4-feature scheme the traffic is nearly 80% WWW, increasing as the amount of features reduce. In the 2-feature scheme, MAIL and DNS no longer have any clusters representing them.

5.3.3 kNN Method Selection Results

The kNN method on the WIDE trace had similar results to the Cambridge trace. The features of total bytes, packets, average size, and variant size were removed first and in nearly the same order on both data sets. Homogeneous and average size features were

Table 5.7: Variance Method Reduction on Full Wide Feature Set

Order Removed	Feature Name
42	Duration_S
41	VarSize_C/S
40	VarSize_S
39	AvgSize_S/C
38	AvgSize_S
37	PktNo_S
36	MaxSize_S
35	VarSize_S/C
34	IntvHomo_C/S
33	IntvHomo_S/C
32	IntvHomo_C
31	IntvHomo_S
30	MinIntv_S
29	MinIntv_C
28	MaxIntv_S
27	VarIntv_C/S
26	MaxIntv_C
25	VarIntv_S/C
24	VarIntv_C
23	VarIntv_S
22	AvgIntv_S/C
21	AvgIntv_C/S
20	AvgIntv_C
19	AvgIntv_S
18	MinDiffSize_S
17	MinDiffSize_C
16	MaxDiffSize_C
15	MaxDiffSize_S
14	AvgDiffSize_S
13	AvgSize_C/S
12	AvgSize_C
11	AvgDiffSize_C
10	SizeHomo_C/S
9	SizeHomo_C
8	VarSize_C
7	SizeHomo_S/C
6	PktNo_C
5	MinSize_C
4	SizeHomo_S
3	MinSize_S
2	MaxSize_C
1	Duration_C

given priority as highly discriminating features. Server-side features again appeared to be more dominant than their client-side counterparts.

Table 5.8: *kNN* Feature Reduction on WIDE Feature Set

Order Removed	Feature Name
18	IntvHomo_S
17	SizeHomo_S
16	AvgSize_S
15	AvgIntv_S
14	VarSize_C
13	AvgDiffSize_S
12	VarIntv_S
11	IntvHomo_C
10	VarIntv_C
9	SizeHomo_C
8	AvgDiffSize_C
7	AvgIntv_C
6	VarSize_C/S
5	VarSize_S
4	AvgSize_C
3	AvgSize_C/S
2	Total Packets
1	Total Bytes

5.3.4 *kNN* Method Clustering Outcomes

The resultant outcomes of the *kNN* feature selection are listed in Figures E.1, E.2, and E.3 of Appendix E. In these schemes, feature reduction breaks the large cluster increasing the ratings for lower represented traffic like 7700 and DNS. The large assignments OTHER and 7700 classifications are no longer located in a WWW cluster, though these assignments contain no more than 16 percent of their respective class flows.

5.4 Feature Selection Summary

Table 5.9 provides a summary of the top 5 selected features of each method, denoted by •. It shows the first five features eliminated by each method, denoted by -. Homogeneous size emerges as a discriminated feature selected amongst several methods. AvgIntv is the next highest feature selected. With the exception of AvgSize_C and the DiffSize features, all other size features are selected in 3 of the 7 trials. Timing features, in particular the AvgIntv features, become popular on the on the kNN method and in WIDE trace.

Table 5.9: Feature Reduction Summary

	Cambridge				WIDE		
	Supervised	Variance (unnormalized)	Variance (normalized)	*kNN*	Variance	Variance Reduced	*kNN*
SizeHomo_C	•	•	•	•		•	
SizeHomo_S	•	•	•	•		•	•
AvgSize_C				-		-	-
AvgSize_S	-			•	•		•
AvgSize_C/S		•	•	-	•	-	-
VarSize_C		-	•		•		•
VarSize_S	•	-	•	-		•	-
VarSize_C/S		-	-		-	-	
AvgDiffSize_C							
AvgDiffSize_S					•		
IntvHomo_C					-	-	
IntvHomo_S						-	•
AvgIntv_C	-	•	-	•	•	•	
AvgIntv_S	-	•					•
VarIntv_C	-		-	•	-	•	
VarIntv_S	-		-		-		
Total Bytes	•	-	-	-			-

Chapter 6

Conclusion

6.1 Number of Clusters

An appropriate number of clusters for the data sets used in these experiments was determined through experiment to be 80. Reducing the number of clusters nearer to the number of classes led to a significant increase in the error of the overall clustering scheme. At a level of 80 clusters, it was shown a purity of nearly 80% can be attained. With an average of 20 singleton clusters, 60 clusters are left to consider meaningful for the 12 classifications of the Cambridge trace. Leaving this 20 cluster window to allow for distant flows, represented as singleton clusters, seems necessary for HC to allow for a better accuracy of the scheme as a whole. Over the remaining 60 clusters, the largest of each class maintained a very high purity rate, while the purity rates amongst all clusters of the same class were only slightly lower, Assigning between approximately 3 and 15 clusters per class maintained a high accuracy over the class and was likely necessary to account for the dissimilarity within the class itself.

6.2 Feature Reduction

Over the set of all experiments performed in Section 4.8, those schemes using only a small number of features were determined to provide a better overall accuracy than those using large amounts of features. These results hold for the Cambridge trace, where

the labeling on the data was issued through an extensive and well defined process and used to analyze the accuracy of the outcomes. Though a naive and likely incorrect port based purity measure was used on the WIDE trace, improvement was still shown when using a reduced number of features.

A reduced set of features does not only provide improved accuracy, but is also directly proportional to computational complexity: the lesser the number of features used in clustering the more efficiently an algorithm can perform. In the HC scheme, the computations of distance and averaging, as well as the size of the proximity matrix itself are directly proportional to the number of features used. A reduced number of features directly implies a reduced number of computations.

Though only an offline analysis using stored data was performed in this analysis, it would be advantageous to extend these findings to data captured in real-time. A reduced number of features would have a great impact on reducing the overhead necessary to perform the gathering and recording of the flow statistics. However, as the highly discriminating features were shown to be size based features involving the maximum and minimum sized packets over the total flow, a further investigation is needed to determine how online clustering is effected by the changing size features since the absolute maximum and minimum packet size values are not available until after a flow is completed.

6.3 Highly Discriminating Features

For the purposes of HC, the ratio of maximum to minimum IP payload size over a flow was shown to be a highly discriminating feature. Five of the seven methods tested selected the client version of this feature, while six of seven selected the server side

measure. Unsupervised feature removal by variance measure on the unlabeled WIDE trace was the only method that did not select one of the SizeHomo features as the top 5 discriminators, however the server variety of the feature was ranked in the sixth most discriminating position.

The significance of this measure is that it was not only selected as a discriminating feature over different selection methods, it was selected over significantly different data sets. The backbone WIDE trace and the enterprise Cambridge trace have very different users generating very different traffic profiles. The average interval feature was also highly rated across both data sets. Selecting a feature set universal features such as these gives an excellent opportunity to improve efficiency of feature selection algorithms.

Remaining size features were selected in at least three of the five methods across the data sets as discriminating features. In a general sense, the size based features were selected far more often than the time based features. The ratio of maximum and minimum size was the predominant size feature, and the average interval was the predominant time based feature.

6.4 Poorly Discriminating Features

Though each of the methods had an order of feature removal, it is not entirely appropriate to consider those features first removed (or near the end, depending on method), as non-discriminating or noisy features. Total Byte counts, while being an important feature in the top performers of the Cambridge trace were quickly eliminated in the other methods. This example, in particular, is due to the variance measure: the

byte counts must have had a maximal variance locally within cluster with a large amount of separation from the global mean of variances.

Several features, including AvgSize_C, VarSize_C/S, and VarIntv_S scored poorly across several methods on both data sets. Features such as these provide another opportunity to improve feature selection by initially removing these features from the feature set before performing the selection algorithm.

Appendix A

WIDE Feature Set Details

Table A.1: Full WIDE Feature Set

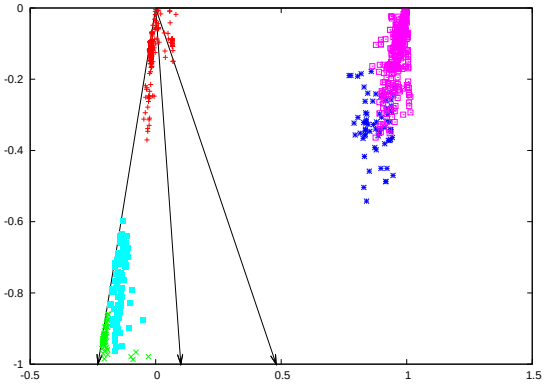
	Feature Name	Feature Details
•	PktNo_C	total number of packets sent by client
•	PktNo_S	total number of packets sent by server
•	Duration_C	duration from client to server
•	Duration_S	duration from serer to client
•	AvgSize_C	average size of IP payload sent by client
•	AvgSize_S	average size of IP payload sent by server
•	AvgSize_C/S	the ratio of AvgSize_C over AvgSize_S
•	AvgSize_S/C	the ratio of AvgSize_S over AvgSize_C
•	VarSize_C	variance of size of IP payload sent by client
•	VarSize_S	variance of size of IP payload sent by server
•	VarSize_C/S	the ratio of VarSize_C over VarSize_S
•	VarSize_S/C	the ratio of VarSize_S over VarSize_C
•	MaxSize_C	maximum size of IP payload sent by client
•	MaxSize_S	maximum size of IP payload sent by server
•	MinSize_C	minimum size of IP payload sent by client
•	MinSize_S	minimum size of IP payload sent by server
	Q1Size_C	first quartile of IP payload size sent by client
	Q1Size_S	first quartile of IP payload size sent by server
	MedSize_C	median size of IP payload sent by client
	MedSize_S	median size of IP payload sent by server
	Q3Size_C	third quartile of IP payload size sent by client
	Q3Size_S	third quartile of IP payload size sent by server
•	SizeHomo_C	ratio of MaxSize_C over MinSize_C
•	SizeHomo_S	ratio of MaxSize_S over MinSize_S
•	SizeHomo_C/S	ratio of SizeHomo_C over SizeHomo_C
•	SizeHomo_S/C	ratio of SizeHomo_S over SizeHomo_S
•	AvgDiffSize_C	average of absolute difference in IP payload size of two consecutive packets sent by client
•	AvgDiffSize_S	average of absolute difference in IP payload size of two consecutive packets sent by server
•	MaxDiffSize_C	maximum of absolute difference in IP payload size of two consecutive packets sent by client
•	MaxDiffSize_S	maximum of absolute difference in IP payload size of two consecutive packets sent by server
	MedDiffSize_C	median of absolute difference in IP payload size of two consecutive packets sent by client
	MedDiffSize_S	median of absolute difference in IP payload size of two consecutive packets sent by server
•	MinDiffSize_C	minimum of absolute difference in IP payload size of two consecutive packets sent by client
•	MinDiffSize_S	minimum of absolute difference in IP payload size of two consecutive packets sent by server
•	AvgIntv_C	average time difference of two consecutive packets sent by client
•	AvgIntv_S	average time difference of two consecutive packets Sent by server
•	AvgIntv_C/S	the ratio of AvgIntv_C over AvgIntv_S
•	AvgIntv_S/C	the ratio of AvgIntv_S over AvgIntv_C
•	VarIntv_C	variance of time difference of two consecutive packets sent by client
•	VarIntv_S	variance of time difference of two consecutive packets Sent by server
•	VarIntv_C/S	the ratio of VarIntv_C over VarIntv_S
•	VarIntv_S/C	the ratio of VarIntv_S over VarIntv_C
•	MaxIntv_C	maximum time difference of two consecutive packets sent by client
•	MaxIntv_S	maximum time difference of two consecutive packets sent by server
•	MinIntv_C	minimum time difference of two consecutive packets sent by client
•	MinIntv_S	minimum time difference of two consecutive packets sent by server
	Q1Intv_C	first quartile of time difference of two consecutive packets sent by client
	Q1Intv_S	first quartile of time difference of two consecutive packets sent by server
	MedIntv_C	median time difference of two consecutive packets sent by client
	MedIntv_S	median time difference of two consecutive packets sent by server
	Q3Intv_C	third quartile of time difference of two consecutive packets sent by client
	Q3Intv_S	third quartile of time difference of two consecutive packets sent by server
•	IntvHomo_C	the ratio of MaxIntv_C over MinIntv_C
•	IntvHomo_S	the ratio of MaxIntv_S over MinIntv_S
•	IntvHomo_C/S	the ratio of IntvHomo_C over IntvHomo_S
•	IntvHomo_S/C	the ratio of IntvHomo_S over IntvHomo_C
	AvgDiffIntv_C	average of order-2 difference of three consecutive packets sent by client
	AvgDiffIntv_S	average of order-2 difference of three consecutive packets sent by server
	MaxDiffIntv_C	maximum of order-2 difference of three consecutive packets sent by client
	MaxDiffIntv_S	maximum of order-2 difference of three consecutive packets sent by server
	MinDiffIntv_C	minimum of order-2 difference of three consecutive packets sent by client
	MinDiffIntv_S	minimum of order-2 difference of three consecutive packets sent by server
	AvgRate_C	average rate of packets sent by client
	AvgRate_S	average rate of packets sent by server
	AvgRate_C/S	the ratio of AvgRate_C over AvgRate_S
	AvgRate_S/C	the ratio of AvgRate_S over AvgRate_C

Unbulleted features were not used.

Appendix B

WIDE Clusters with Variance Selected Features

Fig. B.1: Clustering Outcome Using Top 4 Features of Wide Feature Set



(a) Five Largest Clusters

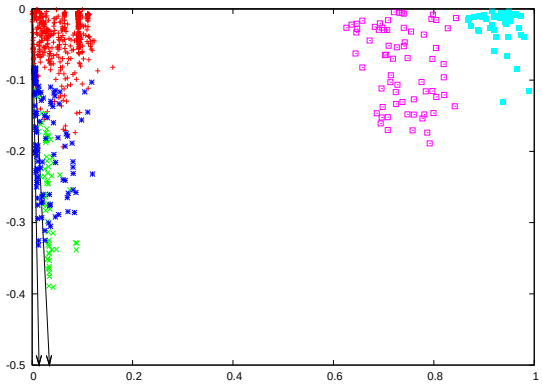
Clusters:	80
Singletons:	27
Features:	VarSize_C
	AvgIntv_C
	AvgSize_S
	AvgDiffSize_S

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	30	45.2	774	81	10.5	22.1	WWW
WWW	37	70.3	774	593	76.6	53.7	WWW
MAIL	0	-	774	47	6.1	51.6	WWW
7700	11	58.2	774	40	5.2	19.3	WWW
DNS	2	75.0	774	13	1.7	17.3	WWW

(c) Clustering Scheme Breakdown by Class

Fig. B.2: Clustering Outcome Using Top 3 Features of Wide Feature Set



(a) Five Largest Clusters

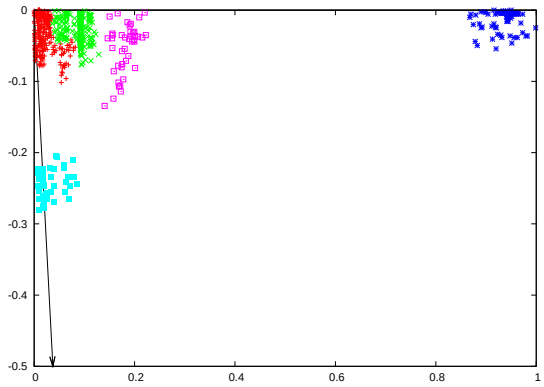
Clusters:	80
Singletons:	22
Features:	VarSize_C
	AvgIntv_C
	AvgDiffSize_S

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	36	45.1	961	98	10.2	26.1	WWW
WWW	28	71.7	961	742	77.2	67.1	WWW
MAIL	0	-	961	65	6.8	71.4	WWW
7700	11	54.0	961	43	4.5	20.8	WWW
DNS	5	85.7	961	13	1.4	17.3	WWW

(c) Clustering Scheme Breakdown by Class

Fig. B.3: Clustering Outcome Using Top 2 Features of Wide Feature Set



(a) Five Largest Clusters

Clusters:	80
Singletons:	12
Features:	VarSize_C
	AvgIntv_C

(b) Cluster Scheme Details

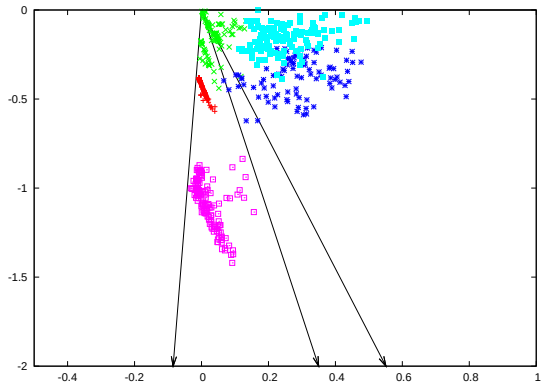
	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	36	50.0	264	62	23.5	16.9	WWW
WWW	25	69.1	759	611	80.5	55.3	WWW
MAIL	0	-	759	52	6.9	57.1	WWW
7700	15	56.7	759	28	3.7	13.5	WWW
DNS	4	71.4	91	9	9.9	12.0	WWW

(c) Clustering Scheme Breakdown by Class

Appendix C

WIDE Clusters with Variance Selected Features (Reduced)

Fig. C.1: Clustering Outcome Using Top 4 Features of Reduced Wide Feature Set



(a) Five Largest Clusters

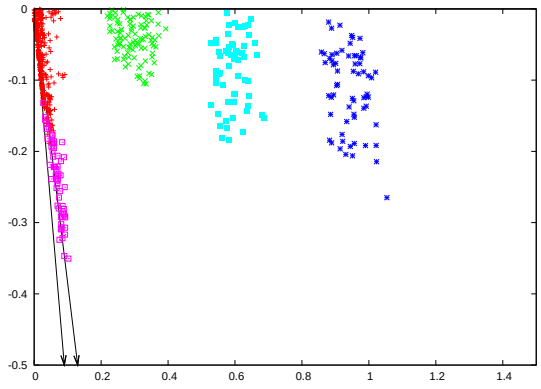
Clusters:	80
Singletons:	28
Features:	VarIntv_C
	AvgIntv_C
	SizeHomo_C
	VarSize_s

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	28	53.0	672	51	7.6	13.9	WWW
WWW	38	68.9	672	556	82.7	50.3	WWW
MAIL	0	nan	672	43	6.4	47.2	WWW
7700	9	56.2	169	30	17.8	14.5	WWW
DNS	5	87.5	108	10	9.3	13.3	WWW

(c) Clustering Scheme Breakdown by Class

Fig. C.2: Clustering Outcome Using Top 3 Features of Reduced Wide Feature Set



(a) Five Largest Clusters

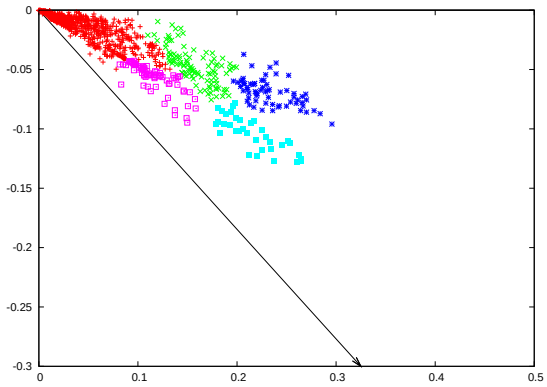
Clusters:	80
Singletons:	24
Features:	VarIntv_C
	AvgIntv_C
	VarSize_S

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	33	46.4	907	116	12.8	31.6	WWW
WWW	33	68.5	907	676	74.5	61.1	WWW
MAIL	0	-	907	50	5.5	54.9	WWW
7700	10	47.0	907	47	5.2	22.7	WWW
DNS	4	83.3	907	18	2.0	42	WWW

(c) Clustering Scheme Breakdown by Class

Fig. C.3: Clustering Outcome Using Top 2 Features of Reduced Wide Feature Set



(a) Five Largest Clusters

Clusters:	80
Singletons:	27
Features:	VarIntv_C
	AvgIntv_C

(b) Cluster Scheme Details

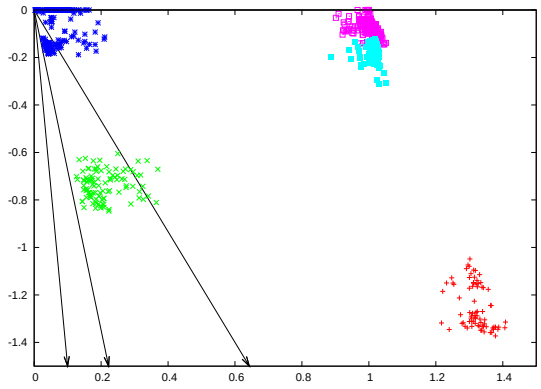
	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	36	43.1	1175	169	14.4	46.0	WWW
WWW	25	69.5	1175	851	72.4	77.0	WWW
MAIL	0	-	1175	69	5.9	75.8	WWW
7700	12	50.0	1175	62	5.3	30.0	WWW
DNS	7	64.7	1175	24	2.0	32	WWW

(c) Clustering Scheme Breakdown by Class

Appendix D

WIDE Clusters with Variance Selected Features (Full)

Fig. D.1: Clustering Outcome Using Top 4 Features of Full Wide Feature Set



(a) Five Largest Clusters

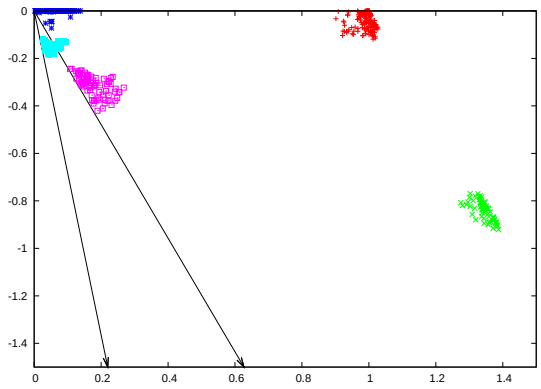
Clusters:	80
Singletons:	17
Features:	Duration_S
	VarSize_C/S
	VarSize_S
	AvgSize_C/S

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	30	43.8	187	75	40.1	20.4	OTHER
WWW	46	77.6	817	652	79.8	59.0	WWW
MAIL	0	nan	817	52	6.4	57.1	WWW
7700	4	51.9	187	53	28.3	25.6	OTHER
DNS	0	nan	187	22	11.8	29.3	OTHER

(c) Clustering Scheme Breakdown by Class

Fig. D.2: Clustering Outcome Using Top 3 Features of Full Wide Feature Set



(a) Five Largest Clusters

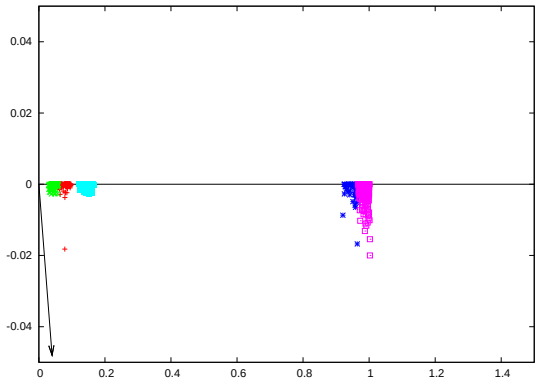
Clusters:	80
Singletons:	11
Features:	Duration_S
	VarSize_C/S
	VarSize_S

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	30	42.8	171	63	36.8	46.6	OTHER
WWW	43	77.7	733	589	80.4	53.3	WWW
MAIL	2	55.6	733	47	6.4	52.2	WWW
7700	5	46.7	171	45	26.3	21.7	OTHER
DNS	0	nan	171	19	11.1	25.3	OTHER

(c) Clustering Scheme Breakdown by Class

Fig. D.3: Clustering Outcome Using Top 2 Features of Full Wide Feature Set



(a) Five Largest Clusters

Clusters:	80
Singletons:	26
Features:	Duration_S
	VarSize_C/S

(b) Cluster Scheme Details

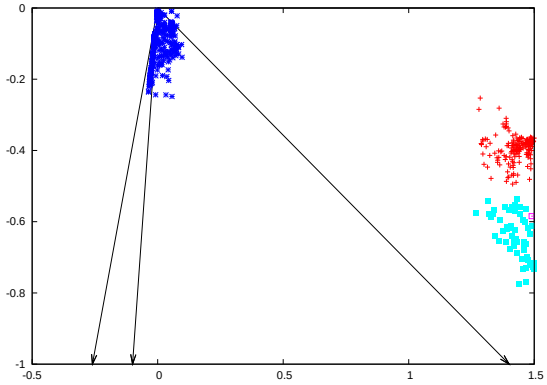
	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	28	43.2	311	136	43.7	37.0	OTHER
WWW	41	76.6	593	486	82.0	44.9	WWW
MAIL	0	nan	593	38	6.4	42.2	WWW
7700	11	68.3	311	72	23.2	34.5	OTHER
DNS	0	nan	311	41	13.2	54.7	OTHER

(c) Clustering Scheme Breakdown by Class

Appendix E

WIDE Clusters with kNN Selected Features

Fig. E.1: Clustering Outcome Using Top 4 kNN Features of Wide Feature Set



(a) Five Largest Clusters

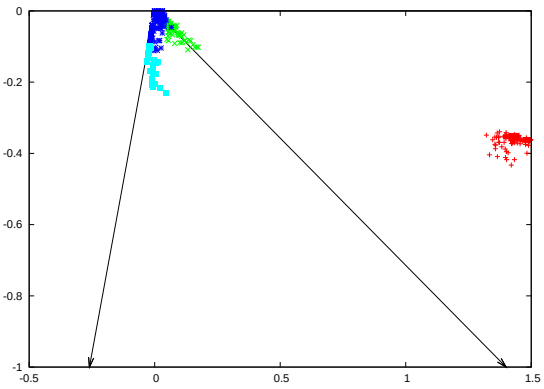
Clusters:	80
Singletons:	22
Features:	IntvHomo_S
	SizeHomo_S
	AvgSize_S
	AvgIntv_S

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	36	50.4	909	110	12.1	27.5	WWW
WWW	27	69.5	909	675	74.3	61.0	WWW
MAIL	1	100.0	909	49	5.4	53.8	WWW
7700	14	43.7	909	53	5.8	25.6	WWW
DNS	2	100.0	909	22	2.4	29.3	WWW

(c) Clustering Scheme Breakdown by Class

Fig. E.2: Clustering Outcome Using Top 3 kNN Features of Wide Feature Set



(a) Five Largest Clusters

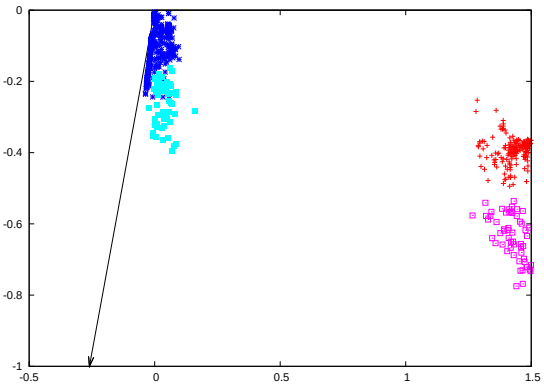
Clusters:	80
Singletons:	24
Features:	IntvHomo_S
	SizeHomo_S
	AvgSize_S

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	32	46.5	793	107	13.5	29.1	WWW
WWW	35	65.9	793	574	72.4	51.9	WWW
MAIL	0	nan	793	42	5.3	46.1	WWW
7700	11	53.3	793	56	7.1	27.0	WWW
DNS	2	100.0	793	14	1.8	18.6	WWW

(c) Clustering Scheme Breakdown by Class

Fig. E.3: Clustering Outcome Using Top 2 kNN Features of Wide Feature Set



(a) Five Largest Clusters

Clusters:	80
Singletons:	19
Features:	IntvHomo_S
	SizeHomo_S

(b) Cluster Scheme Details

	Rankings By Class		Cluster Containing Most Class Flows				
Class	Clusters	Purity	Total	Class	Purity	% of Class	Classification
OTHER	33	44.3	118	41	34.7	11.2	OTHER
WWW	36	67.7	556	455	81.8	41.1	WWW
MAIL	1	42.9	556	36	6.5	40.0	WWW
7700	9	72.7	118	32	27.1	15.5	OTHER
DNS	1	100.0	65	11	16.9	86.7	WWW

(c) Clustering Scheme Breakdown by Class

Bibliography

[1] G. Gu, R. Perdisci, J. Zhang, and W. Lee. "Botminer: clustering analysis of network traffic for protocol- and structure-independent botnet detection," in *SS'08: Proceedings of the 17th conference on Security symposium*. Berkeley, CA, USA: USENIX Association, pp. 139-154, 2008.

[2] "Introduction to Cisco IOS NetFlow - A Technical Overview," Technical Whitepaper. Oct. 2007. Available: http://www.cisco.com.

[3] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. pp. 517-564. Wiley-Interscience, New York, 2001.

[4] A. Mcgregor, M. Hall, P. Lorier, and J. Brunskill, "Flow clustering using machine learning techniques," *Passive and Active Network Measurement*. pp. 205-214, 2004.

[5] A. W. Moore and D. Zuev, "Internet traffic classification using Bayesian analysis techniques," *SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 1, pp. 50-60, June 2005.

[6] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157-1182, 2003. Available: http://dblp.uni-trier.de/rec/bibtex/journals/jmlr/GuyonE03

[7] R. Xu. "Survey of clustering algorithms," *IEEE Transitions, Neural Netrowks*, vol. 16, no. 3, pp. 645-678, May 2005.

[8] J. Erman, M. Arlitt, and A. Mahnati. "Traffic classification using clustering algorithms," *Proceedings of the 2006 SIGCOMM workshop on Mining network data*, p.281-286, September 11-15, 2006, Pisa, Italy.

[9] "Hierarchial clustering explorer," Available: http://www.cs.umd.edu/hcil/hce.

[10] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. V50, no. 2, pp. 159-179, June 1985.

[11] L. Hubert. "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193-218, July 2005.

[12] E. Kandogan, "Star coordinates: A multi-dimensional visualization technique with uniform treatment of dimensions," in *In Proceedings of the IEEE Information Visualization Symposium, Late Breaking Hot Topics*, pp. 9-12, 2000.

[13] Y. Zhang, P, Patankar, G. Kesidis, and D. Miller. "Comparative Empirical Study of Advanced Statistical Classification Techniques," 2008.

[14] A. W. Moore, D. Zuev, and M. Crogan. "Discriminators for use in flow-based classification," *Technical Report, RR-05-13, Department of Computer Science*, Queen Mary, University of London, August, 2005. Data available http://www.cl.cam.ac.uk/research/srg/netos/nprobe/data/papers/sigmetrics/index.html

[15] A. Moore, J. Hall, C. Kreibich, E. Harris, and I. Pratt, "Architecture of a network monitor," *Passive & Active Measurement Workshop (PAM)*, 2003.

[16] A. W. Moore and K. Papagiannaki, "Toward the accurate identification of network applications," *Passive and Active Network Measurement*, pp. 41-54, 2005.

[17] MAWI Working Group Traffic Archive. Available: http://tracer.csl.sony.co.jp/mawi/.

[18] P. Mitra, C. A. Murthy, and S. K. Pal, "Unsupervised feature selection using feature similarity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, 2002.

Vita
Steven Coulter

Education

The Pennsylvania State University State College, Pennsylvania 2006–Present
B.S. in Computer Engineering, expected May 2010

The Pennsylvania State University State College, Pennsylvania 2006–Present
B.S. in Mathematics, expected May 2010

Westmoreland County Community College Pennsylvania 2001–2003
A.A.S In Computer Technology, Networking, 2003
Honors

Westmoreland County Community College Pennsylvania 2001–2003
A.A.S In Computer Technology, Telcommunications, 2000
High Honors

Westmoreland County Community College Pennsylvania 2001–2003
Diploma In Computer Technology, 2000
High Honors

Awards and Honors

H. Thomas and Dorothy Willits Hallowell Scholars Endowment 2006–Present
Eve Joran Willard Trustee Scholarship 2008-Present
Lockheed Martin Engineering Scholars Award 2007
HRB Systems Student Scholarship 2007

Research Experience

Undergraduate Research Pennsylvania State University 2009–Present
Research Advisor: Dr. James Brannick
Investigation of Spectral Properties of Block Jacobi Methods for Domain Wall Formation of Quantum Chromodynamics.

Projects and Activities

Undergraduate Projects Pennsylvania State University 2009–Present
Project Advisor: Dr. Timothy Wheeler
Development of a wireless three-dimensional positioning system to be used within the interior of an aircraft fuselage.