2018

High Performance Attack Estimation in Large-Scale Network Flows

Christopher B. Freas
Georgia State University

Robert W. Harrison
Georgia State University

Yuan Long
Georgia State University

High Performance Attack Estimation in Large-Scale Network Flows

Christopher B. Freas
Department of Computer Science
Georgia State University
Atlanta, USA
cfreas@cs.gsu.edu

Robert W. Harrison
Department of Computer Science
Georgia State University
Atlanta, USA
rharrison@cs.gsu.edu

Yuan Long
Department of Computer Science
Georgia State University
Atlanta, USA
ylong4@gsu.edu

***Abstract*—Network based attacks are the major threat to security on the Internet. The volume of traffic and the high variability of the attacks place threat detection squarely in the domain of big data. Conventional approaches are mostly based on signatures. While these are relatively inexpensive computationally, they are inflexible and insensitive to small variations in the attack vector. Therefore we explored the use of machine learning techniques on real flow data. We found that benign traffic could be identified with high accuracy.**

***Index Terms*—Networks, Flow analysis, Attack Detection, Machine Learning**

I. Introduction

Big data problems are characterized by five Vs: Volume, Velocity, Variety, Veracity, and Value [1]. Reliable estimation of the threat from a network attack requires rapid (Velocity) and accurate (Veracity, Value) estimation of a non-homogeneous threat (Variety) in the presence of terabytes of data (Volume).

Forecasts on both the Volume and Velocity of network traffic show a doubling in the next three years [2]. Estimating a variety of attacks at increasing scale is critical to network security.

Many toolkits exist to aid in launching attacks. Examples include network scanners, botnets, and data injection tools, to name a few. When coupled with certain protocols and services, attacks become particularly effective. Attackers rely on the connectionless user datagram protocol (UDP) to hide themselves. Such attacks exploit UDP by spoofing the packet's source Internet protocol (IP) address. Modern attacks use the simple network management protocol (SNMP) or the network time protocol (NTP). When queried, these protocols return a response payload larger than the query size. Thus, attacks are "amplified" to devastating effect. A recent attack utilizing the Memcached service crossed the terabit per second scale [3].

In this paper, we explore the use of conventional machine learning approaches on experimental network flow data in order to determine baseline performance as the first step to developing better approaches. In particular, we analyze the Intrusion Detection Evaluation Dataset published by Sharafaldin et al in [4]. This dataset consists of 15 different attacks over a week of collection. The dataset also includes benign background traffic which mimics typical user behavior. The Intrusion Detection Evaluation Dataset is hereafter referred to as the CIC-IDS dataset.

We also examined the KDD '99 network intrusion dataset [5]. This dataset is often used for machine learning applications. The KDD and CIC-IDS dataset differ in several important ways:

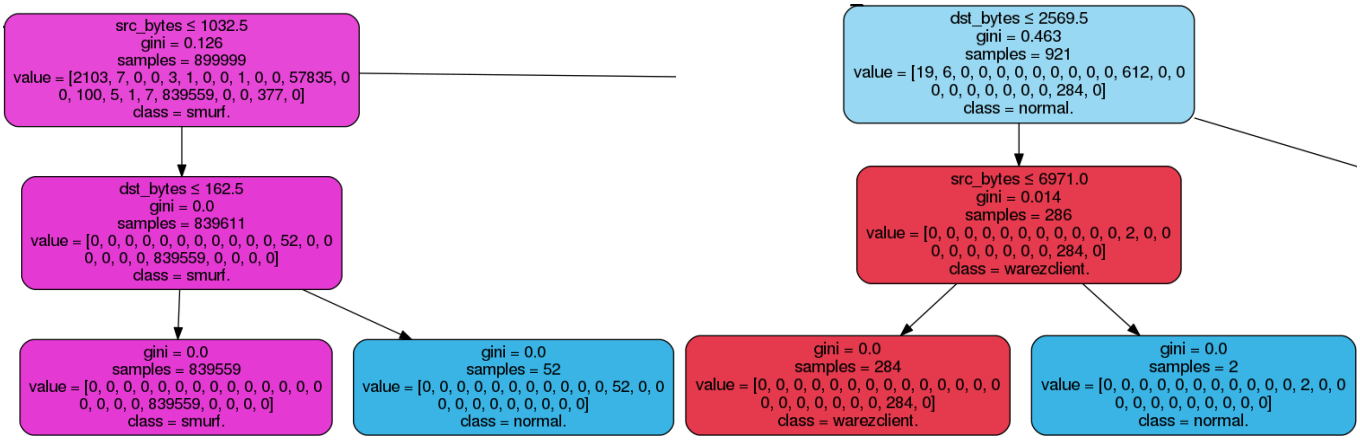
- The KDD data consists of *connections* between endpoints and not whole flows
- Much of the KDD data has connections with a duration of zero
- The KDD data has no source or destination Internet Protocol (IP) addresses
- The KDD data has no source ports
- The KDD data has destination ports, but they are the nominal service names instead of numeric values

The most glaring difference between the datasets is that the KDD dataset is much older. There is a well known truism that "Attacks always get better; they never get worse." [6] Modern attacks and attack methods have evolved since the publishing of the KDD dataset. It is possible that the KDD dataset may no longer represent common attacks. We include this dataset to show how well machine learning methods work for attack classification in disparate datasets.

The contributions of this paper are as follows. After showing that conventional machine learning algorithms converge to high accuracy on both datasets, we break the CIC-IDS data into non-overlapping sets based on the time of collection. The accuracy significantly deteriorates when attacks are present in the test data but not the training data which demonstrates that the system is non-stationary. Since conventional machine learning algorithms converge poorly on this non-stationary dataset, we conclude that generalizing on network flow data requires more advanced machine learning algorithms.

II. Related Work

Existing research into network intrusion detection falls into two categories. The first is misuse detection and the second is anomaly detection [7]. Misuse detection uses an attack signature database to recognize a potential attack. Anomaly



(a) A sample set of decisions that discriminate between normal and smurf traffic.

(b) A sample set of decisions that discriminate between normal and warezclient traffic.

Fig. 1: Samples from the decision tree trained with the feature-reduced KDD data set.

detection relies on a model of normal system behavior to detect an attack. Both techniques have been well researched and each has its own disadvantages. Misuse detection cannot detect new attacks since it relies on known attack signatures. Anomaly detection does not rely on attack signatures. Instead, a profile of "normal" behavior must be built to detect anomalous behavior. Building such a profile is difficult.

Several previous approaches use machine learning models for misuse detection and anomaly detection. Accuracy and speed improvements in anomaly detection have come from supervised learning methods such as k-Nearest Neighbor (kNN) [8], Support Vector Machines (SVM) [9] and decision trees [10]. Unsupervised learning methods such as K-means [11] have improved malicious behavior detection. If the training and testing datasets come from the same unknown distribution, supervised methods in general outperform unsupervised methods [12]. Decision trees can yield accuracy as high as 95% [12]. This justifies our use of decision trees on the CIC-IDS dataset in this paper.

Hybrid intrusion detection combines misuse and anomaly detection. It has been suggested as a way to resolve the disadvantages in traditional intrusion detection techniques. In [13], a decision tree is used to decompose the data into subsets. Support vector machines (SVMs) are trained on the subsets. Training and testing times are substantially reduced and high accuracy is obtained. In [14], Rai et el. modified the C4.5 decision tree algorithm to perform more granular feature splitting. The splitting function is augmented to average the values for each feature. The information gain ratio is then the information gain divided by the result of the splitting function. The result is that any bias towards frequent values in attributes is removed.

Dimensionality reduction is crucial to applying machine learning to intrusion detection. In [15], Zhang et el. use a Bayesian Network classifier to iteratively arrive at an optimal feature set. Their technique uses the wrapper approach to feature selection. In the wrapper approach, feature selection is based on classification accuracy improvements. Another dimensionality reduction technique is the filter approach. The filter approach evaluates and ranks each feature independent of the classifier. The result is a feature subset that best approximates the original dataset.

In [16], Mukherjee, et el. make use of Correlation-based Feature Selection (CFS), Information Gain (IG), and Gain Ratio (GR) for feature selection. The authors compare these techniques to their proposed Feature-Vitality Based Reduction Method (FVBRM). FVBRM functions like the Bayesian Network described in [15] with the addition of feature vitality. Feature vitality includes classification accuracy, true-positive rate (TPR), and false-positive rate (FPR). Feature vitality thus defines the subset of features used in the reduced dataset.

III. Using Decision Trees to Classify Attacks

Decision trees are a machine learning model used for classification and regression tasks. Decision trees work by creating rules for splitting nodes based on the features in the data. These rules are analogous to asking a series of yes/no questions on the data. The predicted class of the input data is that of the leaf node once reached.

Several algorithms exist for constructing a decision tree (e.g. ID3, C4.5, CART, etc...). In this paper we use the CART algorithm [17]. CART runs in logarithmic time and uses Gini impurity on features to split nodes. Gini impurity is the probability of obtaining two different output predictions for a given input. The Gini impurity of node t for $j = \{1, \cdots, k\}$ possible classes is given by Equation 1:

$$1 - \sum_{j=1}^{k} p^2(j|t) \tag{1}$$

The tree generated by the decision tree algorithm is easy to understand (for example Figures 1a and 1b). This simplicity makes decision trees very popular for many machine learning tasks. Effective attack classification requires that the machine

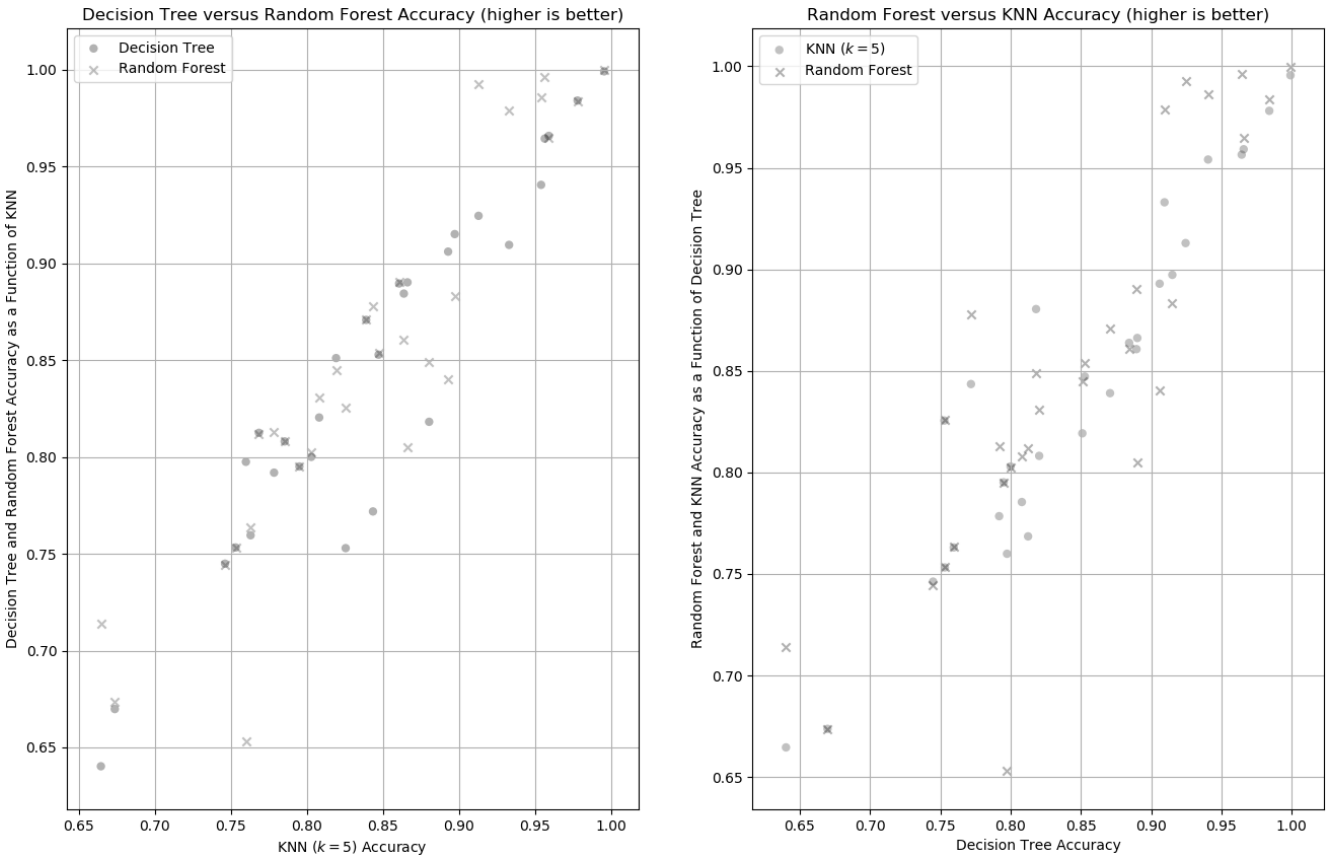


Fig. 2: Performance comparison for the different models. Random forest performs the best, but only by a small margin.

learning model generalize on the data. We explore how well decision trees generalize by measuring accuracy on an expanding dataset.

IV. Feature Reduction

Raw network data are quite verbose. Reducing the data to relevant features is critical for obtaining high classification accuracy. This process comprises feature reduction techniques. The idea is to drop features which do not provide adequate variance in the data. The remaining features then provide enough variance to achieve reasonable classification accuracy.

There are benefits and drawbacks to feature reduction. Benefits include simpler models, improved accuracy, and a reduction of the effects of high dimension data. High dimension data suffers from the so-called "curse of dimensionality" [18]. This problem affects machine learning when the data lacks enough samples for combinations of all the features. Removing unnecessary features eases the effects of an insufficient amount of samples. Feature reduction can also enable the use of other machine learning models. Removing unnecessary features can remove the noise from data, but can remove the signal as well. This can cause a close grouping of the data, destroying the variance and reducing accuracy.

Reducing the size of the dataset is a practical advantage of feature reduction. Network flow data can easily exceed gigabytes per day, and threat detection requires rapid and accurate response to be useful. Therefore, minimizing the size of the data has important practical effects.

We applied Principal Components Analysis (PCA) [19] and the entropy and information gain metrics from the ID3 algorithm [20] to both datasets and were able to significantly reduce the number of features while retaining the same level of accuracy. Applying PCA can obscure the connection between the reduced dataset and the original. PCA can also "load" the first principal component with the highest variance which makes the reduced dataset non-optimal. The remaining com-

	CIC-IDS				KDD			
Split (train % / test %)	Accuracy/Std Dev	Precision	Recall	F1 Score	Accuracy/Std Dev	Precision	Recall	F1 Score
67/33	97.881/0.010	97.881	97.881	97.881	99.982/0.0004	99.982	99.982	99.982
50/50	97.837/0.033	97.837	97.837	97.837	99.981/0.001	99.981	99.981	99.981
33/66	97.759/0.026	97.759	97.759	97.759	99.973/0.008	99.973	99.973	99.973

TABLE I: Cross-validated performance on the entire CIC-IDS and KDD datasets. The IP addresses are excluded from the CIC-IDS dataset. All scores are percentages.

	CIC-IDS		KDD	
Split (train % / test %)	Train Time/Std Dev	Test Time/Std Dev	Train Time/Std Dev	Test Time/Std Dev
67/33	25.259/2.651	0.093/0.002	10.439/0.797	0.175/0.029
50/50	16.176/2.234	0.067/0.002	6.712/1.264	0.120/0.0006
33/66	10.422/1.783	0.042/0.007	3.780/0.934	0.0869/0.0134

TABLE II: Computational performance on the entire CIC-IDS and KDD datasets. The IP addresses are excluded from the CIC-IDS dataset. All times are in seconds.

ponents thus no longer provide an accurate interpretation of the original data. Measuring information gain is useful for working around these limitations. In doing so, we can arrive at a set of features that are the most characteristic of attacks.

V. Results

We applied machine learning to both the CIC-IDS data and the KDD data and found excellent cross-validated accuracy when we looked at the entire data set. When the data set was segmented by time to study the effect of novel attacks the results were less impressive. We also looked at the effects of removing various features from the data to determine the minimal set of features needed to achieve high accuracy.

A. Accuracy of Machine Learning

Table I shows the performance of the decision tree for both datasets. We removed the IP addresses from the CIC-IDS data for two important reasons. First, attackers spoof IP addresses to hide themselves and defeat IP filtering systems. Second, the high variance of the IP addresses "crowded out" the other attributes when applying feature reduction. This complicates efforts to find a set of features that best characterize attacks. We applied k-fold cross-validation with $k = 3$ in all cases. The accuracy scores shown are averages taken over the folds. Table II shows the running time of the decision tree. The running times shown are averages taken over the folds.

Though the results in Table 1 and Table 2 are appealing, they obscure a problem we found with the data. Because the amount and type of traffic varies by day (for the CIC-IDS data) and by connection (for the KDD data) the data are not stationary. This lack of stationarity means that generalizing on the data is difficult. Traditional machine learning algorithms such as decision trees, support vector machines, and so on will not work without modifications because they expect the underlying data to have a stationary distribution.

To test this, we first trained a decision tree on the CIC-IDS Monday dataset. We then checked its accuracy on the rest of the week. We obtained scores of 96.897, 63.524, 99.517, and 71.172 for Tuesday, Wednesday, Thursday, and Friday, respectively. We tested retrospective learning by training on Friday's data. We then checked accuracy on Thursday, Wednesday, Tuesday, and Monday and obtained scores of 95.264, 64.117, 96.811, 99.714, respectively. We obtained similar results when we tested using a random forest with 10 estimators.

We further tested decision tree and random forest generalization on a power set of the days of the CIC-IDS data. The decision tree result is shown in Table IV. The random forest result is shown in Table VI. The average score is 84.407% and the standard deviation is 8.530% for the decision tree. The average score is 85.289% and the standard deviation is 9.541% for the random forest.

We also tested classification using the kNN algorithm. Figure 2 provides a comparison between the three models we tested. The left side of the figure shows how the decision tree and random forest compare as a function of kNN. The right side of the figure shows how random forest and kNN compare as a function of the decision tree. Random forest showed better generalization than the decision tree and kNN. The decision tree performed better than random forest. kNN had the worst run time at over 660 seconds on average. This run time is over 9 times longer than the decision tree, which was almost 4 times longer than random forest.

B. Feature Reduction

Table III shows the results of applying the ID3 entropy and information gain metrics to varying sample sizes for both datasets. The KDD data does not show an interesting result since the initial feature chosen is the same for all three sample sizes. Both algorithms select the same initial feature for the first split. The CIC-IDS result varies as the sample size increases until the initial feature is ultimately the same for all datasets.

Figure 3 shows the per-feature variance for all five days of the CIC-IDS data. All features have been scaled to the range [0,1] so that the largest value of each feature is scaled to unit size. It is obvious that many features have a small variance

Dataset	1,000 Samples	5,000 Samples	10,000 Samples
CIC-IDS Monday	Flow Bytes/second	Flow Bytes/second	Flow Bytes/second
CIC-IDS Tuesday	Flow Packets/second	Source Port	Flow Bytes/second
CIC-IDS Wednesday	Source Port	Source Port	Flow Bytes/second
CIC-IDS Thursday	Flow Bytes/second	Flow Bytes/second	Flow Bytes/second
CIC-IDS Friday	Source Port	Source Port	Flow Bytes/second
CIC-IDS Full dataset	Fwd Packets/second	Flow Bytes/second	Flow Bytes/second
KDD Full dataset	Destination Bytes	Destination Bytes	Destination Bytes

TABLE III: Feature reduction on the CIC-IDS and KDD datasets using the ID3 entropy and information gain metrics. The given feature for each dataset is the initial feature to split on.

and can thus be eliminated. Further feature reduction could be based on a variance threshold or some other criteria.

We applied PCA to the CIC-IDS data and reduced the number of features from 85 to 5. The largest eigenvalue corresponded with the Backward Inter Arrival Time (IAT) Total feature. The second largest was Flow Bytes per second. The third largest was Flow Duration. The fourth largest was Forward IAT Total. The last was Forward IAT Max. We reduced the KDD dataset from 41 features to 2. The largest eigenvalues corresponded with the Source Bytes and Destination Bytes features. It is worth noting that the ID3 splitting algorithm always chose Destination Bytes as the best feature to split on in Table III.

We retrained the decision tree on both reduced datasets. Table V shows the results in a format comparable to Table I. The accuracy on the CIC-IDS dataset increased by an average of 0.097% while the accuracy on the KDD dataset decreased by an average of 2.79%.

We then tested generalization on the reduced per-day CIC-IDS dataset for the decision tree and random forest. The rightmost column in Table IV shows the decision tree results. Accuracy decreased by an average of 0.72% with a standard deviation of 9.541%. The rightmost column in Table VI shows the random forest results. Accuracy decreased once again, this time by an average of 0.74% and a standard deviation of 8.728%.

VI. Conclusions

We showed that decision trees provide good performance in estimating attacks. The lack of stationarity in the data means that traditional machine learning models will not generalize well. This presents a problem when an attack signal is present only in certain features. When using a decision tree, the result is that features near the top of the tree for one day are lower down for other days. There are two possible approaches to fixing this. The first is to increase the size of the training set

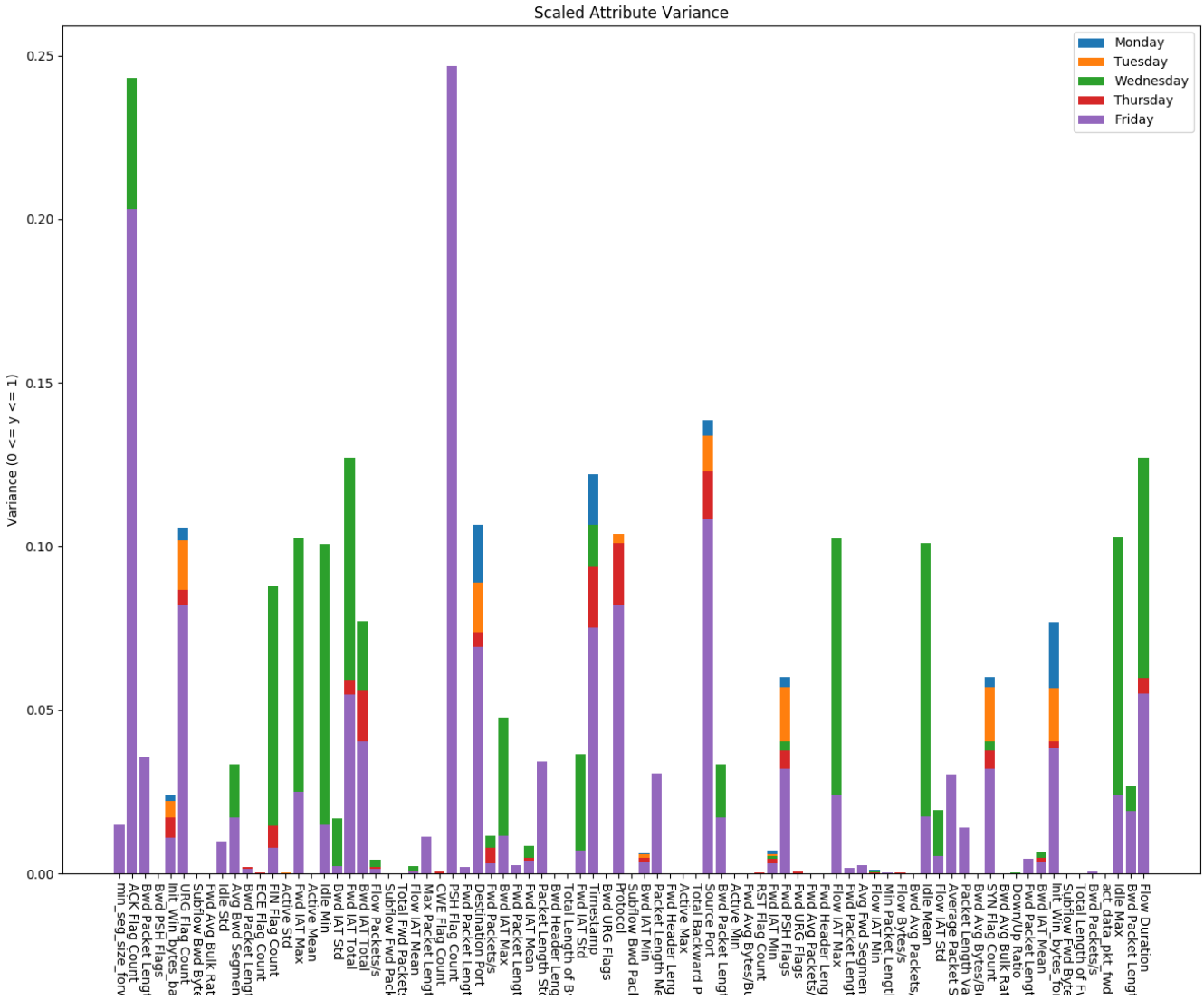


Fig. 3: The variance of all flow features. All values have been scaled to the range $[0, 1]$.

		Full Dataset	Reduced Dataset
Training Days	Test Days	Score	Score
Tuesday	Wednesday, Friday, Monday, Thursday	80.877	78.008
Wednesday	Tuesday, Friday, Monday, Thursday	88.927	85.649
Friday	Tuesday, Wednesday, Monday, Thursday	87.455	90.464
Monday	Tuesday, Wednesday, Friday, Thursday	79.509	79.509
Thursday	Tuesday, Wednesday, Friday, Monday	80.317	80.163
Tuesday, Wednesday	Friday, Monday, Thursday	85.924	85.383
Tuesday, Friday	Wednesday, Monday, Thursday	83.097	88.793
Tuesday, Monday	Wednesday, Friday, Thursday	75.356	75.082
Tuesday, Thursday	Wednesday, Friday, Monday	83.761	76.353
Wednesday, Friday	Tuesday, Monday, Thursday	97.38	94.021
Wednesday, Monday	Tuesday, Friday, Thursday	85.357	82.477
Wednesday, Thursday	Tuesday, Friday, Monday	85.525	84.445
Friday, Monday	Tuesday, Wednesday, Thursday	86.78	84.854
Friday, Thursday	Tuesday, Wednesday, Monday	84.747	87.733
Monday, Thursday	Tuesday, Wednesday, Friday	74.943	74.493
Tuesday, Wednesday, Friday	Monday, Thursday	96.635	95.410
Tuesday, Wednesday, Monday	Friday, Thursday	81.212	78.537
Tuesday, Wednesday, Thursday	Friday, Monday	81.822	81.738
Tuesday, Friday, Monday	Wednesday, Thursday	75.948	79.835
Tuesday, Friday, Thursday	Wednesday, Monday	79.997	84.532
Tuesday, Monday, Thursday	Wednesday, Friday	67.535	67.405
Wednesday, Friday, Monday	Tuesday, Thursday	96.674	93.566
Wednesday, Friday, Thursday	Tuesday, Monday	98.197	96.038
Wednesday, Monday, Thursday	Tuesday, Friday	79.075	78.537
Friday, Monday, Thursday	Tuesday, Wednesday	77.142	80.640
Tuesday, Wednesday, Friday, Monday	Thursday	92.343	92.620
Tuesday, Wednesday, Friday, Thursday	Monday	99.958	98.669
Tuesday, Wednesday, Monday, Thursday	Friday	68.913	69.368
Tuesday, Friday, Monday, Thursday	Wednesday	80.21	72.858
Wednesday, Friday, Monday, Thursday	Tuesday	96.613	93.392

TABLE IV: Generalization on the power set of per-day full and feature-reduced CIC-IDS datasets using the decision tree. IP addresses are excluded from the full dataset. All scores are percentages.

to more than a day and retrain the model after some time. This way, the model is aware of malicious traffic and the structure of the tree can adapt. We observed this lack of awareness when the accuracy on the model trained on the Monday data dropped from 96.897% for Tuesday to 63.524% for Wednesday. The cause of this was the fact that Monday's data were all benign and no traffic for Wednesday was benign. We intend to explore ways of determining the optimal size of this training set. The second is to explore models that do generalize well. Based on our previous work in [21], the Restricted Boltzmann Machine (RBM) is a good choice.

Feature reduction provides a powerful means of reducing the data to the features that are most characteristic of attacks. 85 features were reduced to 5 significant ones with this data, which suggests that the data are highly redundant. This process must be applied iteratively to the data. Doing so systematically eliminates high-variance features that do not contribute to the discrimination between benign and malicious traffic.

The results shown in Table V suggest two possible findings related to our feature reduction efforts. First, the reduced CIC-IDS dataset features are close to those most characteristic of attacks. Second, the reduction in features for the KDD dataset was too aggressive. It is obvious that both reduced datasets need further feature engineering. For this paper, our goal was to show the effects of feature reduction. We intend to explore ways of adding useful features without adding redundancy and retaining the performance improvement that comes with feature reduction.

The overall decrease in accuracy shown in Tables IV and VI compared to the increase for cross-validated training on the entire data set (shown in Table V) strongly suggests that the non-stationarity affects the accuracy of the results. The changes of the accuracy in the power set data between the reduced and full data sets are not uniform, sometimes increas-

	CIC-IDS				KDD			
Split (train % / test %)	Accuracy/Std Dev	Precision	Recall	F1 Score	Accuracy/Std Dev	Precision	Recall	F1 Score
67/33	98.005/0.020	98.005	98.005	98.005	97.176/0.044	97.176	97.176	97.176
50/50	97.929/0.014	97.929	97.929	97.929	97.182/0.031	97.182	97.182	97.182
33/66	97.828/0.032	97.828	97.828	97.828	97.203/0.048	97.203	97.203	97.203

TABLE V: Cross-validated performance on the feature reduced CIC-IDS and KDD datasets. The features used in the CIC-IDS dataset are Bwd IAT Total, Flow Bytes/sec, Flow Duration, Fwd IAT Total, and Fwd IAT Max. The features used in the KDD dataset are Source Bytes and Destination Bytes. All scores are percentages.

		Full Dataset	Reduced Dataset
Training Days	Test Days	Score	Score
Tuesday	Wednesday, Friday, Monday, Thursday	80.820	77.797
Wednesday	Tuesday, Friday, Monday, Thursday	88.279	86.399
Friday	Tuesday, Wednesday, Monday, Thursday	89.428	90.706
Monday	Tuesday, Wednesday, Friday, Thursday	79.509	79.509
Thursday	Tuesday, Wednesday, Friday, Monday	80.290	80.197
Tuesday, Wednesday	Friday, Monday, Thursday	86.265	86.010
Tuesday, Friday	Wednesday, Monday, Thursday	88.634	89.313
Tuesday, Monday	Wednesday, Friday, Thursday	75.341	75.343
Tuesday, Thursday	Wednesday, Friday, Monday	76.382	76.361
Wednesday, Friday	Tuesday, Monday, Thursday	98.760	95.674
Wednesday, Monday	Tuesday, Friday, Thursday	84.551	83.806
Wednesday, Thursday	Tuesday, Friday, Monday	87.385	84.629
Friday, Monday	Tuesday, Wednesday, Thursday	87.852	87.679
Friday, Thursday	Tuesday, Wednesday, Monday	92.390	88.274
Monday, Thursday	Tuesday, Wednesday, Friday	74.098	74.512
Tuesday, Wednesday, Friday	Monday, Thursday	99.686	98.333
Tuesday, Wednesday, Monday	Friday, Thursday	82.148	80.776
Tuesday, Wednesday, Thursday	Friday, Monday	83.580	82.038
Tuesday, Friday, Monday	Wednesday, Thursday	79.568	82.069
Tuesday, Friday, Thursday	Wednesday, Monday	86.313	84.840
Tuesday, Monday, Thursday	Wednesday, Friday	67.393	67.393
Wednesday, Friday, Monday	Tuesday, Thursday	95.620	95.392
Wednesday, Friday, Thursday	Tuesday, Monday	98.590	96.190
Wednesday, Monday, Thursday	Tuesday, Friday	78.830	78.508
Friday, Monday, Thursday	Tuesday, Wednesday	82.956	82.808
Tuesday, Wednesday, Friday, Monday	Thursday	99.656	97.864
Tuesday, Wednesday, Friday, Thursday	Monday	99.963	98.879
Tuesday, Wednesday, Monday, Thursday	Friday	69.378	69.629
Tuesday, Friday, Monday, Thursday	Wednesday	68.704	71.761
Wednesday, Friday, Monday, Thursday	Tuesday	96.316	93.523

TABLE VI: Generalization on the power set of per-day full and feature-reduced CIC-IDS datasets using a random forest with 10 estimators. IP addresses are excluded from the full dataset. All scores are percentages.

ing and sometimes decreasing. This suggests a complicated interaction between feature selection and stationarity. It may be interesting to explore this effect on larger and more complete data sets in future work.

REFERENCES

[1] B. Marr, "Why only one of the 5 vs of big data really matters," *IBM Big Data\& Analytics Hub. Available online at www. ibmbigdatahub. com/blog/whyonly-one-5-vs-big-data-really-matters (last accessed February 29*, 2015.

[2] V. N. I. Cisco, "The zettabyte era: Trends and analysis," *Updated (07/06/2017), https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html*, 2017.

[3] G. Slocombe *et al.*, "World's largest publicly revealed distributed denial of service attack," *Asia-Pacific Defence Reporter (2002)*, vol. 44, no. 3, p. 30, 2018.

[4] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of fourth international conference on information systems security and privacy, ICISSP*, 2018.

[5] ACM SIGKDD, "KDD Cup 1999 : Computer network intrusion detection." [Online]. Available: http://www.kdd.org/kdd-cup/view/kdd-cup-1999/Data

[6] P. Hoffman and B. Schneier, "Attacks on cryptographic hashes in internet protocols," Internet Requests for Comments, RFC Editor, RFC 4270, November 2005, https://tools.ietf.org/html/rfc4270#section-6. [Online]. Available: https://tools.ietf.org/html/rfc4270#section-6

[7] P. Ning and S. Jajodia, "Intrusion detection techniques," *The Internet Encyclopedia*, vol. 2, pp. 355–367, 2003.

[8] Y. Liao and V. R. Vemuri, "Use of k-nearest neighbor classifier for intrusion detection1," *Computers & security*, vol. 21, no. 5, pp. 439–448, 2002.

[9] W. Hu, Y. Liao, and V. R. Vemuri, "Robust anomaly detection using support vector machines," in *Proceedings of the international conference on machine learning*, 2003, pp. 282–289.

[10] C. Kruegel and T. Toth, "Using decision trees to improve signature-based intrusion detection," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2003, pp. 173–191.

[11] G. Münz, S. Li, and G. Carle, "Traffic anomaly detection using k-means clustering," in *GI/ITG Workshop MMBnet*, 2007, pp. 13–14.

[12] M. Zamani and M. Movahedi, "Machine learning techniques for intrusion detection," *arXiv preprint arXiv:1312.2177*, 2013.

[13] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1690–1700, 2014.

[14] K. Rai, M. S. Devi, and A. Guleria, "Decision tree based algorithm for intrusion detection," *International Journal of Advanced Networking and Applications*, vol. 7, no. 4, p. 2828, 2016.

[15] F. Zhang and D. Wang, "An effective feature selection approach for network intrusion detection," in *Networking, Architecture and Storage (NAS), 2013 IEEE Eighth International Conference on*. IEEE, 2013, pp. 307–311.

[16] S. Mukherjee and N. Sharma, "Intrusion detection using naive bayes classifier with feature reduction," *Procedia Technology*, vol. 4, pp. 119–128, 2012.

[17] L. Breiman, *Classification and regression trees*. Routledge, 2017.

[18] M. Köppen, "The curse of dimensionality," in *5th Online World Conference on Soft Computing in Industrial Applications (WSC5)*, vol. 1, 2000, pp. 4–8.

[19] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[20] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[21] R. W. Harrison and C. Freas, "Fuzzy restricted boltzmann machines," in *North American Fuzzy Information Processing Society Annual Conference*. Springer, 2017, pp. 392–398.