Incorporating Multiple Cluster Models for Network Traffic Classification

Anil Kumar, Jinoh Kim, Sang C. Suh
Department of Computer Science
Texas A&M University, Commerce, TX
Email:akatta1@leomail.tamuc.edu,
jinoh.kim@tamuc.edu, sang.suh@tamuc.edu

Ganho Choi
Sysmate Inc.
1290 Dunsan-Dong Seo-Gu, Deajeon, 302-830, Korea
Email: ghchoi@sysmate.com

***Abstract*—Network traffic classification is one of the essential functions for local and ISP networks for quality of service, network usage statistics, resource provisioning, and security. With its importance, a substantial number of previous studies have explored various machine learning techniques based on network flow statistics to improve the accuracy of classification, and using a semi-supervised learning with clustering is one branch of research directions. However, we observed fairly unacceptable results from previous clustering-based techniques with our own data set recently collected. In particular, simply combining flow attributes for clustering leads to unexpectedly poor accuracy (less than 70%).**

With this observation, we first examine the impact of attributes to the classification performance, and then propose a new technique based on *multiple* trained cluster models to take multiple sets of combinations of attributes into account for classification, rather than simply merging the attributes in a *single* model. We will show that our proposed technique significantly improves classification performance by applying multiple trained models, yielding up to 95% accuracy. Our proposed technique includes a *selection* step to reach the final classification decision from outputs from individual trained models, and we will explore a set of selection strategies to see their performance impact.

I. Introduction

Accurately identifying network-based applications is of major interest for local and ISP networks for various purposes, including quality of service, network usage statistics, resource provisioning, and security [2, 7, 12]. Earlier, network protocols/applications were identified simply based on TCP/UDP port numbers. However, the traditional technique based on port-numbers are proved to be ineffective where accuracy is less than 70% [8] since many existing and emerging applications use random or non-standard port numbers [1, 10]. Also the increasing use of tunneling makes it more difficult to identify network applications based on port numbers [4]. Due to these reasons, a substantial body of research has been conducted to replace or complement the classical port-based identification.

One approach to overcome the limitation of the port-based application identification is to inspect the packet payload information using pre-constructed signature sets [13, 11] or using machine learning techniques [8]. While reportedly highly accurate, the critical drawbacks of the deep packet inspection-based strategy include its inherent limitation to encrypted traffic transformed with cryptographic keys and privacy concerns since many countries may not permit the extraction of full payload information from packets to protect individual privacy.

These limitations have suggested to utilize transport layer characteristics of the application as the differentiator for network application identification, instead of referring to payload information. From the previously proposed techniques [2, 7, 12], we can see the combination of transport layer characteristics with machine learning techniques would be an effective alternative for network traffic classification. Several techniques were proposed based on supervised learning [12], while some other techniques utilized unsupervised or semi-supervised clustering techniques [5, 2], reporting promising accuracy for network traffic classification. However, what we actually observed with our own data set recently collected are clearly less than what had been reported in the previous studies. With the K-means clustering, we observed very poor classification accuracy less than 70% with the entire attributes (proposed in [5]) and up to 91% accuracy with only average packet size attribute (as in [2] but considered the entire packets in the flow for maximum performance). The supervised technique with Adaboot produced overall better results up to 95% accuracy. It is known that supervised learning yields better classification accuracy than semi-supervised one [12], but at the same time, the latter has several advantages over supervised learning including a benefit of discovering "new" applications that have not considered in the trained model [5]. In this work, we examine a semi-supervised clustering approach, and show that it is possible to achieve comparable performance to the supervised one.

Here, one interesting observation with clustering-based classification is that it works much better with only a single attribute than with the entire attributes. This motivates us to thoroughly evaluate the significance of flow attributes to classification accuracy with a notion of *attribute groups*. An attribute group consists of a set of attributes derived from the same source of information. For example, the packet size group includes the minimum, maximum, average, and standard deviation of packet sizes, which are derived from the "packet size" information. We consider four attribute groups to examine the performance impact of their combinations: flow information group, packet size group, packet inter-arrival time group, and relative packet inter-arrival time group. From our preliminary experiments, we observed that some combinations of attributes work quite better than the other combinations, as will be presented.

From the initial observation, we may want to choose the best combination of attributes for applying clustering-based classification. However, relying only a few attributes could lead to any biased results with inconsistency for a diverse range of network applications to be classified. Alternatively, we may want to combine a set of attributes that showed relatively better results, but an interesting observation we made is that simply merging those attributes does not lead to any significant improvement. In this work, we develop a new technique based on *multiple* trained cluster models to take multiple sets of combinations of attributes into account for classification, rather than simply merging the attributes in a *single* model. One of key challenges for this approach is how to incorporate multiple trained models to make a single classification decision. To address this, we set up a set of selection strategies and examine their performance with respect to classification accuracy. We will present that our proposed technique yields up to 95% in classification accuracy, comparable to the supervised technique.

The key contributions of this paper can be summarized as follows:

- We evaluate the significance of flow attributes to classification accuracy with a notion of attribute group. We form four groups from 17 flow attributes, which include flow information group, packet size group, packet inter-arrival time group, and relative packet inter-arrival time group, based on inherent characteristics.
- From the evaluation results with the attribute groups, we examine classification accuracy with a subset of attributes that showed relatively better results, using supervised learning methods to ensure their validity.
- We present a new clustering technique that utilizes multiple trained cluster models. A set of selection methods will be presented to incorporate multiple trained models.
- We also present experimental results for evaluating the proposed technique. Experiments for sensitivity study are also conducted to see the impact of configurable parameters.

The paper organization is as follows. We provide a summary of related studies in Section II. In Section III, we examine the significance of attribute groups to classification accuracy and performance with selected attributes with supervised learning methods. We then present the new clustering technique incorporating multiple cluster models in Section IV and evaluation results are presented in Section V. Finally we conclude our presentation with a summary and future direction in Section VI.

II. Related Work

A substantial body of research has been conducted for network traffic classification with machine learning techniques. The work can broadly be divided into the following three categories:

- *Un-supervised* []: Labeling information of the training data is not available at the time of training. We use various clustering algorithms for the classification of unlabeled data[].
- *Supervised* []: We provide the labels for the flows when we train the model and then use this model to test each incoming flow whether it belongs to any of the application which is provided at the time of training[].
- *Semi-supervised* []: We provide partial labeling information at the time of training and we use clustering algorithms to cluster the training data. We use the partially available labeling information to label each cluster. Heuristics have been proposed on how to label the cluster from partial training information[]

ACAS[8]: ACAS encodes initial *n-bytes* of the payload into space and uses supervised machine learning algorithm as the classification algorithm. We observe this technique is effective in identifying applications with very high accuracy with our data set ($\approx$ 99%). Drawback of this technique is it requires completely labeled data. Which means we should know prior to the start classification it should know all the applications that classifier may encounter in future (which is not a feasible solution). It also As mentioned, however, it requires the access to the payload, which could be limited by laws due to privacy concerns. In addition, encryption plays an important role in classification, which may significantly lower the accuracy.

K-Means classification technique[5]. We ran the K-Means classification technique across by varying the number of clusters from 10 to 140, and Figure 1(a) show the result. As can be seen from the figure, classification performance is largely not acceptable with quite less than 70% accuracy.

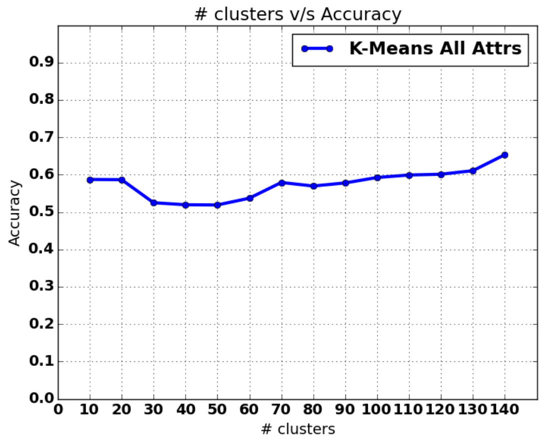
Early Application Identification.[2] We also evaluated this technique. Rather than considering only the first 4 packets in a flow as suggested in the paper, we considered the whole packets to see the maximum performance. Although this technique is based on clustering as the above technique and uses only the average packet size attribute, it yielded quite enhanced results as shown in Figure 1(b) across a diverse set of training fractions (i.e., the ratio between training data set and testing data set).

III. Grouping of Attributes

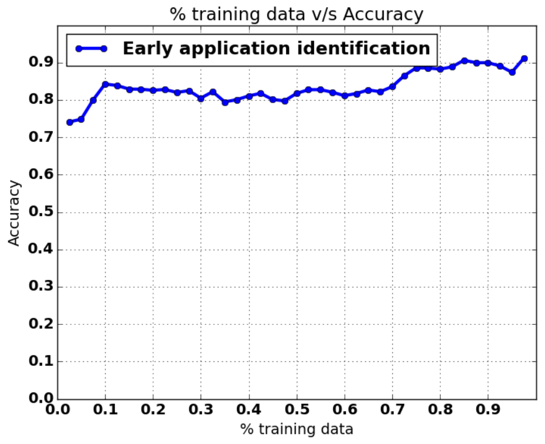
In this section, we present the impact of flow statistics attributes to classification accuracy based on grouping of attributes. We first describe the flow attributes used in this study and attribute groups, and then discuss classification accuracy for individual groups.

From the network traffic data set[1], totally 17 flow attributes are considered, as summarized in Table I. As discussed in the previous section, using only a single attribute of average packet

[1]The description the data set used in this study will be described in Section V in detail.



(a) K-mean clustering [5]



(b) Early identification [2]

Fig. 1: Network traffic classification with previous techniques

TABLE I: Attributes used in out studies

Attribute	Type	Description
Flow size	flow level	Total number of bytes in the flow
Flow duration	flow level	Duration of the flow from initiation to termination
Number of Packets	flow level	Total number of packets inside the flow
Number of packets per second	flow level	Number of packets / Flow duration
Number of bytes Per second	flow level	Total packet size / Flow duration
Payload size	flow level	the payload size in bytes without network and transport layer headers
Packet size	packet level	min/max packet size and average and standard deviation of packet sizes in the flow
Packet inter-arrival time (IAT)	packet level	min/max packet inter-arrival time and average and standard deviation of inter-arrival times in the flow
Relative IAT (RIAT)	packet level	Recomputation of IAT after dividing IAT by round-trip delay: max RIAT and average and standard deviation of RIATs

size works much better than using the entire attributes when using clustering. Thus, it would be beneficial to examine the significance of the attributes in detail. Since there can exist too many combinations of the attributes, we categorize the attributes into four groups based on their characteristics, as follows:

- *Flow group*: Any flow-level attribute belongs to this group, including flow size, flow duration, and so forth.
- *Packet size group*: Attributes related to packet size belong to this group: minimum, maximum, average, and standard deviation of packet sizes
- *Inter-Arrival Time (IAT) group*: Attributes from inter-arrival time: minimum, maximum, average, and standard deviation of IAT
- *Relative Inter-Arrival Time (RIAT) group*: Attributes related to relative inter-arrival time: minimum, maximum, average, and standard deviation of RIAT, which is defined to consider end-to-end latency:

$$\tau'_i = \frac{\tau_i}{\min\limits_{k=1..|f|-1} \tau_k}$$

Here, f is a flow with $|f|$ number of packets, τ_i is the time difference between i-th packet and $(i+1)$-th packet in that flow, and τ'_i is the relative inter-arrival time between the two packets.

To evaluate the attributes' impact to network classification, we applied the simple K-means clustering technique described in the previous section with different combinations of attributes for each group. To measure classification accuracy, we used a simple metric of true positive, and thus, accuracy indicates true positive rate in this paper. By default, we used 90%:10% for the ratio of the number of training flows to the number of test flows. Thus, the results with the default training set ratio would be the approximations of the maximum classification accuracy since using a smaller set of training flows generally leads to worse accuracy in classification, and vice versa. For ease of exposition, we define *training fraction* = $\frac{\text{the number of training flows}}{\text{the number of total flows}}$, and the default is 90%. Note that we

TABLE II: Accuracy of flow group

Flow Size	Flow Duration	Number of Packets	Avg Pkts Per Sec	Avg Bytes Per Sec	Payload Size	Accuracy
✓	✓	×	×	×	×	92.92%
✓	×	✓	×	×	×	91.66%
✓	×	×	✓	×	×	91.12%
✓	×	×	×	×	✓	94.53%
×	✓	×	×	×	✓	92.58%
×	×	✓	×	×	✓	93.80%
✓	✓	×	✓	×	×	92.31%
×	✓	✓	×	×	✓	92.02%
✓	✓	✓	✓	×	×	91.86%

TABLE III: Accuracy of packet size group

Avg Pkt Size	Min Pkt Size	Max Pkt Size	Stddev Pkt Size	Final Accuracy
✓	×	×	×	92.43%
×	✓	×	×	75.95%
×	×	✓	×	92.22%
×	×	×	✓	91.82%
✓	✓	×	×	91.46%
✓	×	✓	×	92.98%
✓	×	×	✓	93.96%
×	✓	✓	×	91.46%
×	✓	×	✓	92.69%
×	×	✓	✓	92.76%
✓	✓	✓	×	93.98%
✓	✓	×	✓	93.62%
✓	×	✓	✓	92.76%
×	✓	✓	✓	91.73%
✓	✓	✓	✓	93.75%

repeated the same experiment at least five time and report the average throughout the paper.

Table II shows accuracy of classification when using a subset of attributes in the flow group. In fact, there can exist many number of combinations with the six attributes in this group, but we report the top 10 results only to save the space. Surprisingly, we can see very high accuracy up to 94.5% with only 2–4 attributes in the flow group.

We next examined the attributes in the packet size group. As shown in Table III, any combination of attributes in the packet size group works very good, showing up to 94% accuracy. When using minimum packet size only, it does not work well with 76% accuracy. The results here explain why it worked well only with the attribute of average packet size in [2]. We observed the combination of *Avg Pkt Size* and *Stddev Pkt Size* gives the best accuracy of the group, and the combination of *Min Pkt Size*, *Max Pkt Size* and *Avg Pkt Size* gives the second best accuracy in this group.

TABLE IV: Accuracy of IAT group

Avg IAT	Min IAT	Max IAT	Stddev IAT	Final Accuracy
✓	×	×	×	47.29%
×	✓	×	×	49.34%
×	×	✓	×	39.20%
×	×	×	✓	42.82%
✓	✓	×	×	45.86%
✓	×	✓	×	44.25%
✓	×	×	✓	47.78%
×	✓	✓	×	39.64%
×	✓	×	✓	42.91%
×	×	✓	✓	40.54%
✓	✓	✓	×	45.20%
✓	✓	×	✓	47.18%
✓	×	✓	✓	42.70%
×	✓	✓	✓	40.10%
✓	✓	✓	✓	42.75%

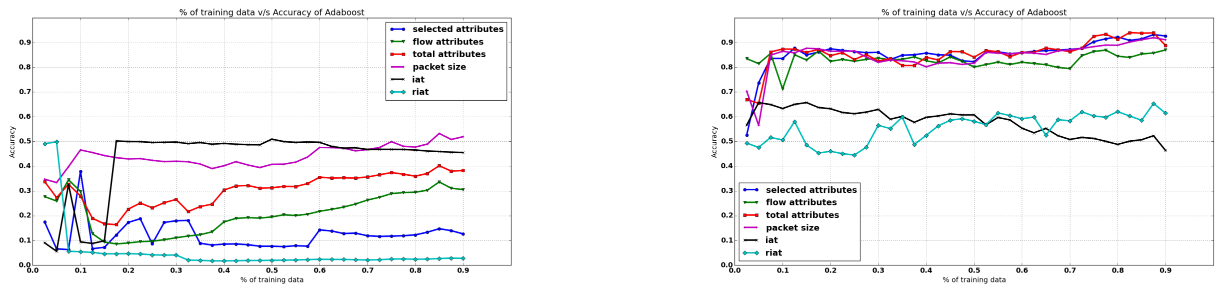
TABLE V: Accuracy of RIAT group

Avg RIAT	Max RIAT	Stddev RIAT	Final Accuracy
✓	×	×	55.86%
×	✓	×	54.08%
×	×	✓	53.41%
✓	✓	×	55.21%
✓	×	✓	54.28%
×	✓	✓	55.25%
✓	✓	✓	56.41%

Unlike the flow and the packet size groups, we observed that the attributes related to packet inter-arrival time are not good with clearly low classification accuracy less than 60%. Hence, these groups are less attractive for network traffic classification than the flow and packet groups.

We also conducted a set of experiments with supervised learning methods (Adaboost and Naive Bayes) with a subset of the attributes to see their impacts. For the experiments we considered the (i) the entire set of attributes, (ii) flow group, (iii) the packet size group, (iv) the IAT group, (v) the RIAT group, and (vi) a combination of selected attributes that showed relatively greater accuracy. The selected attributes include: flow size, payload size, number of packets, and min/max/avg/stdev packet size.

Figure 2(a) and Figure 2(b) show the classification results with Naive Bayes and Adaboost [6], respectively. We observed Naive Bayes works poorly showing less than 60% accuracy, and thus, we do not analyze furthermore. Unlike this, Adaboost works well over 85% accuracy for some combinations of attributes with training fraction $\geq$ 10%. As shown from the figure, statistics related to packet arrival time (i.e., IAT and RIAT) do not work well like clustering. Unlike clustering,



(a) Naive Bayes classifier (b) Adaboost classifier

Fig. 2: Network traffic classification with supervised learning techniques

however, we can see that using the entire attributes works almost the best. We also observed that using the packet size group or the selected attributes group showed outstanding performance compared to the rest of the groups, yielding comparable results with the use of the entire attributes.

In sum, we observed that applying the entire attributes degrades classification accuracy when using clustering-based techniques. For supervised learning techniques, using the entire attributes does not degrade classification accuracy but using only half of the attributes shows the comparable result with Adaboost.

IV. Multiple Cluster Models

As discussed in the previous sections, some attributes may have greater impacts on classification accuracy for our application. While a selected supervised learning technique works well with the entire attributes (Figure 2(a)), a semi-supervised learning we tested did not work well with the entire attributes showing less than 60% accuracy (Figure 1(a)), which is often unacceptable for network traffic classification. However, if we apply a selected set of attributes used in Section III, we observed that both supervised and semi-supervised learning techniques worked quite well, showing around 90% accuracy. Although not shown in Figure 1(a), the classification accuracy with the selected attributes we observed was around 90%, much better than using the entire attributes. More interesting results would be that it is possible to obtain even better accuracy up to 94% using a clustering-based technique with only 2–4 attributes as shown from Table II – Table V, which is better than the case of using the selected seven attributes altogether.

Here, a question may arise that using only a couple of attributes would always yield better results than using many more attributes and if the results would be consistent and reliable when considering a diverse range of network protocols. For example, using only packet size statistics may work well with a certain set of network applications, but it may be degraded with a different set of applications. In this work, we develop a clustering-based technique that utilizes a set of attribute groups for network traffic classification. As we observed in Table II – Table V, several sets of attributes are beneficial to yield relatively high classification accuracy over 90%. Thus, our hypothesis is that utilizing multiple attribute groups would yield more consistent results than considering only a couple of attributes for traffic classification. In this work, we utilize *multiple* training models, each of which is from a different set of attributes, rather than simply merging the attributes together to form a *single* cluster model. In other words, multiple trained models (with different combinations of attributes) are constructed, and classification takes place with the constructed models, as will be described in detail next.

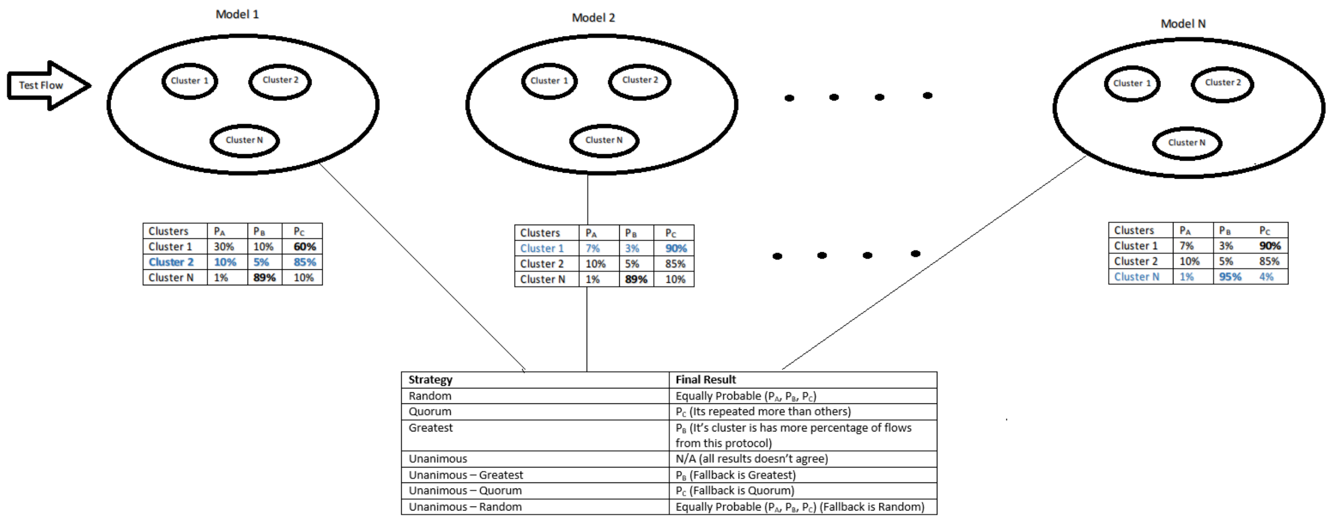
With a single classifier, supervised or unsupervised, it is not possible to use multiple training models since we should put every attribute to be considered through one training model. For example, we can easily have one model based on the packet size group and the IAT group combined together, but we cannot have those two groups separately in one training model. If we should consider multiple training models, the key challenge is how to effectively utilize them to reach the final classification. Our strategy for this question is that we use the population of each protocol in each cluster for each training model, which will be used for making the final decision. In this paper, we define a new term called *population fraction*, which stands for percentage of the flows of a protocol to the total number of flows in that cluster:

$$P_{prot(A)} = \left(\frac{N_{prot(A)}}{N_{total}}\right) \quad (1)$$

Here, $P_{prot(A)}$ is the population fraction for protocol A, $N_{prot(A)}$ is the number of flows belonging to protocol A, and N_{total} is the total number of flows residing in the cluster.

Figure 3 shows the multi-cluster models we propose in this paper. There can be a set of training models more than one, each of which maintains population fraction information for each protocol. For instance, *Model 1* can be constructed with attributes of average and standard deviation of packet sizes, while *Model 2* can be from the number of packets and payload size attributes, and so forth. When constructing individual models, it is straightforward to calculate population fraction for each protocol for every cluster.

We next describe how we can make the final classification decision from multiple models for a given input flow that



Clusters	P_A	P_B	P_C
Cluster 1	30%	10%	**60%**
Cluster 2	**10%**	**5%**	**85%**
Cluster N	1%	**89%**	10%

Clusters	P_A	P_B	P_C
Cluster 1	7%	3%	**90%**
Cluster 2	10%	5%	85%
Cluster N	1%	**89%**	10%

Clusters	P_A	P_B	P_C
Cluster 1	7%	3%	**90%**
Cluster 2	10%	5%	85%
Cluster N	1%	**95%**	4%

Strategy	**Final Result**
Random	Equally Probable (P_A, P_B, P_C)
Quorum	P_C (Its repeated more than others)
Greatest	P_B (It's cluster is has more percentage of flows from this protocol)
Unanimous	N/A (all results doesn't agree)
Unanimous – Greatest	P_B (Fallback is Greatest)
Unanimous – Quorum	P_C (Fallback is Quorum)
Unanimous – Random	Equally Probable (P_A, P_B, P_C) (Fallback is Random)

Fig. 3: Multi-Cluster models represented graphically.

needs to be classified. From each model, it is not difficult to identify the winner that has the greatest population fraction among a set of protocols. For example, in Figure 3, Cluster 2 is selected for the test flow and we can see that P_c (population fraction for protocol C) is the most dominant in *Model* 1. It is straightforward to make the final classification decision if all the trained models produce the identical classification result. However, there is no guarantee for this, and it is necessary to coordinate non-identical results to reach the final decision.

Let us define F_{M_i} as the population fraction of the dominant protocol in *Model i*, and D_{M_i} as the name of the dominant protocol in the model. Then, we get $D_{M_1} = P_c$ and $F_{M_1} = 0.8$ from Figure 3, for example.

We additionally define the following two sets to represent all the outcomes from all the N models:

$D = \{D_{M_i} | 1 \leq i \leq N\}$, and
$F = \{F_{M_i} | 1 \leq i \leq N\}$

We then make use of the results from the multiple models in determining the final classification result of the incoming flow. To do this, we employ a set of different selection methods that chooses a model k based on their own criteria function, and we accept D_{M_k} as the final decision. If the actual protocol of the test flow is P_t, then the classification result is:

$$D_{M_k} = \begin{cases} P_t, & \text{correct classification} \\ \text{otherwise}, & \text{incorrect classification} \end{cases}$$

We tested the following set of different strategies to select k:

- *Random*: Select k randomly, and hence, $k = rand(N) + 1$.
- *Greatest*: Select the greatest population fraction, that is, $k = \arg\max_i \{F_{M_i} | 1 \leq i \leq N\}$.
- *Quorum*: Select the majority result from the trained models. In case of a tie, it falls back to *Random*.
- *Unanimous-Random*: Select the final classification only when all the results from the trained models are exactly the same (*Unanimous*); otherwise it falls back to *Random*.
- *Unanimous-Greatest*: It tries *Unanimous* first and falls back to *Greatest* if *Unanimous* has no result.
- *Unanimous-Quorum*: It tries *Unanimous* first and falls back to *Quorum* if *Unanimous* does not make a final decision.

V. Evaluation

In this section, we evaluate the proposed multi-cluster models with the selection functions to orchestrate the classification results from individual models. We first provide a short description of the data set we used for our evaluation, then present the experimental results.

A. Experimental Setup

The network traffic data had been collected with full payload information in early 2014 over several months. It has been gathered on various interfaces 1) Wired, 2) WiFi, and 3) 3G and LTE. Each data set for individual applications were collected in isolation, by generating requests intentionally and capturing bidirectional data. To construct a flow base, we considered five tuples (i.e., source IP, destination IP, source port, destination port, and protocol), and used TCP flags to mark the start and end of a flow. For TCP traffic, flows started before the capture or flows terminating after the capture were not considered in construction of flow information. For UDP traffic, a fixed amount of idle time (60 seconds) was used as the indication of connection termination, as there is no connection management information for UDP flows. A python library *scapy*[3] was used to read from *pcap* files. We then

TABLE VI: Protocols used for the experiments

Protocol	TCP/UDP	Number of flows
http	TCP	
bittorrent	TCP	
edonkey	TCP	
gnutella	TCP	
skype	UDP	

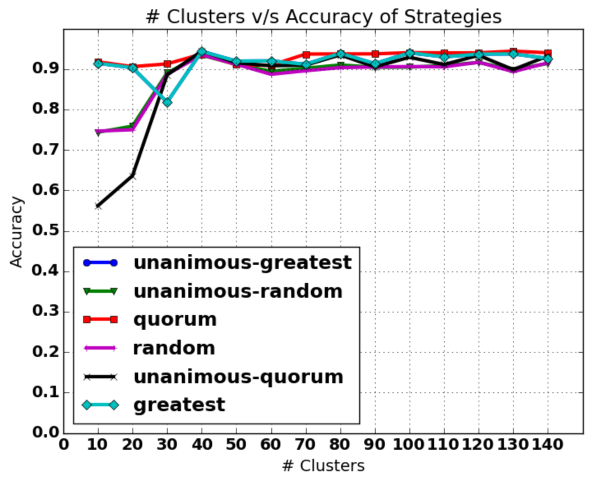


Fig. 4: Comparison of the selection methods over different numbers of clusters

performed a refining process using community maintained signatures [9] available from L7 filters. We matched payload of each constructed flow with corresponding signatures from the L7 filters; any unmatched flow was discarded. We selected five protocols (http, bittorrent, http, edonkey, gnutella, and skype) that have the greatest number of flows from the flow base, as shown in Table VI.

We divide the flow data into training and testing partitions; training data to train the model and testing data to test actual classification using the trained model. The default ratio used is 90%:10% (i.e., training fraction = 90%), but we also explored a variety of ratios as will be presented. In addition, we set the default number of clusters in each trained model to 40, based on our preliminary experiments that showed less than 40 may result in unstable results. To take variations into account, the same experiment was repeated at least five times and we report the average. As mentioned, we used a simple metric of true positive rate for measuring classification accuracy.

B. Experimental Results

We now report our observations obtained from our experiments. We first evaluate the classification performance of the selection methods with a fixed number of trained models (4 models), and then examine the impact of the number of trained models. We finally compare our proposed method with existing techniques.

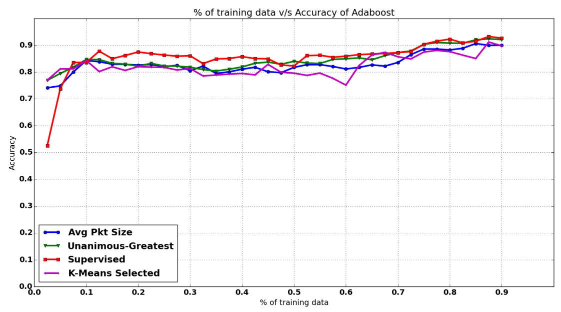


Fig. 6: Percent of training data v/s Accuracy Comparing all the existing and proposed technique

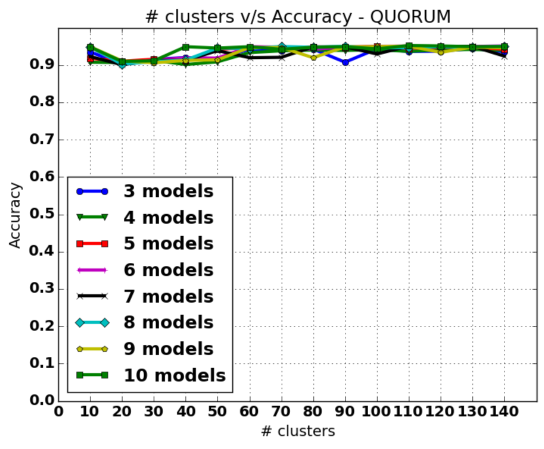
Figure 4 compares the selection methods over different numbers of clusters from 10 to 140 clusters in each training model. For this experiment, we constructed four training models that showed the best results in Table II – Table V; this is, (i) flow size and payload size, (ii) the number of packets and payload size, (iii) avg/min/max packet size, and (iv) avg/stdev packet size, for each trained model. As can be seen from the figure, *Quorum* shows quite stable results over the number of clusters, yielding over 90% accuracy for any number of clusters. *Greatest* also shows good results but it is degraded when the number of clusters = 30, which indicates a bit more sensitive to this parameter. *Unanimous-Quorum* yields fairly stable results with 40 or greater number of clusters. As expected, the *Random* strategy is worse than the others overall.

We next explore with different numbers of trained models, varying from three to ten models. As mentioned, the models were constructed based on the results from Table II – Table V. Since *Quorum* showed fairly consistent results in Figure 4, we selected the *Quorum* strategy for this experiment. From Figure 5, there is no clear distinction with different numbers of trained models. One observation is that using many more models yields slightly more stable results. Thus, the results here indicate a potential trade-off between the training/testing complexity and classification accuracy.

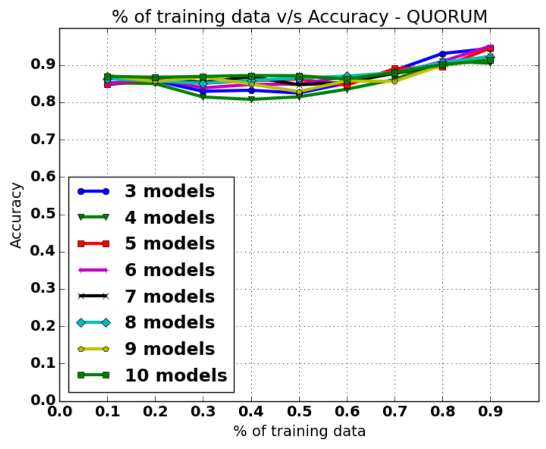
We finally compare our proposed technique with some of existing techniques. In Figure 6, "Avg Pkt Size" is from the early identification method [2], "K-means Selected" from [5] with the best combination of attributes, "Supervised" with Adaboost with the selected attributes, and "Proposed" is based on our multi-cluster models with the *Quorum* selection strategy. As shown from the figure, our technique works better than "Avg Pkt Size" and "K-means Selected" that are based on clustering. The supervised technique shows better results than the others overall, but our technique also showed almost comparable results with training fraction $\geq$ 50%, which indicates that the semi-supervised learning would be an comparable alternative to the supervised method.

VI. Conclusion

The importance of network traffic classification has increased for local and ISP networks for a variety of purposes



(a) Impact of the number of clusters



(b) Impact of the training:testing flows ratio

Fig. 5: Impact of the number of trained models

including quality of service and security. Using a semi-supervised learning with clustering is one branch of research directions for traffic classification, and several studies have been presented with interesting results. However, we observed fairly less classification accuracy with our own data set than what have been reported in such previous studies (91% at best).

In this work, we presented a new clustering-based technique that utilizes multiple trained models to improve overall classification performance. We first examined the impact of combinations of attributes to the classification performance and constructed multiple trained models, each of which is based on a combination of superior attributes. To incorporate results from multiple models, we also set up a set of selection strategies and examined their performance with respect to classification accuracy. From a set of experiments, we showed that our technique consistently outperforms the existing clustering-based methods and is fairly comparable to the supervised learning technique with Adaboost, yielding up to 95% classification accuracy. We also observed that the *Quorum* selection strategy works consistently compared to the other strategies and using many more models works better with respect to stability, over different experimental settings.

Acknowledgment

This work was in part supported by the IT R&D program of MOTIE/KEIT [10041548, "240Gbps realtime automatic signature generation system for application traffic classification supporting over 95% completeness and accuracy"]

References

[1] "The BitTorrent Protocol Specification." [Online]. Available: http://www.bittorrent.org/beps/bep_0003.html

[2] L. Bernaille, R. Teixeira, and K. Salamatian, "Early application identification," in *Proceedings of the 2006 ACM CoNEXT Conference*, ser. CoNEXT '06, 2006, pp. 6:1–6:12.

[3] P. Biondi, "Scapy: Scapy is a packet manipulation tool for computer networks," 2011–. [Online]. Available: http://www.secdev.org/projects/scapy/

[4] T. En Najjary, G. Urvoy Keller, and M. Pietrzyk, "Application-based feature selection for internet traffic classification," in *ITC 2010, 22nd International Teletraffic Congress, September 7-9, 2010, Amsterdam, The Netherlands*, Amsterdam, NETHERLANDS, 09 2010. [Online]. Available: http://www.eurecom.fr/publication/3174

[5] J. Erman, M. Arlitt, and A. Mahanti, "Traffic classification using clustering algorithms," in *Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data*, ser. MineNet '06, 2006, pp. 281–286.

[6] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational learning theory*. Springer, 1995, pp. 23–37.

[7] L. Grimaudo, M. Mellia, and E. Baralis, "Hierarchical learning for fine grained internet traffic classification," in *8th International Wireless Communications and Mobile Computing Conference, IWCMC 2012, Limassol, Cyprus, August 27-31, 2012*, 2012, pp. 463–468.

[8] P. Haffner, S. Sen, O. Spatscheck, and D. Wang, "Acas: automated construction of application signatures," in *Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data*, ser. MineNet '05, 2005, pp. 197–202.

[9] E. S. Justin Levandoski, A. E. Matthew Strait, with support from Sebastian Celis, and L. Kittredge, "L7 signatures for various protocols." [Online]. Available: http://l7-filter.sourceforge.net/protocols

[10] S. Ohzahata, Y. Hagiwara, M. Terada, and K. Kawashima, "A traffic identification method and evaluations for a pure p2p application," in *PAM*, 2005, pp. 55–68.

[11] B. Park, Y. J. Won, M. Kim, and J. W. Hong, "Towards automated application signature generation for traffic identification," in *IEEE/IFIP Network Operations and Management Symposium: Pervasive Management for Ubioquitous Networks and Services, NOMS 2008, 7-11 April 2008, Salvador, Bahia, Brazil*, 2008, pp. 160–167.

[12] G. Xie, M. Iliofotou, R. Keralapura, M. Faloutsos, and A. Nucci, "Subflow: Towards practical flow-level traffic classification," in *Proceedings of the IEEE INFOCOM 2012, Orlando, FL, USA, March 25-30, 2012*, 2012, pp. 2541–2545.

[13] M. Ye, K. Xu, J. Wu, and H. Po, "Autosig-automatically generating signatures for applications." in *CIT (2)*. IEEE Computer Society, 2009, pp. 104–109.