Summarization - Compressing Data into an Informative Representation

Varun Chandola
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455
chandola@cs.umn.edu

Vipin Kumar
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455
kumar@cs.umn.edu

Abstract

In this paper, we formulate the problem of summarization of a dataset of transactions with categorical attributes as an optimization problem involving two objective functions - compaction gain and information loss. We propose metrics to characterize the output of any summarization algorithm. We investigate two approaches to address this problem. The first approach is an adaptation of clustering and the second approach makes use of frequent itemsets from the association analysis domain. We illustrate one application of summarization in the field of network data where we show how our technique can be effectively used to summarize network traffic into a compact but meaningful representation. Specifically, we evaluate our proposed algorithms on the 1998 DARPA Off-line Intrusion Detection Evaluation data and network data generated by SKAION Corp for the ARDA information assurance program.

1 Introduction

Summarization is a key data mining concept which involves techniques for finding a compact description of a dataset. Simple summarization methods such as tabulating the mean and standard deviations are often applied for exploratory data analysis, data visualization and automated report generation. Clustering [12, 21] is another data mining technique that is often used to summarize large datasets. For example, centroids of document clusters derived for a collection of text documents can provide a good indication of the topics being covered in the collection. The clustering based approach is effective in domains like text summarization, where the features are asymmetric binary [21, 9], and hence cluster centroids are a meaningful description of the clusters. However, if the data has categorical attributes, then the standard methods for computing a cluster centroid are not applicable and hence clustering cannot directly be applied for summarization[1]. One such application is in the analysis of netflow data to detect cyber attacks.

Feature	Type	Possible Values
Source IP	Categorical	2^{32}
Source Port	Categorical	2^{16}
Destination IP	Categorical	2^{32}
Destination Port	Categorical	2^{16}
Protocol	Categorical	≤ 10
Number of Packets	Continuous	1 - ∞
Number of Bytes	Continuous	1 - ∞
TCP Flags	Categorical	≤ 10

Table 1. Different features for netflow data

Netflow data is a set of records that describe network traffic, where each record has different features such as the IPs and ports involved, packets and bytes transferred (see Table 1). An important characteristic of netflow data is that it has a mix of categorical and continuous features. The volume of netflow data which a network analyst has to monitor is huge. For example, on a typical day at the University of Minnesota, more than one million flows are collected in every 10 minute window. Manual monitoring of this data is impossible and motivates the need for data mining techniques. Anomaly detection systems [8, 16, 4, 20] can be used to score these flows, and the analyst typically looks at only the most anomalous flows to identify attacks or other undesirable behavior. Table 2 shows 17 flows which were ranked as most suspicious by the MINDS Anomaly Detection Module [8] for the network traffic analyzed on January 26, 2003 (48 hours after the *Slammer Worm* hit the Internet). These flows are involved in three anomalous activities - *slammer worm* related traffic on port 1434, flows asso-

[1]Traditionally, a centroid is defined as the average of the value of each attribute over all transactions. If a categorical attribute has different values (say red, blue, green) for three different transactions in the cluster, then it does not make sense to take an average of the values. Although it is possible to replace a categorical attribute with an asymmetric binary attribute for each value taken by the attribute, such methods do not work well when the attribute can take a large number of values, as in the netflow data – see Table 1.

	Score	srcIP	sPort	dstIP	dPort	prot	pkts	bytes
T_1	37675	63.150.X.253	1161	128.101.X.29	1434	udp	[0,2)	[0,1829)
T_2	26677	63.150.X.253	1161	160.94.X.134	1434	udp	[0,2)	[0,1829)
T_3	24324	63.150.X.253	1161	128.101.X.185	1434	udp	[0,2)	[0,1829)
T_4	21169	63.150.X.253	1161	160.94.X.71	1434	udp	[0,2)	[0,1829)
T_5	19525	63.150.X.253	1161	160.94.X.19	1434	udp	[0,2)	[0,1829)
T_6	19235	63.150.X.253	1161	160.94.X.80	1434	udp	[0,2)	[0,1829)
T_7	17679	63.150.X.253	1161	160.94.X.220	1434	udp	[0,2)	[0,1829)
T_8	8184	63.150.X.253	1161	128.101.X.108	1434	udp	[0,2)	[0,1829)
T_9	7143	63.150.X.253	1161	128.101.X.223	1434	udp	[0,2)	[0,1829)
T_{10}	5139	63.150.X.253	1161	128.101.X.142	1434	udp	[0,2)	[0,1829)
T_{11}	4048	142.150.Y.101	0	128.101.X.142	2048	icmp	[2,4)	[0,1829)
T_{12}	4008	200.250.Z.20	27016	128.101.X.116	4629	udp	[2,4)	[0,1829)
T_{13}	3657	202.175.Z.237	27016	128.101.X.116	4148	udp	[2,4)	[0,1829)
T_{14}	3451	63.150.X.253	1161	128.101.X.62	1434	udp	[0,2)	[0,1829)
T_{15}	3328	63.150.X.253	1161	160.94.X.223	1434	udp	[0,2)	[0,1829)
T_{16}	2796	63.150.X.253	1161	128.101.X.241	1434	udp	[0,2)	[0,1829)
T_{17}	2694	142.150.Y.101	0	128.101.X.168	2048	icmp	[2,4)	[0,1829)

Table 2. Top 17 anomalous flows as scored by the anomaly detection scheme of the MINDS system for the network data collected on January 26, 2003 at the University of Minnesota (48 hours after the *Slammer Worm* hit the Internet). The third octet of IPs is anonymized for privacy preservation.

	Size	Score	srcIP	sPort	dstIP	dPort	prot	pkts
S_1	13	15102	63.150.X.253	1161	***	1434	udp	[0,2)
S_2	2	3833	***	27016	128.101.X.116	***	udp	[2,4)
S_3	2	3371	142.150.Y.101	0	***	2048	icmp	[2,4)

Table 3. Summarization output for the dataset in Table 2. The last column has been removed since all the transactions contained the same value for it in the original dataset.

ciated with a *half-life* game server on port 27016 and *ping scans* of the inside network by an external host on port 2048. In a typical window of data being analyzed, there are often several hundreds or thousands of highly ranked flows that require the analyst's attention. But due to the limited time available, analysts look at only the first few pages of results that cover the top few dozen most anomalous flows. If many of these most anomalous flows can be summarized into a small representation, then the analyst can analyze a much larger set of anomalies than is otherwise possible. For example, if the dataset shown in Table 2 can be automatically summarized into the form shown in Table 3 (the last column has been removed since all the transactions contained the same value for it in Table 2), then the analyst can look at only 3 lines to get a sense of what is happening in 17 flows. Table 3 shows the output summary for this dataset generated by an application of our proposed scheme. We see that every flow is represented in the summary. The first summary S_1 represents flows $\{T_1$-T_{10},T_{14}-$T_{16}\}$ which correspond to the *slammer worm* traffic coming from a single external host and targeting several internal hosts. The second summary S_2 represents flows $\{T_{12}$,$T_{13}\}$ which are the connections made to *half-life* game servers made by an internal host. The third summary, S_3 represents flows $\{T_{11}$,$T_{17}\}$ which correspond to a *ping scan* by the external host. In general, such summarization has the potential to reduce the size of the data by several orders of magnitude.

In this paper, we address the problem of summarization of data sets that have categorical features. We handle continuous features by discretizing them using equal-width binning and then treating the resulting features as categorical. We view summarization as a transformation from a given dataset to a smaller set of individual summaries with an objective of retaining the maximum information content. A fundamental requirement is that *every data item should be represented in the summary*.

1.1 Contributions

Our contributions in this paper are as follows –

- We formulate the problem of summarization of transactions that contain categorical data, as a dual-optimization problem and characterize a good summary using two metrics – *compaction gain* and *information loss*. Compaction gain signifies the amount of reduction done in the transformation from the actual data to a summary. Information loss is defined as the total amount of information missing over all original data transactions in the summary.
- We investigate two approaches to address this problem. The first approach is an adaptation of clustering and the second approach makes use of frequent itemsets from the association analysis domain [3].
- We illustrate one application of summarization in the field of network data where we show how our technique can be effectively used to summarize network traffic into a compact but meaningful representation. Specifically, we evaluate our proposed algorithms on the 1998 DARPA Off-line Intrusion Detection Evaluation data [14] and network data generated by SKAION Corp for the ARDA information assurance program [1].

2 Related Work

Many researchers have addressed the issue of finding a compact representation of frequent itemsets [2, 19, 10, 18, 6]. However, their final objective is to approximate a collection of frequent itemsets with a smaller subset, which is different from the problem addressed in this paper, in which we try to represent a collection of transactions with a smaller summary. Text summarization [17] is a widely-researched topic in the research community, and has been addressed mostly as a natural language processing problem which involves semantic knowledge and is different from the problem of summarization of transaction data addressed in this paper. Another form of summarization is addressed in [11] and [15], where the authors aim at organizing and summarizing individual rules for better visualization while not addressing the issue of compacting the data.

	src IP	sPort	dst IP	dPort	pro	flags	packets	bytes
T_1	12.190.84.122	32178	100.10.20.4	80	tcp	—APRS-	[2,20]	[504,1200]
T_2	88.34.224.2	51989	100.10.20.4	80	tcp	—APRS-	[2,20]	[220,500]
T_3	12.190.19.23	2234	100.10.20.4	80	tcp	—APRS-	[2,20]	[220,500]
T_4	98.198.66.23	27643	100.10.20.4	80	tcp	—APRS-	[2,20]	[42,200]
T_5	192.168.22.4	5002	100.10.20.3	21	tcp	—A-RSF	[2,20]	[42,200]
T_6	192.168.22.4	5001	100.10.20.3	21	tcp	—A-RS-	[40,68]	[220,500]
T_7	67.118.25.23	44532	100.10.20.3	21	tcp	—A-RSF	[40,68]	[42,200]
T_8	192.168.22.4	2765	100.10.20.4	113	tcp	—APRS-	[2,20]	[504,1200]

Table 4. A synthetic dataset of network flows.

	src IP	sPort	dst IP	dPort	pro	flags	packets	bytes
S_1	*.*.*.*	***	100.10.20.4	***	tcp	—APRS-	[2,20]	***
S_2	*.*.*.*	***	100.10.20.3	21	tcp	***	***	***
S_3	192.168.22.4	2765	100.10.20.4	113	tcp	—APRS-	[2,20]	[504,1200]

Table 5. A possible summary for the dataset shown above.

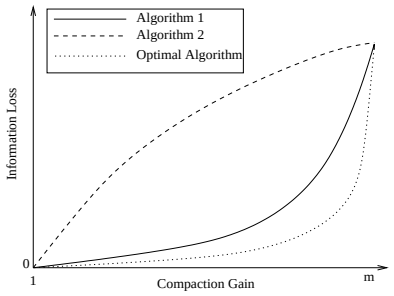

Figure 1. ICC Curve for summarization algorithms

3 Characterizing a Summary

Summarization can be viewed as compressing a given set of transactions into a smaller set of patterns while retaining the maximum possible information. A trivial summary for a set of transactions would be itself. The information loss here is zero but there is no compaction. Another trivial summary would be the empty set ϵ, which represents all the transactions. In this case the gain in compaction is maximum but the summary has no information content. A good summary is one which is small but still retains enough information about the data as a whole and also for each transaction.

We are given a set of n categorical features $F = \{F_1, F_2, \ldots, F_n\}$ and an associated weight vector W such that each $W_i \in W$ represents the weight of the feature $F_i \in F$. A set of transactions T, such that $|T| = m$, is defined using these features, and each $T_i \in T$ has a specific value for each of the n features. Formally, a summary of a set of transactions can be defined as follows:

DEFINITION 1. (**Summary**) *A summary S of a set of transactions T, is a set of individual summaries $\{S_1, S_2, \ldots, S_l\}$ such that (i) each S_j represents a subset of T and (ii) every transaction $T_i \in T$ is represented by at least one $S_j \in S$.*

Each individual summary S_j essentially a covers a set of transactions. In the summary S, these transactions are replaced by the individual summary that covers them. As we mentioned before, computing the centroid for data with categorical attributes is not possible. For such data, a feature-wise intersection of all transactions is a more appropriate description of an individual summary. Hence, from now on, an individual summary will be treated as a feature-wise intersection of all transactions covered by it, i.e., if S_j covers $\{T_1, T_2, \ldots, T_k\}$, then $S_j = \bigcap_{i=1}^{k} T_i$. For the sake of illustration let us consider the sample netflow data given in Table 4. The dataset shown is a set of 8 transactions that are described by 6 categorical features and 2 continuous features (see Table 1). Let all the features have equal weight of $\frac{1}{8}$. One summary for this dataset is shown in Table 5 as a set of 3 individual summaries. The individual summary S_1 covers transactions $\{T_1,T_2,T_3,T_4,T_8\}$, S_2 covers transactions $\{T_5,T_6,T_7\}$ and S_3 covers only one transaction, T_8.

To assess the quality of a summary S of a set of transactions T, we define following metrics -

DEFINITION 2. (**Compaction Gain for a Summary**) *Compaction Gain = $\frac{m}{l}$. (Recall that $m = |T|$ and $l = |S|$.)*

For the dataset in Table 4 and the summary in Table 5, *Compaction Gain for $S = \frac{8}{3}$.*

DEFINITION 3. (**Information Loss for a transaction represented by an individual summary**) *For a given transaction $T_i \in T$ and an individual summary $S_j \in S$ that covers T_i, $loss_{ij}$ = $\sum_{q=1}^{n} W_q * b_q$, where, $b_q = 1$ if $T_{iq} \notin S_j$ and 0 otherwise.*

The loss incurred if a transaction is represented by an individual summary will be the weighted sum of all features that are absent in the individual summary.

DEFINITION 4. (**Best Individual Summary for a transaction**) For a given transaction $T_i \in T$, a best individual summary $S_j \in S$ is the one for which $loss_{ij}$ is minimum.

The total information loss for a summary is the aggregate of the information lost for every transaction with respect to its best individual summary.

For the dataset in Table 4 and its summary shown in Table 5, transactions T_1-T_4 are best covered by individual summary S_1 and each has an information loss of $\frac{4}{8}$. Transactions T_5-T_7 are best covered by individual summary S_2 and each has an information loss of $\frac{5}{8}$. T_8 is represented by S_1 and S_3. For T_8 and S_1, information loss = $4 \times \frac{1}{8} = \frac{1}{2}$, since there are 4 features absent in S_1. For T_8 and S_3, information loss = 0 since there are no features absent in S_3. Hence the best individual summary for T_8 will be S_3. Thus, we get that *Information Loss for S =* $\frac{4}{8} \times 4 + \frac{5}{8} \times 3 + 0 = \frac{31}{8} = 3.875$.

	src IP	sPort	dst IP	dPort	protocol	flags	packets	bytes
C_1	*.*.*.*	***	100.10.20.4	***	tcp	—APRS-	[2,20]	***
C_2	*.*.*.*	***	100.10.20.3	21	tcp	***	***	***

Table 6. A summary obtained for the dataset in Table 4 using the clustering based algorithm

It is to be noted that the characteristics, *compaction gain* and *information loss*, follow an optimality tradeoff curve as shown in Figure 1 such that increasing the compaction results in increase of information loss. We denote this curve as ICC (*Information-loss Compression-gain Characteristic*) curve.

The ICC curve is a good indicator of the performance of a summarization algorithm. The beginning and the end of the curve are fixed by the two trivial solutions discussed earlier. For any summarization algorithm, it is desirable that the area under its ICC curve be minimal. It can be observed that getting an optimal curve as shown in Figure 1 involves searching for a solution in exponential space and hence not feasible. But a good algorithm should be close enough to the optimal curve like 1 and not like 2 in the figure shown.

As the ICC curve indicates, there is no global maxima for this dual-optimization problem since it involves two orthogonal objective functions. So a typical objective of a summarization algorithm would be - *for a given level of compaction find a summary with the lowest possible information loss.*

4 Summarization Using Clustering

In this section, we present a direct application of clustering to obtain a summary for a given set of transactions with categorical attributes. This simple algorithm involves clustering of the data using any standard clustering algorithm and then replacing each cluster with a representation as described earlier using feature-wise intersection of all transactions in that cluster. The weights W are used to calculate the distance between two data transactions in the clustering algorithm. Thus, if $\bar{C}$ is a set of clusters obtained from a set of transactions T by clustering, then each cluster produces an individual summary which is essentially the set of feature-value pairs which are present in all transactions in that cluster. The number of clusters here determine the compaction gain for the summary. For illustration consider again the sample dataset of 8 transactions in Table 4. Let clustering generate two clusters for this dataset – C_1 = $\{T_1,T_2,T_3,T_4,T_8\}$ and C_2 = $\{T_5,T_6,T_7\}$. Table 6 shows a summary obtained using the clustering based algorithm.

The clustering based approach works well in representing the frequent modes of behavior in the data because they are captured well by the clusters. However, this approach performs poorly when the data has outliers and less frequent patterns. This happens because the outlying transactions are forced to belong to some cluster. If a cluster has even a single transaction which is different from other cluster members, it degrades the description of the cluster in the summary. For instance consider the clusters in Table 6. Let us assume that there is another transaction T_9 in the dataset shown in Table 4 and clustering assigns it to cluster C_1. Let the different features of T_9 be

srcip = 12.190.84.122, srcport = 32178, dstip = 100.10.20.10, dstport = 53, protocol = udp, flags = *none*, packets = [25,60], bytes = [2200,5000]

On adding T_9 to C_1, the summary generated from C_1 will be empty. The presence of this outlying transaction makes the summary description very lossy in terms of information content. Thus this approach represents outliers very poorly, which is not desirable in applications such as network intrusion detection and fraud detection where such outliers are of special interest.

5 A Two-step Approach to Summarization using Frequent Sets

In this section we propose a two-step methodology to address the problem of summarization. The basic idea is to start with a set of candidate summaries that are frequent sets derived from association pattern analysis, in addition to individual transactions. Each of these candidates represent one or more transactions. Thus a summary for the entire transaction dataset would be a subset of these candidates such that, for every transaction in T, there is at least one candidate in this subset that covers the transaction. In this context, the summarization problem can be viewed as - *Given a set of candidates, C and a desired compaction level, find a summary $S \subseteq C$ for the transaction dataset T with the least information loss.*

This problem is solved in two steps. The first is the choice of a candidate set and the second is how to select a subset of these candidates as the summary such that we optimize the information loss for a given compaction level.

One possible choice of candidate set is all frequent itemsets with a support threshold of 2 transactions, as well as individual transactions. This ensures that all possible ways of summarizing the transactions can be considered. But this can lead to too many candidates, which increases the computational complexity of the second step. Higher values of the support threshold can be used to constrain the number of possible candidates, but this can impact the quality of the summaries obtained. The second step of this approach is to select an appropriate subset of C. This can be done in several ways. We first observe that the brute-force algorithm for selecting an optimal subset of C is not feasible since it requires searching in exponential space with respect to $|C|$.

We have explored the realm of greedy algorithms to obtain *approximately* optimal solutions.

The general idea is that, starting with a set of transactions T and a set of candidates C, we want to obtain a set, $S \subseteq C$, such that every transaction in T is covered by some member of S.

One greedy way to approach this problem is to allow every transaction to select a best candidate for itself and add it to the summary (*top-down*). Similar transactions will tend to select the same candidate and hence it would result in compaction gain. Another approach is to incrementally increase the compaction of a summary by adding a best candidate from the candidate set at every step (*bottom-up*). We have investigated both approaches in our research [7], but in this paper we discuss only the bottom-up algorithm due to space limitations.

5.1 BUS - A Bottom-up Summarization Algorithm

In this section we present an incremental bottom-up algorithm - **BUS**. The main idea behind this algorithm is to incrementally select best candidates from the candidate set such that at each step, for a certain gain in compaction, minimum information loss is incurred. The inputs to this algorithm are - the set of transactions, T, the set of candidates, C, the initial value k_s^{init} of the tradeoff parameter k_s and the increment δ_k for k_s.

Before describing the algorithm we first define the scoring function which we use to determine the best candidate.

DEFINITION 5. (**Score of a Candidate Summary**) For a given candidate summary $C_i \in C$, its score is given by

$$score_i = k_s \times size_{C_i} - loss_{C_i}$$

$size_{C_i}$ refers to the compaction, and $loss_{C_i}$ refers to the information loss caused by adding C_i to the summary.

k_s is a trade-off parameter which determines which entity to favor – higher compaction gain or lower information loss. A low value of k_s favors very specific candidate summaries which result in lower information loss, while a higher value for k_s favors more general candidate summaries which cause a higher compaction gain.

DEFINITION 6. (**Size of a Candidate**) Size of a candidate, $C_i \in C$, is defined as, $size_{C_i}$ = # individual summaries in S_c, the current summary, which are covered[2] by C_i.

Let the individual summaries in S_c covered by C_i be $\{S_1, S_2, \ldots, S_{size_{C_i}}\}$. Note that each of these individual summaries can be either transactions or candidates.

[2]An individual summary is covered by C_i if it is more specific than C_i

DEFINITION 7. (**Loss for a Candidate**) Loss for candidate C_i is defined as, $loss_{C_i} = \sum_{j=1}^{size_{C_i}} (l_{C_i} - l_{S_j})$, where $l_{Candidate} = \sum_{k=1}^{f} W_k * b_k$, such that b_k = 1 if $F_k \notin Candidate$ and 0 otherwise.

The algorithm starts by considering the initial summary as the set of transactions T, which has no information loss but no compaction gain. Definitions 6 and 7 are used to compute the score for each candidate in C using Definition 5. The candidate with the highest score is selected and added to the current summary, S_c, replacing all summaries that are covered by it. If the candidate with maximum score has already been added to the summary, the value of k_s is incremented by δ_k. The size of each of the candidates is revised equal to the number of individual summaries in S_c covered by that candidate.

From the ICC curve perspective, the algorithm moves from no compaction gain to higher compaction gain in small steps determined by the value k_s. The initial value for k_s is chosen as 0. This ensures that initially, candidates with minimum information loss are selected. These will tend to be very specific and hence the overall compaction gain will be low. After all such candidates are chosen, the value of k_s is incremented by δ_k so that more general candidates with larger sizes are selected.

The selection criterion of this algorithm ensures that the increment in the compaction of the global summary at any step incurs the minimum possible information loss. Thus this algorithm ensures that we attain a locally optimal solution at any step. Starting with a very low value of k_s and incrementing it in very small amounts ensures that the summary obtained at any step is also very close to the globally optimal solution for that particular compaction.

The first step of BUS involves generation of frequent itemsets from the data using any association rule mining algorithm. The categorical attributes are binarized before applying the algorithm. The computational complexity of this step depends on the size and nature of the data. It can be controlled using the support threshold. At the University of Minnesota, the summarization module of MINDS takes less than 10 seconds to generate frequent itemsets for the top 5000 anomalous connections with a support threshold of 8 transactions. By using *closed* frequent itemsets [18], the number of frequent itemsets can be pruned considerably. In the second step of BUS, in each iteration, the algorithm selects the best candidate to be added to the current summary. The size of the current summary cannot be more than the total number of transactions. Thus, if the transaction dataset is of of size m, the candidate set is of size l and the algorithm runs for k iterations, the computational complexity of the second step will be $O(mlk)$. In our experiments (as discussed in Section 6), we ran BUS to generate summaries of sizes ranging from the size of the data itself down to 5. The

time taken for a dataset of size 8459 was under 5 minutes and for a dataset of size 2903, it was under 2 minutes, with a support threshold of 2 transactions.

6 Experimental Evaluation And Results

In this section we present the performance of our proposed algorithms on network data. We compare the performance of BUS with the clustering based approach to show that it performs better in terms of achieving lower information loss for a given degree of compaction. We also illustrate the summaries obtained for different algorithms to make a qualitative comparison between them. The algorithms were implemented in GNU-C++ and were run on the Linux platform on a 4-processor *intel*-i686 machine.

6.1 Input Data

We ran our experiments on two different artificial datasets generated by DARPA [14] and SKAION corporation [1] for the evaluation of intrusion detection systems. The DARPA dataset is publicly available and has been used extensively in the data mining community as it was used in KDD Cup 1999. The SKAION data was developed as a part of the ARDA funded program on information assurance and is available only to the investigators involved in the program. Both these datasets have a mixture of normal and attack traffic. The SKAION dataset had 8459 flows. The DARPA dataset was a subset of the week 4, Friday, training data containing only attack related traffic corresponding to the following attacks - *warezclient, rootkit, ffb, ipsweep, loadmodule* and *multihop*. The size of this dataset was 2903 flows.

Both these datasets exhibit different characteristics in terms of data distribution. Figure 2(a) gives the distribution of the *lof* (local outlier factor) score (see [5]) for the transactions in the SKAION dataset. The *lof* score reflects the outlierness of a transaction with respect to its nearest neighbors. The transactions which belong to tight clusters tend to have low *lof* scores while outliers have high *lof* scores. For the SKAION dataset we observe that there are a lot of transactions which have high outlier scores. The *lof* distribution for the DARPA dataset in Figure 2(e) shows that most of the transactions belong to tight clusters , and only a few transactions are outliers.

6.2 Comparison of ICC curves for the clustering-based algorithm and BUS

We ran the clustering based algorithm by first generating clusters of different sizes using the *CLUTO* hierarchical clustering package [13]. For finding the similarity between transactions, the features were weighted as per the scheme used for evaluating the information loss incurred by a summary. We then summarized the clusters as explained in Section 4. For BUS, we present the results using frequent itemsets generated by the *apriori* algorithm with a support threshold of 2 as the candidates. The BUS algorithm was executed with initial value of $k_s = 0$ and the increment, $\delta_k = 0.1$. The different features in the data and the weights used are given in Table 7. These weights reflect the typical relative importance given to the different features by network analysts. The continuous attributes in the data were discretized using *equal depth binning* technique with a fixed number of intervals (= 75).

feature name	weight
Source IP	3.5
Source Port	0
Destination IP	3.5
Destination Port	2
Protocol	0.1
Time to Live(ttl)	0.1
TCP Flags	0.1
Number of Packets	0.3
Number of Bytes	0.3
Window Size	0.1

Table 7. Different features and their weights used for experiments.

Figures 2(b) and 2(f) show the ICC curves for the clustering-based algorithm and BUS on the DARPA and SKAION data sets respectively. From the two graphs we can see that BUS performs better than the clustering-based approach. We also observe that the difference in the curves for each case reflects the *lof* score distribution for each dataset. In the SKAION dataset there are a lot of outliers which are represented poorly by the clustering-based approach while BUS handles them better. Hence the difference in the information loss is very high. In the DARPA dataset, most of the transactions belong to well-defined clusters which are represented equally well by both the algorithms. Thus, the difference in information loss for the two algorithms is not very high in this case.

To further strengthen our argument that clustering tends to ignore the infrequent patterns and outliers in the data, we plot the information loss for transactions which have lost a lot of information in the summary. Figure 2(c) shows the difference in the ICC curves for the transactions in the DARPA dataset which have lost more than 70% information. The graph shows that for BUS, none of the transactions lose more than 70% information till a compaction gain of about 220, while for the clustering based approach, there are considerable number of transactions which are very poorly represented even for a compaction gain of 50. A similar result for the SKAION dataset in Figure 2(g) shows that BUS generates summaries in which very few transac-

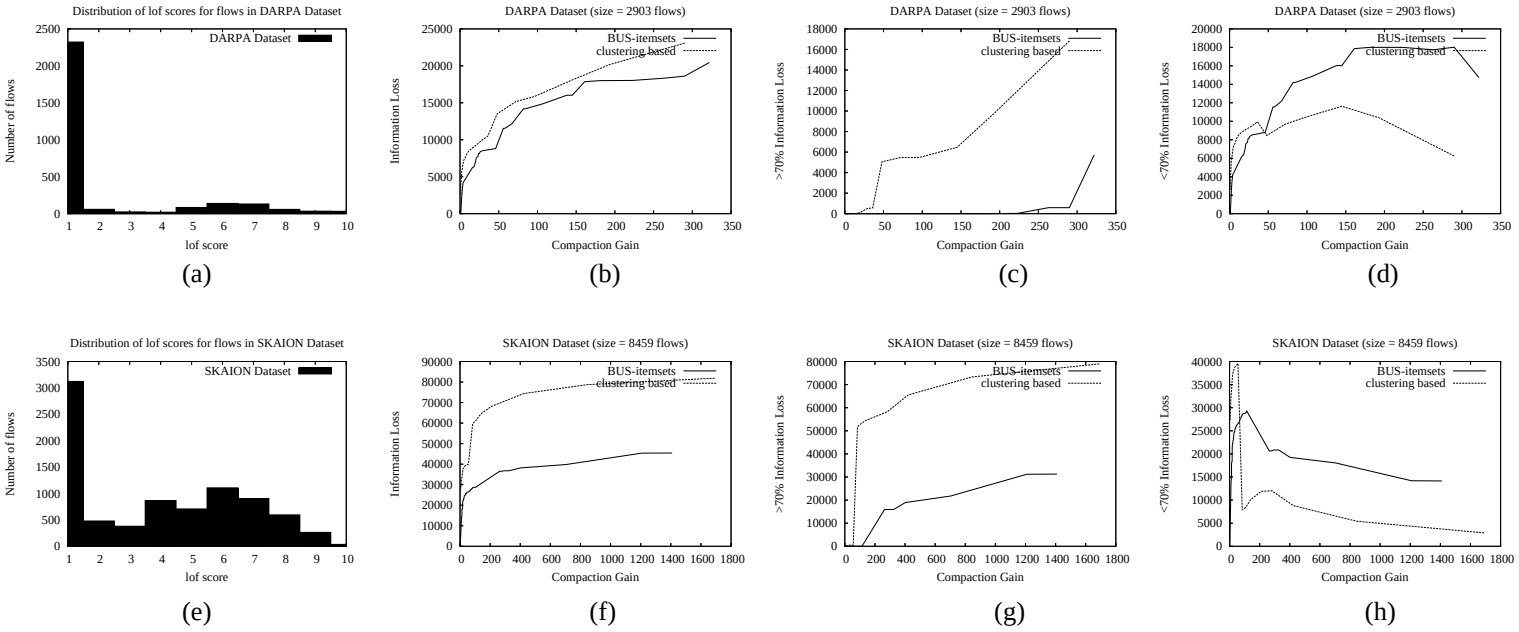

Figure 2. Figures (a) – (d) present results for the DARPA dataset, Figures (e) – (h) present results for SKAION dataset. (a,e) Distribution of *lof* scores. (b,f) ICC Curve for the clustering based algorithms and BUS. (c,g) Sum of the Information Loss for transactions that have lost more than 70 % of information. (d,h) Sum of the Information Loss for transactions that have lost less than 70 % information.

tions have a high loss, which is not true in the case of the clustering based approach.

Figure 2(d) shows the difference in the ICC curves for each algorithm for the transactions which have lost less than 70% of information for the DARPA dataset. This plot illustrates the difference in behavior of the two algorithms in terms of summarizing the transactions which belong to some frequent pattern in the data. The clustering based approach represents these transactions better than BUS. A similar result can be seen for the SKAION dataset in Figure 2(h).

6.3 Qualitative Analysis of Summaries

In this section we illustrate the summaries obtained by running the clustering based algorithm (see Table 8), and BUS using frequent itemsets (see Table 9) on the DARPA dataset described above. This dataset is comprised of different attacks launched on the internal network by several external machines. The tables do not contain all the features due to the lack of space. However, the information loss was computed using all the features shown in Table 7.

In the summary obtained from the clustering based approach, we observe that S_1 and S_3 correspond to the *icmp* and *udp* traffic in the data. Summaries S_2, S_4 and S_6 represent the *ftp* traffic on port 20, corresponding to the *warezclient*, *laodmodule* and *ffb* attacks which involve illegal *ftp* transfers. S_5 represents traffic on port 23 which correspond to the *rootkit* and *multihop* attacks. The rest of the summaries, S_7-S_{10}, do not have enough information as most of the features are missing. These cover most of the infrequent patterns and the outliers which were ignored by the clustering algorithm. Thus we see that the clustering based algorithm manages to bring out only the frequent patterns in the data. The summary obtained from BUS gives a much bet-

	size	src IP	sPort	dst IP	dPort	proto	packets	bytes
S_1	513	***	0	***	0	icmp	[1,1]	[28,28]
S_2	51	172.16.112.50	20	***	***	tcp	***	***
S_3	119	***	***	***	***	udp	***	***
S_4	362	197.218.177.69	20	***	***	tcp	[5,5]	***
S_5	141	***	***	***	23	tcp	***	***
S_6	603	172.16.114.148	20	***	***	tcp	***	***
S_7	507	***	***	***	***	tcp	***	***
S_8	176	***	***	***	***	tcp	***	***
S_9	249	***	***	***	***	tcp	***	***
S_{10}	182	***	***	***	***	tcp	***	***

Table 8. A size 10 summary obtained for DARPA dataset using the clustering based algorithm. Information Loss=23070.5

ter representation of the data. Almost all the summaries in this case contain one of the IPs (which have high weights), which is not true for the output of the clustering-based algorithm. Summaries S_1 and S_2 represent the *ffb* and *loadmodule* attacks since they are launched by the same source IP. The *warezclient* attack on port 21 is represented by S_3. The

	size	src IP	sPort	dst IP	dPort	proto	packets	bytes
S_1	279	***	***	135.13.216.191	***	***	***	***
S_2	364	135.13.216.191	***	***	***	***	***	***
S_3	138	***	***	***	21	tcp	***	***
S_4	76	172.16.112.50	***	***	***	***	***	***
S_5	249	***	***	197.218.177.69	***	***	***	***
S_6	1333	197.218.177.69	***	***	***	***	***	***
S_7	629	172.16.114.148	***	***	***	tcp	***	***
S_8	153	***	***	***	23	tcp	***	***
S_9	1	172.16.114.50	23	207.230.54.203	1028	tcp	[1,1]	[41,88]
S_{10}	5	***	0	197.218.177.69	0	icmp	[1,1]	[28,28]

Table 9. A size 10 summary obtained for DARPA dataset using BUS algorithm. Information Loss=18601.7

ipsweep attack, which is essentially a single external machine scanning a lot of internal machines on different ports, is summarized in S_6. S_5 summarizes the connections which correspond to internal machines which replied to this scanner. The real advantage of this scheme can be seen if we observe summary S_9 which is essentially a single transaction. In the data, this is the only connection between these two machines and corresponds to the *rootkit* attack. The BUS algorithm preserves this outlier even for such a small summary because there is no other pattern which covers it without losing too much information. Similarly, S_{10} represents 5 transactions which are *icmp* replies to an external scanner by 5 internal machines. Note that these replies were not merged with the summary S_5 but were represented as such. Thus, we see that summaries generated by BUS algorithm represent the frequent as well as infrequent patterns in the data.

7 Concluding Remarks and Future Work

The two schemes presented for summarizing transaction datasets with categorical attributes demonstrated their effectiveness in the context of network traffic analysis. A variant of our proposed two-step approach is used routinely at the University of Minnesota as a part of the MINDS system to summarize several thousand anomalous netflows into just a few dozen summaries. This enables the analyst to visualize the suspicious traffic in a concise manner and often leads to the identification of attacks and other undesirable behavior that cannot be captured using widely used intrusion detection tools such as SNORT.

Future work involves using clusters along with frequent itemsets as candidates for the BUS algorithm. Another possibility is to incorporate the knowledge of the anomaly scores of the network connections to be summarized, as well as normal behavior to generate summaries which capture the anomalous behavior of the highly ranked transactions in a ranked dataset.

Acknowledgements

The authors thank Gaurav Pandey and Shyam Boriah for their extensive comments on an earlier draft of the paper.

This work was supported by Army High Performance Computing Research Center contract number DAAD19-01-2-0014, by the ARDA Grant AR/F30602-03-C-0243 and by the NSF grant IIS-0308264. The content of this work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. Access to computing facilities was provided by the AHPCRC and the Minnesota Computing Institute.

References

[1] SKAION Corporation. SKAION Intrusion Detection System Evaluation Data.

[2] F. Afrati, A. Gionis, and H. Mannila. Approximating a collection of frequent sets. In *KDD '04*.

[3] R. Agrawal, T. Imieliski, and A. Swami. Mining association rules between sets of items in large databases. In *SIGMOD '93*.

[4] D. Barbara, J. Couto, S. Jajodia, and N. Wu. ADAM: A testbed for exploring the use of data mining in intrusion detection. *SIGMOD Rec.*, 30(4):15–24, 2001.

[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *SIGMOD '00*.

[6] T. Calders and B. Goethals. Mining all non-derivable frequent itemsets. In *PKDD '02*.

[7] V. Chandola and V. Kumar. Summarization - compressing data into an informative representation. Technical Report TR 05-024, Dept. of Computer Science, University of Minnesota, Minneapolis, MN, USA, 2005.

[8] L. Ertöz, E. Eilertson, A. Lazarevic, P.-N. Tan, V. Kumar, J. Srivastava, and P. Dokas. MINDS - Minnesota Intrusion Detection System. In *Data Mining - Next Generation Challenges and Future Directions*. MIT Press, 2004.

[9] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000.

[10] J. Han, J. Wang, Y. Lu, and P. Tzvetkov. Mining top-k frequent closed patterns without minimum support. In *ICDM '02*.

[11] M. Hu and B. Liu. Mining and summarizing customer reviews. In *KDD '04*.

[12] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., 1988.

[13] G. Karypis. Cluto 2.1.1 software for clustering high-dimensional datasets.

[14] R. P. Lippmann et al. Evaluating intrusion detection systems - the 1998 DARPA off-line intrusion detection evaluation. In *DISCEX '00*, volume 2, pages 12–26, 2000.

[15] B. Liu, M. Hu, and W. Hsu. Multi-level organization and summarization of the discovered rules. In *KDD '00*.

[16] M. V. Mahoney and P. K. Chan. Learning non-stationary models of normal network traffic for detecting novel attacks. In *KDD '02*.

[17] I. Mani. *Advances in Automatic Text Summarization*. MIT Press, Cambridge, MA, USA, 1999.

[18] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT '99*.

[19] J. Pei, G. Dong, W. Zou, and J. Han. On computing condensed frequent pattern bases. In *ICDM '02*.

[20] S. J. Stolfo, W. Lee, P. K. Chan, W. Fan, and E. Eskin. Data mining-based intrusion detectors: An overview of the columbia ids project. *SIGMOD Rec.*, 30(4):5–14, 2001.

[21] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*, chapter 8. Addison-Wesley, April 2005.