Exquisite Feature Selection for Machine Learning Powered Probing Attack Detection

Hamidah Alanazi†, Shengping Bi†, Tao Wang† and Tao Hou‡
†New Mexico State University, Las Cruces, NM, USA, {hamidah, sbi, taow}@nmsu.edu
‡Texas State University, San Marcos, TX, USA, taohou@txstate.edu

***Abstract*—Network attacks have been intensively studied by recent research. Probing attacks, however, seem not receiving as much attention as others, because they do not explicitly impact the network operations. Nevertheless, probing attacks may monitor network behaviors, extract web-sensitive information, and gather topology information of a target network, which opens a door for other attacks. It is critically important to understand the traffic patterns of network probing attacks and prevent suspicious probing activities from attackers.**

In this work, we present a novel user selection tool to build the optimal feature set that can characterize probing attacks. It consists of three modules: 1) feature correlation analyzer to remove highly correlated features for training efficiency; 2) coarse-grain feature selection to select key features that can describe the traffic patterns of probing attacks; 3) fine-grain feature refinement to understand temporal/spatial correlations among multiple packets to further improve the detection rate. In addition, we propose a fast hybrid training architecture that allows simultaneous training for both feature selection and attack detection to improve the overall training efficiency. In the experiment, we build a real-world network testbed to validate our design. The results show that the detection model can achieve a detection rate of up to 99.74% with the proposed fine-grain feature selection tool.

***Index Terms*—Probing Attacks, Machine Learning, Feature Selection, DDoS Attacks, Intrusion Detection.**

I. Introduction

The Proliferation of different mobile devices (e.g., IoT devices, mobile phones, laptops) has significantly facilitated data sharing and benefit network applications in various domains [1, 2]. However, it also raises a range of security risks and privacy concerns and leads to a dramatic growth of network attacks during the past few years, such as distributed denial-of-service (DDoS) attacks, user-to-root attacks (U2R), remote-to-local attacks (R2L), and probing attacks. Many of these attacks have been intensively studied by recent research [3–13]. Multiple machine learning models (e.g., Naive Bayes, Random Forest, J48, and MLP) and various feature sets have been proposed to achieve a high detection rate for different attacks (e.g., DDoS).

Network probing attacks, however, seem not receiving as much attention as others, because they are passively launched and do not explicitly compromise web servers and affect their operations. Nevertheless, it is critically important to understand the attack patterns of network probing attacks and prevent suspicious probing activities from attackers. By scanning open ports and listening to online activities, a probing attack is able to monitor behaviors of a network critical infrastructure, extract sensitive information from a web database, and gather topology information of a target network, which further opens a door for other attacks to exploit these vulnerabilities and exacerbate their adverse impacts. Particularly, an attacker can leverage probing attacks to identify the most critical infrastructure in the network and craft an advanced DDoS attack on the particular infrastructure.

In this work, we aim to understand different types of probing attacks and extract an effective feature set to identify them. Machine learning techniques have been proven an effective way to identify different types of network attacks. However, initial examination shows that machine learning models designed for other network attacks cannot achieve a good performance when identifying probing attacks. The main reason is that probing attacks behave quite differently from other network attacks. The feature patterns applied to other attacks may not work for probing attacks. For example, protocol type is an important feature when identifying DDoS attacks, but not for probing attacks. Further, we also notice that most datasets are only packet-based and do not include any temporal/spatial features. However, temporal/spatial features such as consecutive packet interval, are inherent characteristics of the network traffics and play an important role when detecting probing attacks.

As such, one of the most challenging tasks in our work is to select the proper feature set that can identify the traffic patterns of the probing attacks. In this paper, we develop a novel fine-grained feature selection tool to keep a pool of the most effective features to improve training efficiency and detection rate. Our feature selection tool consists of three modules:

- **Feature correlation analyzer** is to measure the linear relationship between two features. When multiple features are highly correlated, the model only requires one of them as the others cannot provide additional information. In our tool, we only keep one of them to improve the efficiency of the model training.
- **Coarse-grain feature selection** combines both *mutual information analysis* and *extra tree classifier* to rank the importance of features towards identifying the probing attacks. We also propose an iterative feature selection procedure that incrementally adds features according to their ranking until the training overhead grows faster than the prediction accuracy.
- **Fine-grain feature refinement** incorporate the Long

Short-Term Memory (LSTM) to understand the temporal correlations among multiple packets and reformat all selected features as a feature vector to further improve the detection accuracy.

In addition, we also propose a fast hybrid training architecture that allows simultaneous training for both feature selection and attack detection to further improve the overall training efficiency.

The contribution of the work is summarized as follows:

1) We have proposed a novel feature selection tool that can select the key features for probing attack detection but also remove redundant ones for training overhead reduction.
2) We have proposed a fine-grain feature refinement to study the temporal/spatial correlations among multiple packets to further improve the detection accuracy.
3) We have implemented a real-world network testbed to validate the proposed feature selection tool via multiple machine learning algorithms (i.e., Random Forest, Naïve Bayes, XGBoost, and AdaBoost). The results show that the detector can achieve a detection rate of 99.74% with the proposed fine-grain feature selection tool.

II. Data Set Description

In this section, we describe the dataset used to build our detection tool. We utilize a well-known dataset for our initial feature selection and model training. We also build a real-world testbed and collect our own dataset to further validate our designs.

A. KDDCUP99 Dataset

Without loss of generality, we adopt the widely used dataset, KDDCUP99 for our feature selection and initial model training [14]. The dataset includes a wide variety of network attacks (e.g., DoS, R2L, U2R, Probing) simulated in a military network environment.

In our study, we are particularly interested in probing attacks, which aim to extract privacy information and find potential vulnerabilities via system and network scanning. The dataset contains four types of probing attacks:

- **IPsweep:** It scans the network to learn the victim's IP address via ICMP echo requests.
- **Portsweep:** It is used by attackers to identify open ports and vulnerable areas of a target victim.
- **Nmap**: It is a tool to scan the target network by looking for open ports and online activities to get information about network parameters.
- **SATAN**: It can be used by attackers to gather essential information about the server and find the weak points to launch attacks.

We take a random sampling (i.e., 41,102 overall samples) of probing/normal traffics from the dataset, each of which comes with 41 features including basic features for individual TCP connections, traffic features within a certain time window, and content features relevant to specific domain knowledge. We will carefully examine all these features and select the most important ones for probing detection.

B. Dataset from Real-world Testbed

We also build a local network for real-world data collection to further validate our design. Our network consists of workstations, laptops, mobile devices, and IoT devices. All devices are connected to a 2.4 GHz WiFi router to form a client-and-server LAN. In particular, we build an Apache HTTP Server, which hosts a personal site as the target of probing attacks. All other devices are served as the clients and the attacker is one of them

The real-time data traffic is collected and saved as a PCAP file. We further develop a feature extraction tool to extract 27 features as KDDCUP99 dataset. Note that we drop 14 content-based features as they are not relevant to probing attacks. Overall, our dataset contains 69,000 samples, of which 34,500 are probing attacks and the other half are benign traffics. The real-world dataset will help us to evaluate the effectiveness of the proposed schemes.

III. Feature Selection

Feature selection is a key step for accurate and efficient attack detection. Since the traffic patterns of probing attacks are quite different from other attacks, features used for others cannot be adapted to detect probing behaviors. In this section, we propose a novel feature selection tool that selects the most important features to identify probing attacks as well as removes redundant ones to improve training efficiency.

A. Overview

As shown in Figure 1, Our feature selection tool consists of three components: 1) correlation analysis, 2) coarse-grain feature selection; 3) fine-grain feature refinement.

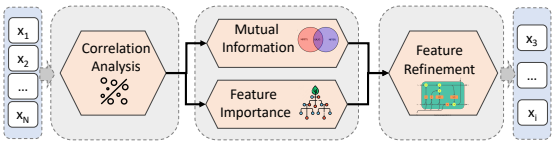

Fig. 1. Structure of the feature selection tool

- Correlation analysis is to measure the linear correlation between different pairs of features. When multiple features are highly correlated, we only keep one of them to improve the efficiency of the model training.
- In coarse-grain feature selection, we combine both mutual information analysis and extra tree classifier to rank the importance of features towards identifying the probing attacks. We iteratively select each feature according to their importance rank until the training overhead grows faster than the prediction accuracy.
- We notice that KDDCUP99 dataset is packet based and does not include flow based features. In fine-grain feature refinement, we incorporate the Long Short-Term Memory (LSTM) to learn the temporal/spatial correlations among

multiple packets and represent all selected features as a feature vector.

The objective of the proposed tool is to construct a feature vector that can characterize the patterns of probing attacks yet keep the training efficient.

B. Correlation Analysis

Correlation is to describe the inter-relationship between a pair of two features. When two features are highly correlated, they are linearly dependent and have the same effect on model training. As such when multiple features are highly correlated, we only keep one of them to improve the training efficiency.

In our tool, we conduct a statistical correlation analysis over all features and build a correlation matrix to indicate the strength of the linear relationship of any feature pairs. Assume we have N features $X_1, X_2, ...X_N$. For any pair of features X_i and X_j, their correlation C_{ij} is defined as

$$C_{ij} = \frac{Cov(X_i, X_j)}{\sigma X_i * \sigma X_j} = \frac{\sum(x_i - \bar{x}_i)(x_j - \bar{x}_j)}{\sqrt{\sum(x_i - \bar{x}_i)^2 \sum(x_j - \bar{x}_j)^2}},$$

where x_i and x_j are samples for features X_i and X_j respectively, and $\bar{x}_i$ and $\bar{x}_j$ are mean values for features X_i and X_j respectively. Figure 2 depicts the correlation matrix of all features. As shown, there indeed exist some features that are highly correlated, while most of them are independent from each other. In our tool, we only keep one feature when the correlation between two features is larger than 0.95 (e.g., the correlation between serror_rate and dst_host_serror_rate is 1). To this end, we remove 8 highly correlated features and keep the remaining 33 features for the next stage.

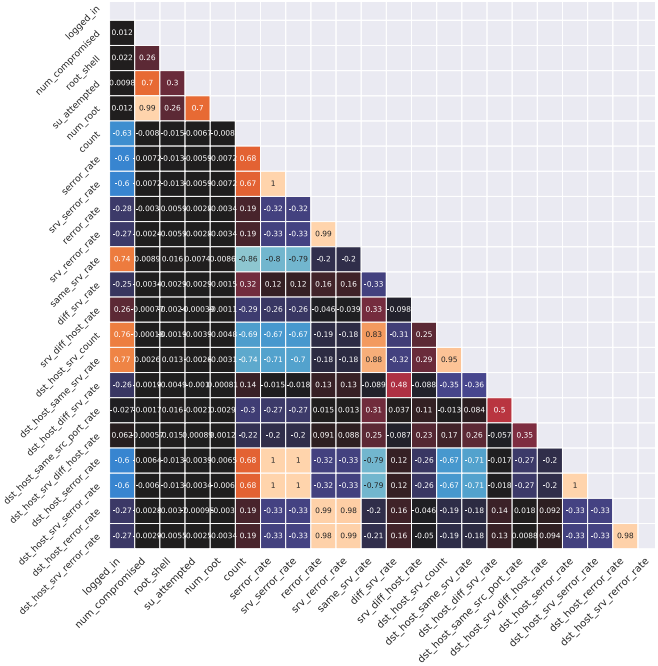

Fig. 2. Feature correlation matrix

C. Coarse-grain Feature Selection

In this session, we aim to build a feature pool that can best characterize the traffic patterns of the probing attacks.

First, we rank the importance of all available features. In order to better quantize the importance of each feature, the proposed tool integrates both *Mutual Information Analysis* and *Extra Trees Classifier*. Second, we build the feature pool for probing attack identification in an iterative manner. Specifically, we incrementally add features into our feature pool until new features cannot gain any performance improvement (i.e., the attack detection rate grows slower than the computational overhead).

1) Mutual Information Analysis: Mutual information (MI) is a model-neutral scheme that can measure the dependency of two variables [15]. In our tool, we adopt MI to measure the potential connection between the probing attack and specific features.

For any feature X_i in the dataset, the mutual information $I(A; X_i)$ between the attack and the feature is defined as the relative entropy conditioned on the feature X_i,

$$I(A; X_i) = H(A) - H(A|X_i),$$

where probing attacks and the feature are treated as two variables A and X_i respectively, $H(A)$ is the entropy of the probing attacks, and $H(A|X_i)$ is the attack entropy conditioned on X_i. Specifically, $H(A)$ indicates the uncertainty of the probing attacks and is defined as $H(A) = -\sum P_A(a) * \log P_A(a)$. Since our dataset is balanced (i.e., we have the same number of attack and benign samples), the statistical probability of a probing attack is 0.5 and $H(A)$ is computed as 1. $H(A|X_i)$ indicates the uncertainty of attacks when the feature X_i is known and is defined as $H(A|X_i) = \sum P_{X_i}(x_i) * H(A|X_i = x_i)$. In our tool, the MI value will fall in the range from 0 to 1, where a higher value indicates a closer connection between the feature and probing attacks, while a lower value indicates a weak connection. We plot the mutual information values for all features in Figure 3.

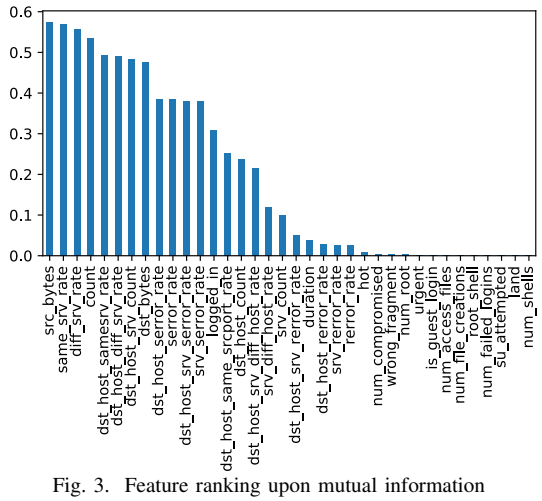

Fig. 3. Feature ranking upon mutual information

2) Extra Trees Classifier: We further incorporate a machine learning based ranking algorithm, extra trees classifier (extremely randomized trees classifier) to better describe the importance of different features. Extra trees classifier is an ensemble supervised machine learning technique that aggregates results from multiple uncorrelated decision trees.

In order to comprehensively assess each feature, we need to make the decision trees diversified and uncorrelated. Specifically, each decision tree in the extra trees classifier is built upon the random sampling of the original training dataset and starts with a randomly selected subset of features. Further, instead of choosing an optimum split for each feature, extra trees classifier randomly selects the splitting points to decorrelate different trees.

The feature importance in the extra trees classifier is computed based on *Gini Index (Gini impurity)*, which indicates how well a decision tree is split upon a specific feature. In particular, it calculates the amount of the probability of a specific feature being wrongly classified at the split node. For a particular split node n in the decision tree j, its Gini Index is defined as $Gini_{nj} = 1 - \sum_{a \in A} P_a^2$, where P_a is the probability of a sample being classified as a distinct class (i.e., probing attack/benign traffic) in this node. Assume the node is then split upon the feature X_i, the importance of X_i at the split node is computed as

$$FI_{X_{in}} = Gini_{nj} - Gini_{lj} - Gini_{rj},$$

Where $Gini_{lj}$ and $Gini_{rj}$ are the Gini Index of two child nodes respectively. Finally, we aggregate and normalize the feature importance for all features obtained from different decision trees and rank each feature based on their importance. Figure 4 plots the feature importance for all the features.

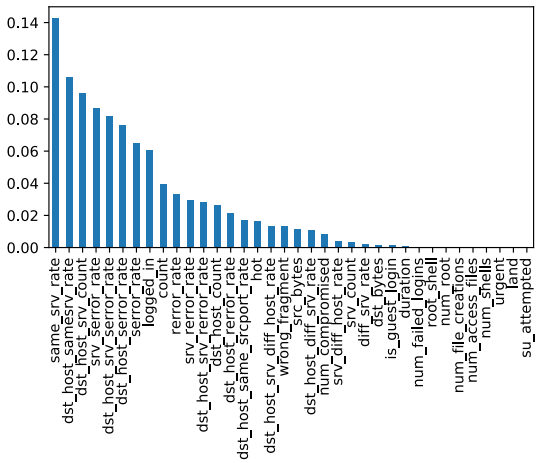

Fig. 4. Feature ranking upon importance value

3) Performance Gain: After ranking each feature according to their mutual information and feature importance, we carefully build our feature pool to improve the predictive accuracy and control over-fitting.

In the proposed tool, we incrementally add top features to our feature pool and compare the evaluation results (e.g., detection rate and training overhead) of different numbers of selected features via model pre-training. Specifically, we define a metric, performance gain (PG) to facilitate the comparison of different selected features.

$$PG = \frac{P_c - P_p}{P_c + P_p} - \frac{T_c - T_p}{T_c + T_p},$$

where P_c and P_p are the detection rates for current and previous number of selected features respectively, and T_c and T_p are training computational overhead for current and previous number of features respectively. A positive performance gain indicates detection rate grows faster than the training overhead when adding new features, while a negative performance gain indicates adding new features cannot benefit overall performance anymore. The proposed tool incrementally expands the feature pool until adding new features cannot yield a positive performance gain.

D. Fine-grain Feature Refinement

We notice that the dataset of KDDCUP99 only contains packet-based features and may not capture the correlations among multiple consecutive packets. However, packet correlations such as consecutive packet interval, are inherent characteristics of network traffics and play an important role when detecting probing attacks. To this end, we further concatenate the long short-term memory (LSTM) into our feature selection tool to understand the temporal/spatial correlations of probing packets and capture correlation features to improve detection efficiency.

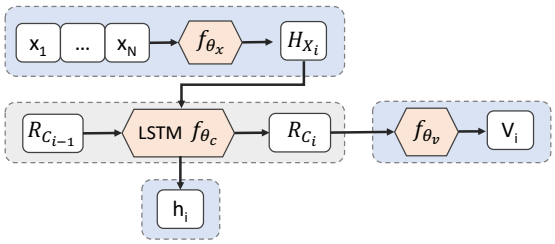

Fig. 5. Fine-grain feature refinement with LSTM

LSTM is a special kind of Recurrent Neural Network that can learn the long-term dependency by adding multiple memory cells. It shows good performance for time-series data processing, prediction, and classification[16]. Figure 5 shows the architecture of LSTM based feature refinement. As shown, the inputs (i.e., $x_1, x_2, ..., x_N$) are all features we have selected from previous steps. We further adopt a fully connected layer f_{θ_x} to summarize all the features and represent them as a feature vector H_{x_i}. The LSTM memory cell takes the feature vector H_{x_i} as input and learn the correlations with previous packet representations $R_{C_{i-1}}$. h_i is the output for attack identification of feature vector H_{x_i}. Finally we add another full connected layer f_{θ_v} to map the updated correlation R_{C_i} into a vector representation V_i. The LSTM based feature refinement could be used independently to identify probing attacks, but may impose additional training

and prediction overhead. Alternatively, it can be adopted for dataset pre-processing and combined with simple machine learning algorithms to improve their detection accuracy but also save the training time.

IV. Machine Learning Algorithm Concatenation

Our feature selection tool is model neutral and can be adopted by any typical machine learning algorithms. We also propose a fast hybrid training architecture to improve training efficiency for both feature selection and attack detector.

A. Hybrid Training

Instead of training feature selection and attack detector individually, we concatenate them altogether to launch a hybrid training procedure. As shown in Figure 6, we add an additional layer in front of the attack detector to accommodate various dimensions of feature inputs, such that we can support hybrid simultaneous training. The coarse-grain feature selection now gets feedback from both feature refinement and attack detector, the additional information can facilitate its decision-making on adding/removing/modifying the features from the candidate pool. The fine-grain feature refinement also gets additional feedback from the detector, which could help LSTM to better understand the temporal correlations of probing packets. The attack detector now starts training at the beginning and does not need to wait until we have the fine-grain feature set, which will expedite the detector training procedure.

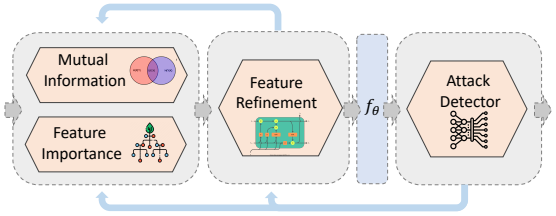

Fig. 6. Hybrid training structure

The hybrid training also supports module freezing. When the fine-grain feature set is obtained, we can freeze the user selection module and keep tuning the detector parameters even when the system is online.

B. Machine Learning Algorithms

We will validate our designs and incorporate our tool with typical machine learning algorithms.

- **Random Forest** is a tree-based ensemble learning classifier that includes a collection of decision trees derived from a subset of the training data [17].
- **Naïve Bayes** is a supervised machine learning algorithm that relies on the conditional independence of features to find the similarity and match to the output class to make the prediction [18].
- **XGBoost** is a shorthand for Extreme Gradient Boosting which supports parallel tree boosting-developed machine learning [19].
- **AdaBoost** is known for adaptive boosting which builds a composite strong learner by repeatedly adding weak learners through several cycles [20].

The above machine learning algorithms show good performance in traffic identification. We will use all of them to evaluate the performance of the proposed feature selection tool.

V. Experimental Analysis

In this section, we demonstrate the effectiveness of the proposed feature selection tool and validate our designs via multiple machine learning algorithms.

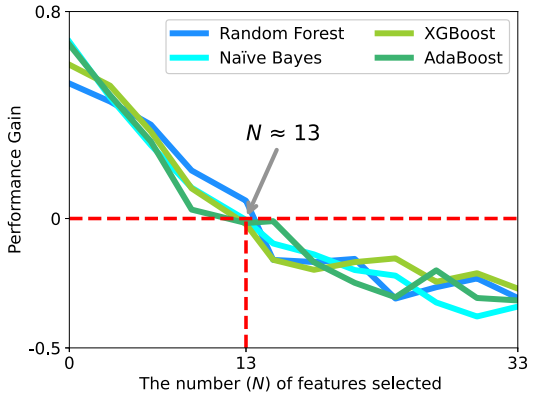

Fig. 7. Performance gain for different numbers of features

A. Performance Gain

In section III-C3, we define performance gain to compare the evaluation results when different numbers of top features have been selected. We start the feature set from the top 5 features. As shown in Figure 7, we can obtain a positive performance gain until the feature set reaches around 13 features, indicating that the top 13 features would be the best-chosen set for probing attack detection. Table I describes the top 13 features used in our experiment.

TABLE I
Top 13 features selected for probing attack detection

In.	*Feature Name*	In.	*Feature Name*
10	Hot	29	same_srv_rate
12	logged_in	30	diff_srv_rate
13	num_compromised	32	dst_host_count
15	su_attempted	33	dst_host_srv_count
23	Count	35	dst_host_diff_srv_rate
25	serror_rate	37	dst_host_srv_diff_host_rate
27	rerror_rate		

B. Evaluation Metrics

We use fowling metrics to evaluate the effectiveness of the proposed selection tool.

- Detection Accuracy: it is defined by the ratio of the number of accurate detections to all the detections, $Accuracy = (TP + TN)/(TP + TN + FP + FN)$,

where TP is the true positive, FN is the false negative, FP is the false positive and TN is the true negative.

- F1 Score: It is a widely used metric to evaluate the quality of a predictive system, $F1 = 2 * (Precision * Recall)/(Precision + Recall)$, where $Precision = TP/(FP + TP)$ and $Recall = TP/(FN + TP)$.
- Training Time: it records the actual time used for detection model training.

C. Evaluation Results on KDDCUP99

1) Coarse-grain Feature Selection: We compare the evaluation results when different numbers of top features are selected via various machine learning algorithms. As shown in Table II, the best-chosen feature set always yields the highest detection accuracy and F1 score for all machine learning algorithms. For example, random forest can achieve a detection accuracy of 99.68% when top 13 features have been used. On the other hand, when an insufficient or excessive number of features have been selected, the detection accuracy will drop a bit. Specifically, the detection rate of random forest with 5 features been selected is 96.65% and will drop to 45.60% with all features been used due to overfitting. As discussed in performance gain, the detection accuracy grows faster than the training overhead until the feature set increases to 13.

TABLE II
EVALUATION RESULTS WITH DIFFERENT FEATURES

Name	# Features	Accuracy	F1 score	Time (secs)
Random Forest	5	96.65%	0.97	5.519
	13	**99.68%**	**1.00**	**7.486**
	33	45.60%	0.63	10.983
Naive Bayes	5	95.83%	0.96	0.379
	13	**96.48%**	**0.97**	**0.58**
	33	54.39%	0.70	0.291
XGBoost	5	96.67%	0.97	6.833
	13	**99.36%**	**0.99**	**13.838**
	33	47.40%	0.63	26.032
AdaBoost	5	96.56%	0.97	42.752
	13	**99.67%**	**1.00**	**66.976**
	33	50.64%	0.65	94.854

2) Fine-grain Feature Refinement: We also adopt LSTM to further refine the selected feature set. As shown in Figure 8, the detector can achieve a trivial improvement and obtain a detection rate higher than 99.78%. But LSTM training takes 30.372s for KDDCUP99 dataset, which is a bit longer compared with Random Forest, Naive Bayers and XGBoost. Since model training may be needed periodically even when the detector is online. We can use LSTM feature refinement to extract the feature vector only. Online detector training can be done independently to save computational overhead.

D. Evaluation Results on Real-world Testbed

We also adopt the proposed feature selection tool into the real-world network testbed. Table III shows the performance when different machine learning techniques have been applied. Overall, the results show consistent performance as the results using KDDCUP99 dataset. The detection model can achieve a detection rate of up to 99.21% with the best chosen set obtained from the proposed feature selection tool.

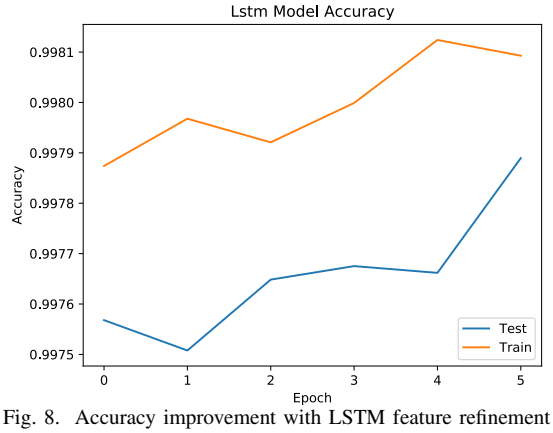

Fig. 8. Accuracy improvement with LSTM feature refinement

TABLE III
EVALUATION RESULTS USING REAL-WORLD NETWORK TESTBED

Name	# Features	Accuracy	F1 score	Time (secs)
Random Forest	5	94.44%	0.97	2.561
	13	**98.86%**	**0.99**	**2.89**
	27	94.31%	0.97	3.791
Naive Bayes	5	98.02%	0.98	0.328
	13	**98.75%**	**0.99**	**0.004**
	27	56.84%	0.50	0.013
XGBoost	5	99.10%	1.00	3.74
	13	**99.21%**	**1.00**	**9.248**
	27	99.19%	1.00	10.195
AdaBoost	5	94.22%	0.97	13.537
	13	**98.99%**	**0.99**	**25.963**
	27	98.24%	0.99	47.349

VI. CONCLUSION

In this work, we have proposed a novel feature selection tool to construct a feature vector that can characterize the patterns of probing attacks yet keep the training efficient. We also developed an LSTM-based fine-grain feature refinement to study the temporal/spatial correlations of probing packets to further improve the detection accuracy. In addition, we have built a real-world network testbed to validate the proposed feature selection tool via multiple machine learning algorithms. We plan to implement a deep learning model to detect the probing attack for future work.

VII. ACKNOWLEDGMENT

The work was sponsored by the DEVCOM Analysis Center under Cooperative Agreement Number W911NF-22-2-0001 and NSF under grant ECCS-2139028. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

[1] F. Jiang, B.-c. Wang, C.-y. Sun, Y. Liu, and X. Wang, "Resource allocation and dynamic power control for d2d communication underlaying uplink multi-cell networks," *Wireless Networks*, vol. 24, pp. 549–563, 2018.

[2] F. Jiang, Y. Liu, B. Wang, and X. Wang, "A relay-aided device-to-device-based load balancing scheme for multitier heterogeneous networks," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1537–1551, 2017.

[3] P. S. Saini, S. Behal, and S. Bhatia, "Detection of ddos attacks using machine learning algorithms," in *2020 7th International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2020, pp. 16–21.

[4] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning ddos detection for consumer internet of things devices," in *2018 IEEE Security and Privacy Workshops (SPW)*, 2018, pp. 29–35.

[5] M. H. Kamarudin, C. Maple, and T. Watson, "Hybrid feature selection technique for intrusion detection system," *International Journal of High Performance Computing and Networking*, vol. 13, no. 2, pp. 232–240, 2019.

[6] S. Fang, Y. Liu, and P. Ning, "Mimicry attacks against wireless link signature and new defense using time-synched link signature," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 7, pp. 1515–1527, 2016.

[7] Y. Liu and P. Ning, "Poster: Mimicry attacks against wireless link signature," in *Proceedings of the 18th ACM conference on Computer and communications security*, 2011, pp. 801–804.

[8] Y. Meng, J. Li, H. Zhu, X. Liang, Y. Liu, and N. Ruan, "Revealing your mobile password via wifi signals: Attacks and countermeasures," *IEEE Transactions on Mobile Computing*, vol. 19, no. 2, pp. 432–449, 2019.

[9] S. Fang, Y. Liu, W. Shen, H. Zhu, and T. Wang, "Virtual multipath attack and defense for location distinction in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 16, no. 2, pp. 566–580, 2016.

[10] A. Alagil, M. Alotaibi, and Y. Liu, "Randomized positioning dsss for anti-jamming wireless communications," in *2016 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2016, pp. 1–6.

[11] Z. Li, Q. Pei, I. Markwood, Y. Liu, M. Pan, and H. Li, "Location privacy violation via gps-agnostic smart phone car tracking," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 5042–5053, 2018.

[12] S. Fang, I. Markwood, Y. Liu, S. Zhao, Z. Lu, and H. Zhu, "No training hurdles: Fast training-agnostic attacks to infer your typing," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1747–1760.

[13] T. Wang, Y. Liu, Q. Pei, and T. Hou, "Location-restricted services access control leveraging pinpoint waveforming," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 292–303.

[14] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE symposium on computational intelligence for security and defense applications*. Ieee, 2009, pp. 1–6.

[15] F. Amiri, M. R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, "Mutual information-based feature selection for intrusion detection systems," *Journal of Network and Computer Applications*, pp. 1184–1199, 2011.

[16] Y. Li and Y. Lu, "Lstm-ba: Ddos detection approach combining lstm and bayes," in *2019 Seventh International Conference on Advanced Cloud and Big Data (CBD)*. IEEE, 2019, pp. 180–185.

[17] R. Santos, D. Souza, W. Santo, A. Ribeiro, and E. Moreno, "Machine learning algorithms to detect ddos attacks in sdn," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 16, p. e5402, 2020.

[18] S. Chen, G. I. Webb, L. Liu, and X. Ma, "A novel selective naïve bayes algorithm," *Knowledge-Based Systems*, vol. 192, p. 105361, 2020.

[19] B. Pan, "Application of xgboost algorithm in hourly pm2. 5 concentration prediction," in *IOP conference series: earth and environmental science*, vol. 113, no. 1. IOP publishing, 2018, p. 012127.

[20] C. Ying, M. Qi-Guang, L. Jia-Chen, and G. Lin, "Advance and prospects of adaboost algorithm," *Acta Automatica Sinica*, vol. 39, no. 6, pp. 745–758, 2013.