EasyChair Preprint

№ 3740

SILU: Strategy Involving Large-scale Unlabeled Logs for Improving Malware Detector

Taishi Nishiyama, Atsutoshi Kumagai, Kazunori Kamiya and Kenji Takahashi

July 3, 2020

SILU: Strategy Involving Large-scale Unlabeled Logs for Improving Malware Detector

Taishi Nishiyama*, Atsutoshi Kumagai*, Kazunori Kamiya* and Kenji Takahashi†
*NTT Secure Platform Laboratories, Tokyo, Japan
†NTT Ltd., PaloAlto, U.S.
*{taishi.nishiyama.pt, atsutoshi.kumagai.ht, kazunori.kamiya.ew}@hco.ntt.co.jp, †kenji.takahashi@global.ntt

***Abstract*—Machine learning is becoming a key component to automatically detect malware-infected hosts by analyzing network logs in a security operations center (SOC). However, machine learning usually requires a large amount of labeled training data, which is difficult to acquire since labels are manually set by professional security analysts. On the other hand, abundant unanalyzed logs are kept stored in daily operation and stay unlabeled even though they could compensate for the lack of existing labeled training data. This paper proposes SILU, a novel semi-supervised learning method, which fully leverages unlabeled data and enhances detection capability without increasing manually labeled data. SILU learns from combined labeled and unlabeled training data to automatically augment labeled training data and then generates a classifier through the screening process. Unlike most semi-supervised learning methods used in cyber security, which use test data as unlabeled training data, SILU does not require retraining every time test data change since it can use different datasets for unlabeled training and test data. This helps SOC operation for practically suppressing detecting time. In addition, though SILU partially includes a supervised learning method, it does not require a specific supervised learning method. Therefore, SILU can be added on to any type of classifier of a supervised learning method. Moreover, SILU can suppress the deterioration of classification performance for test data through the screening process. We evaluated SILU using two types of real-world logs: proxy logs from a large enterprise and NetFlow from a large ISP. We demonstrated that by evaluating with different types of classifiers, SILU always improves detection capability for supervised learning methods. SILU also outperforms current semi-supervised methods. As a whole, SILU works as an add-on to existing supervised learning methods with little overhead and performs better than conventional supervised learning methods. Our evaluation also shows that using NetFlow from ISP as unlabeled training data works better than using only labeled proxy logs in the same enterprise. These results suggest that SILU can extend detection capability more when different organizations, e.g., SOCs and ISPs, collaborate and share unlabeled data.**

***Index Terms*—Malware Detection, Semi-supervised Learning, Log Analysis, HTTP Traffic, NetFlow**

I. Introduction

For security management, log analysis can improve malware detection, which is important to mitigate the damage of malware infections [1], [2]. Machine learning methods for log analysis are attracting attention to sophisticate malicious traffic detection methods and reduce the burden on analysts in a security operations center (SOC). Although malware creators are constantly sophisticating their attack methods, most new malicious traffic has similar characteristics to existing malicious traffic since malware creators often reuse existing attack methods [3]. Machine learning methods are thus likely to detect malicious traffic by capturing those characteristics.

One of the most serious challenges in creating a machine-learning-based classifier that classifies a host as legitimate or malicious is to prepare abundant labeled training data to achieve a versatile and highly accurate classifier; labeled training data are data that have already been determined to be legitimate or malicious. However, labeled training data are difficult to obtain because analysts manually assign labels to training data after detailed inspection.

On the other hand, unlabeled training data are much easier to obtain in large quantities since they do not require labels. For example, unanalyzed proxy logs are stored for recording purposes in a SOC, and unanalyzed NetFlow is stored for provisioning purposes in an ISP. NetFlow was originally implemented by Cisco IOS Software [4] and becomes a default standard to collect IP network traffic information. Unlabeled training data could include much malicious and legitimate traffic that do not appear in labeled training data. Therefore, if the classifier can learn some characteristics from unlabeled training data in addition to labeled training data, we can achieve a more accurate classifier. Specifically, since an ISP provides connectively to many enterprises, ISP NetFlow contains much malicious traffic and legitimate traffic that do not appear in several corporate networks, which would be useful for creating a versatile and highly accurate classifier.

This paper proposes SILU, a novel semi-supervised learning method, for fully leveraging abundant unlabeled training data as well as labeled training data to generate a classifier for detecting malware-infected hosts. SILU mainly consists of three parts: 1) propagation, which predicts the probability that unlabeled training data are legitimate or malicious, 2) screening, which selects unlabeled training data with a high probability of being legitimate or malicious, and 3) supervised learning, which creates a classifier from original labeled and selected unlabeled training data. The key advantages of SILU are that it can predict the test data that are not used for unlabeled training data, be used as an add-on to any type of classifier of a supervised learning method, and suppress the deterioration of classification performance for test data due to mislabeling. The details of SILU are described in section IV.

This paper provides evaluations according to actual use cases of a SOC by using real-world traffic logs. More specifi-

cally, we used proxy logs from an actual large enterprise and NetFlow from an actual large ISP network. Since proxy logs are mainly used in a security log analysis service in a SOC, we consider creating a high performance classifier for proxy logs. Hence, evaluations were executed using labeled proxy logs. For evaluating the effectiveness of adding unlabeled data, we used unlabeled proxy logs and unlabeled NetFlow.

In the evaluation, we compared SILU with conventional supervised learning methods (logistic regression (LR), support vector machine (SVM), and random forest (RF)) on the basis of the area under the curve (AUC) and true positive rate (TPR) at a low false positive rate (FPR). We focus on TPR at a low FPR as a crucial performance index since it affects the operation cost in a SOC. The results indicate that SILU always improves the AUC and TPR at FPR $= 0.1\%$. The results also indicate that the more SILU learns from unlabeled training data, the better the classification performance. In addition, we compared SILU with current semi-supervised learning methods for cyber security, i.e., hybrid learning (HL) [5] and learning with local and global consistency (LLGC) [6], since these methods are graph-based semi-supervised learning methods similar to SILU. From the results, SILU also achieves higher classification performance than the current semi-supervised learning methods.

This paper makes the following contributions:

- We propose SILU, a novel semi-supervised learning method, for enhancing detection capability without increasing manual labeling cost. The advantages of SILU are that it does not require retraining every time test data change in the detecting phase, can easily be extended to conventional supervised learning methods, and can suppress mislabeling.
- We evaluated SILU with real-world proxy logs from a large enterprise and NetFlow from a large ISP and show that SILU can obtain better AUC and TPR at FPR $= 0.1\%$ than conventional supervised learning methods. We also showed the effectiveness of learning with unlabeled training data.
- We demonstrate that the performance of SILU improves when using NetFlow as unlabeled training data while classifying proxy logs. NetFlow can be used as a source of information that can not be observed by monitoring traffic logs in a SOC.

II. Related Work

This paper intersects the domains of cyber security and semi-supervised learning. We briefly mention related works involving both domains below.

Some studies in cyber security have focused on generative-model-based [7] and discriminative-model-based [8] semi-supervised learning methods to handle both labeled and unlabeled training data. Osada et al. [7] used deep generative models and a variational auto-encoder to detect malicious logs from logs of network intrusion detection systems. Zhang et al. [8] presented a collaborative SVM method that uses two SVM classifiers to detect malicious code from executable files. Although their methods show the effectiveness in terms of classification performance, they do not have model interpretability. Since analysts eventually confirm whether customer hosts are infected by manually analyzing customer logs, the reasons for detection should be clarified to save time. On the other hand, graph-based semi-supervised learning methods can handle a sparse matrix well, which must be suitable for log analysis, and make it easy to understand the relationships among network logs. Therefore, SILU utilized label spreading [9], a graph-based semi-supervised learning method.

There are also several studies using graph-based semi-supervised learning methods. Shi et al. [5] proposed HL, a system that independently utilizes graph-based semi-supervised learning and supervised RF, to propagate known malicious domain reputation through an unweighted bipartite graph representing the communication relationship between users and domains. However, this method is difficult to apply to data having multiple features because the significance of each feature is not considered in an unweighted bipartite graph. Santos et al. [6] proposed the LLGC method including a label spreading algorithm for detecting unknown malware from files. Stringhini et al. [10] presented Marmite, a system that used a semi-supervised Bayesian label propagation to propagate the reputation of known files across a download graph that comprehensively encapsulates how files are downloaded. Though Bayesian label propagation tends to achieve high classification performance when there are few neighbors by providing a confidence level to the inference results, it increases computational cost and makes the relationships among nodes difficult to understand, which are disadvantages for SOC operation.

The difference between SILU and the above semi-supervised learning methods [5]–[8], [10] is that SILU does not require retraining since it can use different datasets for the training and detecting phases. Retraining is unrealistic in SOC operation in terms of computational cost. In addition, SILU avoids mislabeled data through the screening part described in subsection IV-B. Moreover, we created a classifier using different types of logs, i.e., labeled proxy logs from a large enterprise and unlabeled NetFlow from a large ISP, for the first time in log analysis.

III. Problem Statement

The goal with SILU is to generate a machine-learning-based classifier that classifies a host as legitimate or malicious by fully leveraging both labeled and unlabeled training data. If a host is classified as malicious, analysts analyze it in detail to confirm whether a host is truly infected in SOC operation. In short, we expect to use SILU to select targets for detailed inspection, which will reduce the burden on analysts.

In this study, we consider two cases since we focus on improving classifier performance in a SOC.

Case 1: Labeled training data: proxy logs
Unlabeled training data: proxy logs
Test data: proxy logs

Case 2: Labeled training data: proxy logs
Unlabeled training data: NetFlow
Test data: proxy logs

Since much malware uses HTTP as transport in drive-by-download and communications between C&C servers, security log analysis services in a SOC widely use proxy logs. Therefore, we use proxy logs as labeled training and test data. Cases 1 and 2 assume that we improve a classifier with unanalyzed proxy logs from a customer's network and NetFlow from a large ISP network, respectively. In Case 2, we expect to obtain much information related to malicious communications from a large-scale network. Collection points of proxy logs are limited to enterprise customers' networks, and proxy logs from other various kinds of networks are not easy to collect. In contrast, NetFlow is easy to collect since they do not include HTTP information, so they can be collected even from large-scale networks such as an ISP. Although proxy logs and NetFlow are different in nature, various kinds of malware can be detected by using information common to proxy logs and NetFlow, e.g., destination IP addresses. Therefore, by using NetFlow as unlabeled training data while classifying proxy logs, SILU can extend detection capability.

IV. Proposed Method: SILU

SILU mainly consists of three parts: 1) propagation, which predicts the probability that unlabeled training data are legitimate or malicious with label spreading, 2) screening, which selects unlabeled training data with a high probability of being legitimate or malicious, and 3) supervised learning, which creates a classifier from original labeled and selected unlabeled training data. We explain the details of each part in subsections IV-A∼IV-C and then summarize the advantages of SILU in subsection IV-D. We consider binary classification for simplicity, although the discussion in the paper can easily be expanded to a multiclass classification.

A. Propagation Part

In the propagation part, SILU predicts the probability of unlabeled training data being legitimate or malicious with label spreading. Label spreading is a graph-based semi-supervised learning method proposed by Zhou et al. [9] that assigns the same label to similar feature vectors. The goal in conventional label spreading is to predict the labels of unlabeled training data, but SILU uses label spreading to calculate the probability of unlabeled data being legitimate or malicious.

Given labeled training data $D_L = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_l, y_l)\}$ and unlabeled training data $D_U = \{\mathbf{x}_{l+1}, \cdots, \mathbf{x}_{l+u}\}$, where $\mathbf{x}_p \in \mathbb{R}^m$ is an m-dimensional feature vector of a data index p, and $y_p \in \{0, 1\}$ is its label. Labels y_p are assigned such as legitimate $= 0$ and malicious $= 1$. We use F_{ij} to denote the probability to be estimated where the label of $\mathbf{x}_i$ is $j-1$ and $F \in \mathbb{R}_+^{n\times 2}$ $(n = l+u)$ to denote a vector function matrix where the (i, j)-element is F_{ij} where $\mathbb{R}_+$ is the set of non-negative real numbers. That is, F_{i1} and F_{i2} correspond to the probability of being legitimate or malicious, respectively. The notation $Y \in \mathbb{R}_+^{n\times 2}$ is an initial label matrix where $Y_{ij} = 1$ if $\mathbf{x}_i$ is initially labeled $y_i = j - 1$ and $Y_{ij} = 0$ otherwise (including unlabeled training data). The propagation part based on label spreading has four steps.

STEP Ⅰ Define the affinity matrix $W \in \mathbb{R}^{n\times n}$ as

$$\begin{array}{ll} i \neq j & W_{ij} = \sigma \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2), \\ i = j & W_{ij} = 0, \end{array} \tag{1}$$

where $\sigma \in \mathbb{R}_+$ is a hyperparameter.

STEP Ⅱ Form matrix $S = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ where D is a diagonal matrix with its (i, i)-element equal to the sum of the i-th row of W.

STEP Ⅲ Set $t = 0$ where t is an iteration index. Iterate $F(t+1) = \rho SF(t) + (1-\rho)Y$ until F converges. ρ is a hyperparameter defined in $(0, 1)$.

STEP Ⅳ Let $F^* = \lim_{t\to\infty} F(t)$.

STEP Ⅰ involves calculating the similarity of the features between $\mathbf{x}_i$ and $\mathbf{x}_j$. When $i \neq j$, W_{ij} becomes large when $\mathbf{x}_i$ is close to $\mathbf{x}_j$. The converse also holds. Therefore, W_{ij} can represent the similarity. In STEP Ⅱ, W is normalized to maintain the interpretations of probability of F. STEP Ⅱ I estimates the probability of being legitimate or malicious. A hyperparameter ρ works as a clamping factor that adjusts the relative amount of its neighbors and initial label information. The smaller the ρ, the higher the importance of the initial label. The converse also holds. In actual data, initial labels, i.e., labeled training data, will sometimes be wrong. If we use this step with incorrect initial labels, incorrect information may explosively propagate. To prevent this, the propagation part permits labels of labeled training data to be partially changed. Furthermore, $F(0)$ is not much of a problem since it converges to F^* after iteration (usually set to $F(0) = Y$). From the above steps, we can predict F^* that denotes the probability of training data being legitimate or malicious.

B. Screening Part

A limitation of SILU is that mislabeled data may occur in the propagation part. This is not limited to SILU but is a general problem with semi-supervised learning methods. To reduce the amount of mislabeled data, we introduce a screening part to remove unreliably estimated labeled data.

In the screening part, SILU selects unlabeled training data whose probability of being legitimate or malicious calculated in the propagation part is high and assigns labels to selected unlabeled training data for use in the supervised learning part; that is, it is equivalent to ignoring unreliable unlabeled training data that are difficult to classify as legitimate or malicious.

Let $q \in \{l+1, \cdots, l+u\}$ be an any unlabeled training data index. We assign a malicious label to unlabeled training data whose index is q when $F^*_{q1} \geq T_m$ (i.e., $F^*_{q2} \leq T_m$) and legitimate label when $F^*_{q1} \leq T_l$ (i.e., $F^*_{q2} \geq T_l$) where T_m and T_l are thresholds. Otherwise, no label is assigned. Note that F^* hold $F^*_{i1} + F^*_{i2} = 1$ for arbitrary i since it is a kind of probability. T_m and T_l are hyperparameters to be determined and work to remove unreliable unlabeled training

data. We define the selected unlabeled training data satisfying the above conditions as D_{U_n}. Finally, we create new labeled training data $D_{new} = D_L \cup D_{U_n}$ by combining the original labeled training data D_L and D_{U_n}. The D_{new} are used in the supervised learning part. Note that if T_m and T_l are set strictly (that is, T_m is close to 1 and T_l is close to 0), the amount of D_{U_n} that can be incorporated into learning in the supervised learning part decreases, but the labeling accuracy improves. Conversely, if T_m and T_l are set loosely (that is, T_m and T_l are close to 0.5), the amount of D_{U_n} increases, but the labeling accuracy worsens.

C. Supervised Learning Part

In the supervised learning part, SILU applies any regular supervised learning methods (e.g., LR, SVM, RF) to D_{new} to train a classifier. In the detecting phase, SILU classifies test data as legitimate or malicious by using the classifier. Note that different features can be selected for the propagation and supervised learning parts. Details are given in section VI.

D. Advantages

The key advantages of SILU compared with other semi-supervised learning methods used in cyber security are:

a) SILU can predict the test data that is not used for unlabeled training data.
b) SILU can be added on to any type of classifier of a supervised learning method.
c) SILU can suppress deterioration of classification performance for test data.
d) In the detection phase, SILU can process in the same time order as the original supervised learning method.

a) Unlike most semi-supervised learning methods used in cyber security, which require retraining every time a new test dataset appears since they use test data as unlabeled training data in training phase, SILU does not require retraining since it can use different datasets for unlabeled training and test data. This helps SOC operation for suppressing detection time. If retraining is required, it takes much time since the classifier must generally learn large-scale data in log analysis.

b) In the supervised learning part, SILU does not specify a specific supervised learning method. Therefore, if there are unlabeled training data in addition to labeled training data, SILU can be used as an add-on to the original supervised learning method.

c) In the screening part, SILU removes some unlabeled training data that have ambiguous probability.

d) In the detection phase, SILU only takes the time required for detection time of the supervised-learning-based classifier in the supervised learning part. Though the detection time is dependent on the number of dimensions of the feature vector in training phase, runtime complexity of detection phase is linear. Therefore, once the model is created in the training phase, the detection phase can be processed at high speed.

V. Datasets

A. Data Source

1) Proxy Logs: Proxy logs were collected in an actual large corporate network located in Japan. Proxy logs include URL, source/destination IP address, source/destination port number, number of source/destination packets, number of request/response bytes, HTTP user agent, HTTP status code, HTTP method, and timestamp. To avoid mixing malicious logs into legitimate logs, we created the legitimate datasets using proxy logs when no incidents were reported by a SOC and malware was not detected by multiple commercial tools installed in different monitoring points from the gateway to endpoint. For the malicious datasets, we collected the latest malware samples from VirusTotal [11] every day and obtained HTTP communication logs through dynamic analysis in a sandbox [12]. All malware samples were confirmed to have different SHA1 when they were first seen. To avoid mixing legitimate logs into malicious logs, we removed some logs whose domains are listed in the Alexa top one million [13]. Note also that we mixed various malware families into the malicious datasets so as not to be biased toward specific malware families. We examined the number of malware families with ESET [14]. As a result, 168 malware families were found to be contained in 12,316 malware samples (1,104,663 logs).

2) NetFlow: NetFlow was collected from an actual Tier1 ISP network located worldwide, e.g., the United States, Europe, and Japan, over a working day in August 2017 by sampling at a rate of 1:10,000. By its nature, NetFlow is always collected by sampling since total flows handled in an ISP are too large and not scaled to collect all of them. NetFlow is bi-directional sequences of packets including specific information such as a source/destination IP address, source/destination port number, byte counts, number of packets, protocol (e.g., TCP and UDP), and timestamp. Note that NetFlow requires preprocessing considering their direction since they are bi-directional traffic logs, though proxy logs are uni-directional. In other words, some NetFlow has the source and destination sides oppositely set for client and server connections. If we input raw bi-directional NetFlow as training data, the accuracy of the classifier will deteriorate because it will not be able to distinguish between the servers and clients. To solve this problem, we replaced source and destination sides for some NetFlow so that a well-known port number (0-1023) was the destination side and a high port number (1024-65535) was the source side. We excluded traffic logs from well-known-port-to-well-known-port and from high-port-to-high-port communication to prevent the classifier from deteriorating.

B. Preparing Datasets

We prepared one labeled training dataset (Dataset L), six unlabeled training datasets (Datasets UP1, UP2, and UP3 for Case 1 and Datasets UN4, UN5, and UN6 for Case 2), one validation dataset (Dataset V), and two test datasets (Datasets T1 and T2). Table I lists the numbers of legitimate hosts/malware samples, the number of logs, and acquisition

TABLE I: Number of hosts/malware samples, number of traffic logs and acquisition date of datasets

	Name	Type	Label	# of hosts / samples	# of logs	Acquisition Date
Labeled Training	L	Proxy	Leg Mal	2500 2500	100,229 60,340	Jul. 2017
Unlabeled Training	UP1	Proxy	-	100	3,257	Aug. 2017
	UP2	Proxy	-	1,000	33,822	
	UP3	Proxy	-	10,000	324,282	
	UN4	NetFlow	-	10,000	212,462	Aug. 2017
	UN5	NetFlow	-	50,000	1,222,605	
	UN6	NetFlow	-	250,000	5,608,194	
Validation	V	Proxy	Leg Mal	1,000 1,000	44,175 18,800	Sept. 2017
Test	T1	Proxy	Leg Mal	1,000 1,000	44,375 19,046	Sept. 2017
	T2	Proxy	Leg Mal	1,000 1,000	44,262 19,537	Sept. 2017

date of these datasets. Note that duplicated logs, i.e., communication from one host to the same URL, were eliminated to improve classification performance within each dataset.

1) Considering Time Series: The validation dataset was prepared for hold-out validation. Cross-validation is generally used for parameter tuning, but we used hold-out validation because we needed to consider time series. Specifically, if we use cross-validation, it is likely to create a classifier by learning malware that was found later in a time series and detect malware that was found earlier, which is not desirable. Therefore, we collected labeled training, unlabeled training, validation, and test datasets in chronological order from the oldest acquisition date, then carried out parameter tuning with Dataset V, and tested with Datasets T1 and T2.

2) Considering the Effects of Imbalance Dataset: Unlabeled training data tend to contain many more legitimate logs than malicious logs in most realistic cases. Therefore, we used an imbalance dataset for unlabeled training datasets and evaluated with performance indexes that can be optimistic on imbalanced datasets. Unlabeled training datasets in Case 1 (Datasets UP1, UP2, and UP3) consisting of proxy logs were created by mixing legitimate and malicious logs so that the ratio of the numbers of legitimate to malicious hosts would be 95:5 and deleting label information. Unlabeled training datasets in Case 2 (Datasets UN4, UN5, and UN6) consisting of NetFlow were created just by using NetFlow as they were.

3) Verifying the Effects of Learning Unlabeled Training Data: This paper assesses whether SILU improves the classification performance as the amount of unlabeled training data is added in subsection VII-B. For the above purpose, unlabeled training datasets were prepared to meet the following conditions: (Dataset UP1) $\subset$ (Dataset UP2) $\subset$ (Dataset UP3), and (Dataset UN4) $\subset$ (Dataset UN5) $\subset$ (Dataset UN6) where $\subset$ means subset.

TABLE II: Mean and standard deviation (SD) of AUC of each feature candidate. x indicates the existing feature.

	Feature Candidate	Proxy logs	NetFlow	Mean AUC (SD)
Word-based Features	**FQDN**	x		**0.9382 (0.0003)**
	TLD	x		**0.8190 (0.0039)**
	Path Element	x		**0.9562 (0.0028)**
	Query String	x		0.6017 (0.0011)
	Query Key	x		**0.8888 (0.0042)**
	Query Parameters	x		0.5000 (0.0001)
	User Agent	x		0.5628 (0.0158)
	Method	x		0.5229 (0.0003)
	Destination IP Address	x	x	**0.8797 (0.0067)**
	Destination Port Numer	x	x	0.6625 (0.0193)
	AS Number	x	x	**0.9173 (0.0013)**
	Country	x	x	**0.8101 (0.0022)**
	City	x	x	**0.8764 (0.0071)**
Statistical Features	URL Length	x		0.6641 (0.0001)
	FQDN Length	x		0.6298 (0.0037)
	Domain Length	x		0.6826 (0.0034)
	Path Length	x		0.6946 (0.0097)
	Query Length	x		0.6683 (0.0100)
	Filename Length	x		**0.7211 (0.0069)**
	Extension Length	x		0.6035 (0.0069)
	# of Numbers in URL	x		0.5950 (0.0040)
	# of Numbers in FQDN	x		0.6025 (0.0009)
	# of Numbers in Path	x		**0.7249 (0.0019)**
	Symbols in Path*	x		0.5966 (0.0020)
	Subdomain in URL*	x		0.6523 (0.0228)
	Extension in URL*	x		0.5930 (0.0065)
	% of English word in Path	x		**0.7366 (0.0012)**

VI. Feature Extraction

To show that SILU is effective even when suitable features are set, we prepared both word-based and statistical features as candidates in Table II. Many of these features have been discussed in previous works [1], [15]–[17]. The Path Element means the element of each word separated by "/" in a URL path. For example, consider the URL "http://www.example.com/RD/index.php." In this case, the Path Elements are "RD" and "index.php." AS number, country and city were resolved with the MaxMind GeoIP Lite database [18].

From our experience, using only statistical features does not achieve high performance. On the other hand, using word-based features in addition to word-based features tends to achieve better performance. Therefore, we prepared both word-based and statistical features. Bartos's results [1] also suggest that using only statistical features did not achieve high performance, e.g., AUC seems less than 0.7.

Word-based features were vectorized with a bag-of-words model. We regarded all unique patterns (e.g., 1.1.1.1 and 2.2.2.2) appearing in each feature (e.g., destination IP address) as one "element" and transformed each log into a feature vector by assigning a Boolean value depending on the presence of an "element": 1 for present and 0 for absent. Statistical features were normalized by dividing by the largest value assumed in the training data. The length of a part of or whole URL is normalized by dividing it by 2083, which is

the maximum number of characters of a URL. Feature vectors marked with * were normalized by assigning 1 for present and 0 for absent.

However, these feature candidates are not all guaranteed to be effective to classify a log as legitimate or malicious. Therefore, it is necessary to consider how much each feature candidate affects the classification results. Adding features that do not greatly affect classification results will require much computation time and may cause overfitting. In this study, we applied LR with L_2 regularization to each feature candidate and selected only the candidates with a high AUC as features. In this feature selection, we used Dataset V as the validation dataset for parameter tuning, and Datasets T1 and T2 as the test datasets. We created a single feature vector for each log and adjusted the hyperparameter of LR to maximize the AUC with Dataset V. Note that we calculated AUC by ignoring logs with no corresponding "element" to a feature candidate, e.g., some URLs do not have queries and some IP addresses do not exist in the GeoIP database. Through the above feature selection, the features in Cases 1 and 2 are selected as follows:

1) Case 1: From the results listed in Table II, we decided to select the features shown in bold for propagation and supervised learning parts.

2) Case 2: We used different features for the propagation and supervised learning parts since NetFlow does not have HTTP-based or statistical features. Note that different features can be selected for the propagation and supervised learning parts. In the propagation part, we used a destination IP address, destination port number, AS number, Country, and City as features, since they are common to both NetFlow and proxy logs. Thus, we predicts the probability that unlabeled NetFlow is legitimate or malicious with labeled proxy logs and unlabeled NetFlow. In the supervised learning part, we used the same features as for Case 1, i.e., features shown in bold in Table II. However, the NetFlow in D_{new} do not have HTTP-based or statistical features. Therefore, we set the features of NetFlow related to HTTP-based and statistical features as zero vectors. In other words, we made good use of all useful features of NetFlow and proxy logs to create a classifier.

VII. Evaluation

A. Experiment Settings

We conducted comparative experiments for two cases, i.e., Cases 1 and 2 introduced in section III, to confirm the superior classification performance of SILU and the effectiveness of learning with unlabeled training data.

1) Comparative Methods: In the experiments, we compared SILU with current semi-supervised and conventional supervised learning methods. We used HL [5] and LLGC [6] as the current semi-supervised learning methods and LR, RF, and SVM with rbf kernel as the conventional supervised learning methods. When implementing HL and LLGC, we adopted the same features as SILU discussed in section VI instead of the features used in the respective original papers to give a fair comparison. We also used test data as unlabeled training data since HL and LLGC require that unlabeled training and test data are the same. Furthermore, we implemented HL by using label spreading, though an unweighted bipartite graph was used in the original paper, because the unweighted bipartite graph of the original paper cannot be applied to a multiple-feature case. Features other than domains are also useful information for accurate classification. On the basis of the above operations, we compared the performances of SILU learning from Dataset L and an unlabeled training dataset (UP1, UP2, or UP3 in Case 1; UN4, UN5, or UN6 in Case 2), HL and LLGC learning from Datasets L and T1 or T2, and LR, RF, and SVM learning from Dataset L.

2) Performance Indexes: We used AUC and $\mathrm{TPR}_{\mathrm{FPR}=0.1\%}$ as the performance indexes. $\mathrm{TPR}_{\mathrm{FPR}=0.1\%}$ is defined as the TPR when adjusting the threshold so that $\mathrm{FPR} = 0.1\%$. The reason for using AUC is that test data tend to be imbalanced in SOC operation, i.e., actual traffic logs are mostly legitimate. Similarly, the reasons for using $\mathrm{TPR}_{\mathrm{FPR}=0.1\%}$ is that a high TPR when the FPR is small is needed to reduce operational cost. We set $\mathrm{FPR} = 0.1\%$ since previous works often compared the classification performance at 0.1% [19].

3) Hyperparameter Tuning: We adjusted the hyperparameters to maximize the AUC with Dataset V and measured the performance indexes with Datasets T1 and T2. SILU has four hyperparameters to be determined in propagation and screening parts: $\sigma \in \{0.1, 0.2, \cdots, 1, 2, \cdots, 10\}$ and $\rho \in \{0.9, 0.99\}$ for the propagation part and $T_m \in \{0.9, 0.99, 0.999\}$ and $T_l \in \{0.1, 0.01, 0.001\}$ for the screening part. In this paper, we set $\sigma = 0.8$, $\rho = 0.99$, $T_m = 0.99$ and $T_l = 0.01$. From our experience, ρ, T_m and T_l are less dependent on test data.

B. Experimental Results

Table III shows the mean AUCs and $\mathrm{TPR}_{\mathrm{FPR}=0.1\%}$ in Cases 1 and 2. SILU-LR, SILU-SVM, and SILU-RF show the case where LR, SVM, and RF are used for the supervised learning part in SILU, respectively. UP1∼UP3 and UN4∼UP6 are unlabeled training data used for training, e.g., the column of UP1/UN4 denotes that SILU is trained with Datasets L and UP1 in Case 1 and with Datasets L and UN4 in Case 2. Note that when applying RF and SILU-RF, we conducted the same experiment five times and calculated the mean performance indexes since RF obtains different results every time.

C. Discussion

1) Effects of learning unlabeled training data: From Table III, comparing the AUCs and $\mathrm{TPR}_{\mathrm{FPR}=0.1\%}$ of LR and SILU-LR, SVM and SILU-SVM, and RF and SILU-RF shows that SILU could successfully learn from both labeled and unlabeled training data and improve classification performance as the amount of unlabeled training data increased. In addition, SILU achieved higher classification performance than the current semi-supervised learning methods, i.e., HL [5] and LLGC [6].

2) Effects of learning unlabeled NetFlow: From Case 2 in Table III, SILU could successfully improve classification performance as the amount of unlabeled NetFlow increased. In other words, it is considered that the classification performance improves since SILU obtained knowledge useful for

TABLE III: Mean AUC and $\mathrm{TPR}_{\mathrm{FPR}=0.1\%}$ in Cases 1 and 2.

	Performance Index	LR	SILU-LR			SVM	SILU-SVM			RF	SILU-RF			HL	LLGC
			UP1/UN4	UP2/UN5	UP3/UN6		UP1/UN4	UP2/UN5	UP3/UN6		UP1/UN4	UP2/UN5	UP3/UN6		
Case 1	AUC	0.99889	0.99897	0.99903	**0.99927**	0.99878	0.99882	0.99891	**0.99921**	0.99878	0.99885	0.99909	**0.99919**	0.99907	0.99817
	$\mathrm{TPR}_{\mathrm{FPR}=0.1\%}$	0.874	0.881	0.895	**0.899**	0.887	0.890	0.895	**0.905**	0.863	0.872	0.897	**0.899**	0.874	0.813
Case 2	AUC	0.99889	0.99925	0.99931	**0.99944**	0.99878	0.99919	0.99925	**0.99937**	0.99878	0.99912	0.99925	**0.99934**	0.99907	0.99817
	$\mathrm{TPR}_{\mathrm{FPR}=0.1\%}$	0.874	0.896	0.907	**0.918**	0.887	0.899	0.901	**0.912**	0.863	0.875	0.899	**0.906**	0.874	0.813

TABLE IV: Confusion matrix of SILU.

Unlabeled Training	TP	FN	FP	TN
UP1	5	0	0	95
UP2	42	1	0	864
UP3	462	16	0	8762

classification from NetFlow, e.g., destination IP address, AS number, country, and city.

3) Labeling accuracy: In general, the accuracy of the classifier tends to deteriorate if the classifier assigns incorrect labels to unlabeled data, but SILU could successfully improve classification performance. The unlabeled training data in Case 1 were created by deleting labels from labeled data. Thus, we investigated the accuracy of labeling after the screening part of SILU in Case 1 as shown in Table IV, which shows the confusion matrix. TP, FN, FP, and TN mean true positive, false negative, false positive, and true negative, respectively. Numbers corresponding TP/FN/FP/TN mean the total number of hosts and malware samples.

4) Imbalanced data: In this paper, unlabeled training datasets in Case 1 were created by mixing legitimate and malicious logs so that the ratio of legitimate to malicious hosts was 95:5. Although details are not shown here due to space considerations, the effectiveness of SILU was also confirmed when that ratio changes from 1:1 to 999:1.

VIII. Conclusion

We introduced SILU to fully utilize useful information from both labeled and unlabeled network traffic logs to generate a versatile and sophisticated automatic log analysis tool, which precisely classifies a host as malware-infected or not. The key advantages of SILU compared with other semi-supervised learning methods in cyber security are that it does not require retraining every time test data changes in the detection phase, can be used as an add-on to any type of supervised-learning-based classifier, and can suppress deterioration of classification performance. Through evaluations, we demonstrated that SILU performs better than current semi-supervised learning methods for cyber security and conventional supervised learning methods and showed the effect of learning with unlabeled training data. In addition, we showed that the performance of SILU improved when using NetFlow as unlabeled training data while classifying proxy logs. These results suggest that SILU can extend detection capability more when different organizations, e.g., SOCs and ISPs, collaborate and share unlabeled data.

References

[1] K. Bartos and M. Sofka, "Optimized invariant representation of network traffic for detecting unseen malware variants," In Proceedings of the 25th USENIX Security Symposium, pp. 807–822, 2016.

[2] M. Antonakakis et al., "From throw-away traffic to bots: detecting the rise of dga-based malware," In Proceedings of the 21th USENIX Security Symposium, pp. 491–506, 2012.

[3] J. Jang, D. Brumley, and S. Venkataraman, "BitShred: feature hashing malware for scalable triage and semantic analysis," In Proceedings of the 18th ACM Conference on Computer and Communications (CCS), pp. 309–320, 2011.

[4] B. Claise, "Cisco systems NetFlow services export version 9," https://tools.ietf.org/html/rfc3954, 2004.

[5] L. Shi, D. Lin, C. V. Fang, and Y. Zhai, "A hybrid learning from multi-behavior for malicious domain detection on enterprise network," In Proceedings of the 2015 IEEE International Conference on Data Mining Workshop (ICDMW), pp. 987–996, 2015.

[6] I. Santos, J. Nieves, and P. G. Bringas, "Semi-supervised learning for unknown malware detection," In Proceedings of the 8th International Symposium on Distributed Computing and Artificial Intelligence (DCAI), pp. 415–422, 2011.

[7] G. Osada, K. Omote, T. Nishide. "Network intrusion detection based on semi-supervised variational auto-encoder," In Proceedings of the 22nd European Symposium on Research in Computer Security (ESORICS), pp. 344-361, 2017.

[8] K. Zhang, C. Li, Y. Wang, X. Zhu, and H. Wang, "Collaborative support vector machine for malware detection," In Proceedings of the International Conference on Computational Science (ICCS), pp. 1682–1691, 2017.

[9] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," In Advances in Neural Information Processing Systems (NIPS), pp. 321–328, 2004.

[10] G. Stringhini, Y. Shen, Y. Han, and X. Zhang, "Marmite: spreading malicious file reputation through download graphs," In Proceedings of the 33rd Annual Computer Security Applications Conference (ACSAC), pp. 91–102, 2017.

[11] VirusTotal, https://www.virustotal.com/.

[12] K. Aoki, T. Yagi, M. Iwamura, and M. Itoh, "Controlling malware http communications in dynamic analysis system using search engine," In Proceedings of the 3rd IEEE International Workshop on Cyberspace Safety and Security (CSS), pp. 1–6, 2011.

[13] AWS, http://s3.amazonaws.com/alexa-static/top-1m.csv.zip.

[14] ESET, https://www.eset.com/.

[15] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Learning to detect malicious URLs," ACM Transactions on Intelligent Systems and Technology (TIST), Vol.2, Issue 3, No.30, 2011.

[16] D. Sahoo, C. Liu, and S. C. Hoi, "Malicious url detection using machine learning: a survey," arXiv preprint arXiv:1701.07179, 2017.

[17] T. Nelms, R. Perdisci, and M. Ahamad, "Execscent: mining for new c&c domains in live networks with adaptive control protocol templates," In Proceedings of the 22th USENIX Security Symposium, pp. 589–604, 2013.

[18] Python, https://pypi.org/project/geoip2/.

[19] Y. Mirsky, T. Doitshman, Y. Elovici, A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," In Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS), 2019.