Evaluation of SVM and KNN in Website Fingerprinting Attack

Mohammad Saidur Rahman

Abstract

Website fingerprinting gives a local passive eavesdropper to identify web activity of a client. The attacker can do that by leveraging the packet information such as packet size, packet direction, timestamps, and packet bursts. These attacks break the privacy of a user of a privacy enhancing technologies including the low-latency anonymity networks such as Tor. Attackers can deanonymize a user by analyzing the traffic. They classify the websites by different machine learning techniques such as k-nearest neighbor (KNN) [1], support vector machines (SVM) [2], and random decision forests [3]. KNN (90.90%) and SVM (90.59%) classifiers showed significant accuracy in terms of attacks. But the features are different for each case. General features, packet ordering, packet direction, concentration of outgoing packets, packet bursts (packet burst is the sequence of packets in any direction), timestamps, and distance are considered for KNN and SVM classification. However, the number of features extracted for KNN is 735 and the number of features extracted for SVM is 105. Both classifiers are able to give 90% accuracy equivalently. In this project, we investigated the significance of the number of features, feature sets and the underlying impact on classifier accuracy. After extensive experiment, we got 89.38% accuracy with KNN features in SVM classifier. And 86.63% accuracy with SVM features in KNN classifier. We have found that the feature sets of KNN is more robust than the features of SVM irrespective of the classifier in this experiment.

Introduction

The right to privacy is one of the fundamental human right that needs to be preserved over the Internet. Tor: a low-latency anonymous communication system is one of the advance privacy enhancing technologies. Tor enables users to communicate in a manner that is untraceable by adversaries and allows people and groups to improve their privacy and security on the internet [4]. Tor is a distributed overlay network based on the idea of mixes to create a secure network circuit [5]. A client establishes a circuit of encrypted connections through relays on the overlay network Figure 1: Tor Network.Figure 1). The circuit is gradually extended one hop at a time, and each relay along the way knows only which relay gave it data and which relay it is giving data to. No individual relay knows complete information about the path that a data packet has flown through [6]. This sophisticated encryption over the Tor circuit obscures both content and meta-data, preventing a passive attacker from identifying the source or destination of communication [3].

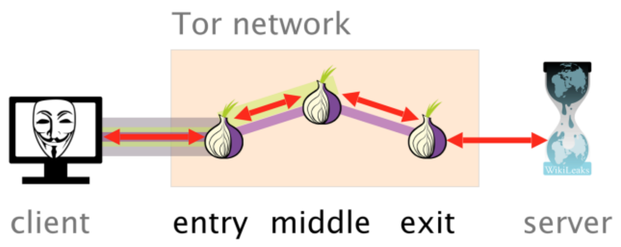

Figure 1: Tor Network.

Background

Tor

Tor is a low-latency anonymity network. A tor circuit consists of three nodes (guard node, middle node, and exit nodes). It uses multilayered encryption in each node. Guard node only knows about the identity of the user and exit node only knows about the original TCP packets sent by the user [5]. One node is only aware of the previous and next node of the circuit. An eavesdropper, who can control guard node or exit node, can deanonymize a user.

Website Fingerprinting Attack

Website fingerprinting attack is one of the potential traffic analysis attacks by monitoring the nature and behavior of traffic [7]. The Website fingerprinting attack is identified as classification problem which has the aim to be able to identify which a website the user has just visited. A so-called local adversary can be for example a local system administrator, an ISP, or anybody who is able to monitor the traffic between the client and the guard node of anonymity system. The local adversary captures the traffic to gather traffic-related statistical information as features such as size, direction, and timing of packets transmitted between clients and web servers [8]. The adversary then compares these extracted features with the set of trained classifiers such as KNN and SVM which were already prepared for expected websites which he would like to monitor.

Machine Learning

Machine learning is a process of taking the input of a certain type of knowledge and predicting another type of knowledge as output. This prediction capability to produce new knowledge is called the learning. In accordance with the view of Blockeel [9], “the system can learn if *it has the capacity to improve its own performance at solving certain problems after receiving additional information about the problem"*.

Website fingerprinting attacker collects traffic of several websites of her interest. These websites are called monitored websites. After that, she feeds these traffic data based on traffic features such as incoming and outgoing packets, bursts, and timestamps to machine learning model to learn from this input. This process is called training.

In the next step, she feeds her collected tor traffic to the machine learning model to predict the new output. This process is called testing.

Related Work

In 2009, first website fingerprinting attack was published by Herrmann et al. [10]. They only considered IP packet size for their classifier. Panchenko et al. designed a new attack adding more features: packet volume, packet direction, and timing [8]. They used support vector machine as their classifier. In 2012, SVM was again used by Cai et al. who proposed a new attack based on a new representation of the classification instances [11]. Their SVM was using the Damerau-Levenshtein edit distance and used the SVM kernel trick to pre-compute distance between the traces. This same attack was improved by Wang and Goldberg [2]. In 2014, Wang et al. proposed a new attack based on a k-Nearest Neighbor classifier on a large feature set with weight adjustment[12].

Hayes et al. use yet a novel feature extraction and selection method: they use random forests to extract robust fingerprints of webpages [13]. In 2016, Panchenko et al. proposed a new attack improving features based on packet size, packet ordering, and packet direction [14]. They used the concept of Wang et al. to develop their K-NN classifier based attack.

Methodology

Attack Model

We assume that the attacker is a passive and local eavesdropper. The attacker can collect the packets in the path of a client and the entry guard of Tor (Figure 2).

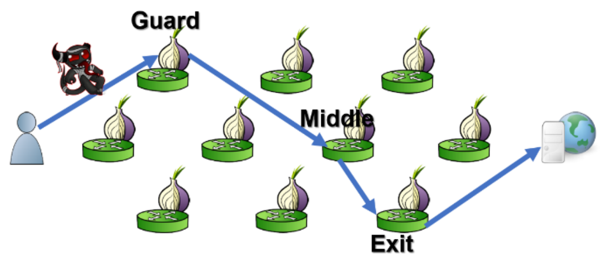

Figure 2: A Passive and Local Eavesdropper.

The attacker can analyze the traffic by machine learning techniques, extract features and feed into machine learning classifiers (Figure *3*).

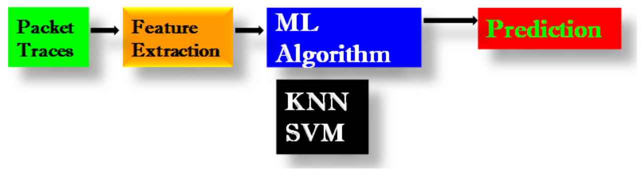

Figure 3: Operation of an Attacker.

Classifiers

Feature set for KNN classifier are packet direction, packet bursts, general features, and packet distributions.

Feature set for SVM classifier considers number of incoming packets, number of outgoing packets, size of incoming packets, and size of outgoing packets.

Experiment

Experimental Environmental

For the experiment, we used Linux-Ubuntu environment. We used GPU for faster computation. Codes are written and modified on Python for classification. For feature extraction, C++ is used. The two robust machine learning techniques KNN and SVM are the classifiers for the experiment.

KNN Features-KNN Classifier

Our experiment was in closed world setting using the data set of Tao Wang. We eliminated the feature *'unique packet length'* as in Tor the packet lengths are uniform for all packets (cell padding). This experiment takes account of 100 monitored sites with 60 instances for learning and 30 instances for testing. The total number of feature extracted was 735 and the best accuracy of attack is 90.90%. Number of neighbor is one in our experiment. And k=5 for our cross-validation process.

Table 1: Experiment of KNN_features__KNN_classifier.

Round	Accuracy (%)	Best Accuracy (%)
0-0	90.90	90.90
0-1	90.50	90.90
0-2	90.90	90.90
0-3	90.43	90.90
0-4	90.23	90.90

SVM Features-KNN Classifier

This experiment is likely to the previous one with some changes in the feature extraction code as we had to change format of the output data files. We changed the output format of SVM features keeping the flearner code unchanged. The features set was taken from Andriy Panchenko. The experimental setting is same. In this case, we got the number of features 105. The best accuracy was 86.63%.

Table 2: Experiment of SVM_features__KNN_classifier.

Round	Accuracy (%)	Best Accuracy (%)
0-0	85.73	85.73
0-1	86.23	86.23
0-2	86.60	86.60
0-3	86.63	86.63
0-4	86.43	86.63

SVM Features-SVM Classifier

For SVM classifier, we had to first process the data in vector format (like an array). After that we used easy.py file to train and test. The number of features is 104 in this case and it is different than the previous one because we omitted the first cumulative feature in this case. and the accuracy rate is 90.59%.

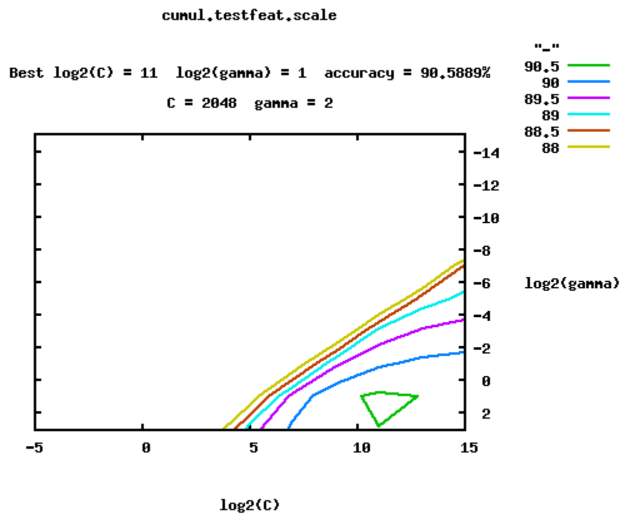

Figure 4: Experiment of SVM_features__SVM_classifier.

KNN Features-SVM Classifier

In this scenario, we had to change the output format to process the KNN features for SVM classification. As in this case, the number of features is 735. It takes almost 22 hours in virtual machine to process the code and gives the result. After getting the output data file, we followed the same process as the previous scenario. The accuracy rate is 89.38%.

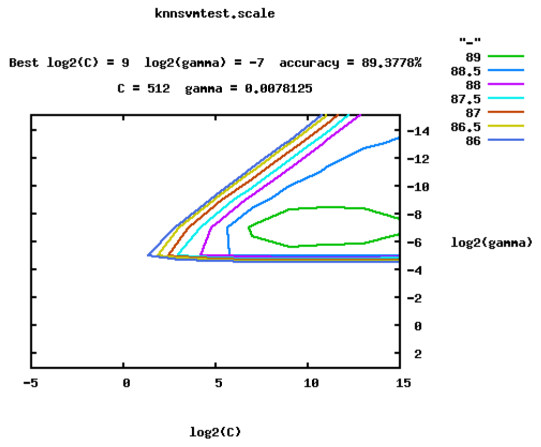

Figure 5: Experiment of KNN_features__SVM_classifier.

Result

We tried to evaluate classifiers (SVM or KNN) and the feature sets. Four experiments are performed for this analysis purpose. First two experiments are similar to the previous work of KNN and SVM. Next two experiments are conducted changing the features and classifiers. The changes are done as KNN features to SVM classifier, and SVM features to KNN classifier. We can see the experimental result at a glance from Table *3*.

Table 3: Result of the Experiment

Classifier	**Accuracy (%)**	
	KNN Features	SVM Features
KNN	90.90	86.63
SVM	89.38	90.59

Analysis of KNN and SVM Features

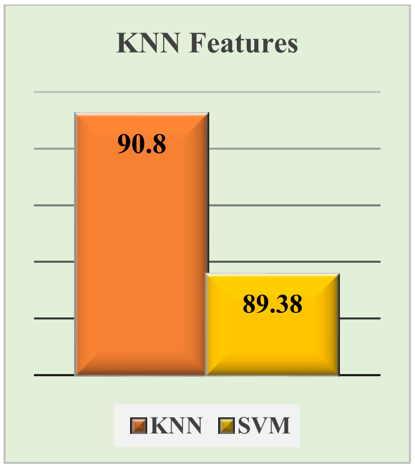

Figure 6: KNN Features in both Classifiers.

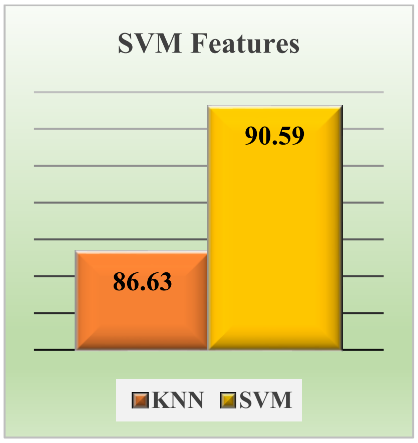

Figure 7: SVM Features in both Classifiers.

From the classifier accuracy or attack accuracy comparison, we can see that the KNN features gave closer accuracy (90.90 and 89.38) in both classifiers. From this observation, we can say that KNN features are better than SVM features regardless of those two classifiers. As the number of SVM features for KNN classifier and SVM features are not the same, this might be the reason for the greater variation of accuracy considering the SVM features.

Conclusion

This project dealt with website fingerprinting experiment in four scenarios taking account of k-nearest neighbor (KNN) and support vector machine (SVM) classifiers. Same set of data were used for all the four scenarios. Firstly, we tried to examine the classification accuracy of predefined features for KNN and SVM classifiers. The features for KNN and the features for SVM were different. In the second phase of the experiment, we tried to examine the classification accuracy by interchanging the classifiers of predefined features, such as features for KNN classifiers were used as the features for SVM classifiers. In the experiment, we can see that, with the interchanged features and classifiers, the accuracy rate is low for both cases. And this experiment gave us an understanding that relatively KNN features are better than the SVM features.

Future Research

Many research works have already been published proposing novel solutions to combat website fingerprinting attack [2, 3, 13-17]. But with the emergence of deep learning which is a very powerful tool to solve classification problem, an attacker can perform website fingerprinting attack with less effort and more efficiently because deep neural network does not require manual feature selection. Only one research is published on website fingerprinting attack using deep learning that considered incoming and outgoing packets for input data[16]. We believe, there are other ways to feed the input data such as considering timestamp with incoming and outgoing packets. This is one of the primary challenges of an attacker to perform website fingerprinting using deep learning. However, we still believe deep learning will answer a lot of open questions in the research of anonymity systems.

Data and Codes

Data and Codes can be found from the following link.

https://rit0-my.sharepoint.com/personal/mr6564_rit_edu/_layouts/15/guestaccess.aspx?folderid=1f696e2f68158477990a73b6e974b8b0b&authkey=AYwq10n6XNVPZkIZI4doi3g

This is a secured link shared in RIT One Drive.

References

[1] T. Wang, X. Cai, R. Nithyanand, R. Johnson, and I. Goldberg, "Effective Attacks and Provable Defenses for Website Fingerprinting," in *USENIX Security*, 2014, pp. 143-157.

[2] T. Wang and I. Goldberg, "Improved website fingerprinting on tor," in *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, 2013, pp. 201-212: ACM.

[3] J. Hayes and G. Danezis, "k-fingerprinting: A robust scalable website fingerprinting technique," *arXiv preprint arXiv:1509.00789,* 2016.

[4] A. Pfitzmann, B. Pfitzmann, and M. Waidner, "ISDN-mixes: Untraceable communication with very small bandwidth overhead," in *Kommunikation in verteilten Systemen*, 1991, pp. 451-463: Springer.

[5] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," DTIC Document2004.

[6] Tor. (2017, March 10). *Tor: Overview*. Available: https://www.torproject.org/about/overview.html.en

[7] S. Bruce, "Applied cryptography: protocols, algorithms, and source code in C," *John Wiley & Sons, Inc., New York,* 1996.

[8] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel, "Website fingerprinting in onion routing based anonymization networks," in *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, 2011, pp. 103-114: ACM.

[9] H. Blockeel, "Machine Learning and Inductive Inference," *ACCO Leuven,* 2010.

[10] D. Herrmann, R. Wendolsky, and H. Federrath, "Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier," in *Proceedings of the 2009 ACM workshop on Cloud computing security*, 2009, pp. 31-42: ACM.

[11] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proceedings of the 2012 ACM conference on Computer and communications security*, 2012, pp. 605-616: ACM.

[12] T. Wang, X. Cai, R. Nithyanand, R. Johnson, and I. Goldberg, "Effective Attacks and Provable Defenses forWebsite Fingerprinting," San Diego, CA, USA, 2014: 23rd USENIX Security Symposium.

[13] J. Hayes and G. Danezis, "Website fingerprinting at scale," *University College of London (UCL), number: Technical report,* 2015.

[14] A. Panchenko *et al.*, "Website fingerprinting at internet scale," in *Network & Distributed System Security Symposium (NDSS). IEEE Computer Society*, 2016.

[15] M. Juarez, M. Imani, M. Perry, C. Diaz, and M. Wright, "Toward an efficient website fingerprinting defense," in *European Symposium on Research in Computer Security*, 2016, pp. 27-46: Springer.

[16] K. Abe and S. Goto, "Fingerprinting Attack on Tor Anonymity using Deep Learning," Pok Fu Lam, Hong Kong, 2016: Proceedings of the APAN – Research Workshop 2016.

[17] V. Rimmer, "Deep Learning Website Fingerprinting," 2017.