

Malware Family Classification using Active Learning by Learning

Chin-Wei Chen*, Ching-Hung Su*, Kun-Wei Lee*, Ping-Hao Bair*

*Cybersecurity Technology Institute, Institute for Information Industry, Taiwan, R.O.C

lishchen@iii.org.tw, arivssu@iii.org.tw, kwlee@iii.org.tw, bensonbair@iii.org.tw

Abstract—In the past few years, the malware industry has been thriving. Malware variants among the same malware family shared similar behavioural patterns or signatures reflecting their purpose. We propose an approach that combines support vector machine (SVM) classifiers and active learning by learning (ALBL) techniques to deal with insufficient labeled data in terms of the malware classification tasks. The proposed approach is evaluated with the malware family dataset from Microsoft Malware Classification Challenge (BIG 2015) on Kaggle. The results show that ALBL techniques can effectively boost the performance of our machine learning models and improve the quality of labeled samples.

Keywords—Active Learning, Cybersecurity, Machine Learning, Malicious Traffic Detection, Malware Classification

I. INTRODUCTION

The malware industry has been thriving in recent years. According to the McAfee Threat Center [9], the volume of malware had exceeded 800 million in Q1 2019. Comparing with that in Q4 2018, the number of malware in Q1 2019 was around 65 million, increased by 15 million. Additionally, research conducted by Symantec [3] indicates that the number of malware is increased by 80 percent in 2018 while the increase in the cryptocurrency mining program was among the highest. Symantec [4] also states that they blocked 115K malicious PowerShell scripts monthly on average which is less than 1 percent of all malicious PowerShell scripts. Dealing with such a great deal of malware is a tough task even for cybersecurity companies because some malware variants can avoid getting detected by the detection of anti-virus software. Given the fact that it is relatively harder to use traditional rule-based techniques to detect current threats, it is necessary to find a faster and more effective way to cope with this issue.

The emergence of artificial intelligence has had a profound impact on many fields, including the cybersecurity industry. Many cybersecurity companies design and implement malware detection systems with the help of machine learning and deep learning techniques. For instance, Park S, Gondal I, Kamruzzaman J, et al [12] detect malware variants with machine learning models by using adversarial auto-encoder and semantic hashing. The system can not only detect the malware more effectively but also reduce the misclassification error rate of benign files comparing with traditional classifiers. Good performance of classification models relies on data with decent

label quality. However, data labeling requires labor costs, especially in the field of cybersecurity, since it is relatively difficult to collect labeled malware files from the real world.

To deal with this problem, we present an approach based on active learning by learning techniques with machine learning in training and testing phases. We use malware family classification dataset provided by Microsoft Malware Classification Challenge (BIG 2015) on Kaggle. Our main contributions of the proposed approach are briefly summarized as the following:

- We propose an approach to deal with insufficient labeled data in terms of the malware classification task, which combines the support vector machine (SVM) classifiers and active learning by learning (ALBL) techniques.
- The fact that malware is prone to mutations leads to many challenges in malware classification. For example, it is necessary to adjust the model structure frequently to learn the characteristics of the new type of malware and reduce the misjudgment caused by mutation. ALBL techniques can integrate the expertise of cybersecurity experts into the model so that it can be continuously enhanced and adapted to the diversity of malware.

The following parts of this paper are organized as follows. The related work is introduced in Section II. Section III describes the details of the proposed approach. Section IV introduces the performance and experiments of the proposed approach. Finally, we provide the conclusion and discussion of our work in Section V.

II. RELATED WORK

In the past, cybersecurity experts needed to analyze the malware architecture through static and dynamic analysis to understand the patterns and behaviors of malware. Nowadays, with the help of machine learning techniques, cybersecurity experts can quickly distinguish the types and characteristics of malware [13], [17]. In general, machine learning techniques rely on many labeled samples to identify different malware families, but only a few cybersecurity manufacturers have such ability due to resource limitation.

Moreover, Kaspersky [7] has developed multiple detection tools related to machine learning and deep learning. It established an association model based on a great deal of malware to detect unknown software risks. Kaggle, a well-known data science competition platform, cooperated with

Microsoft and held a malware classification competition in 2015. Participants managed to observe the relationship between malware of different families from *.bytes* format and *.asm* format of malware files and developed many highly accurate prediction models using machine learning and deep learning techniques.

In the field of cybersecurity, the applications of active learning techniques can enhance malware detection techniques [10], [11], [14], training intrusion detection systems [15] and labeling network traffic data [17], etc. The above-mentioned applications possess some characteristics of active learning such as learning, effectiveness, and repeatability. Although active learning techniques can effectively help machine learning models grow and solve the lack of labeled data, it is also challenged by a single learning strategy in the algorithm. The learning strategies are used as the basis for the growth of the model. If the model is not adjusted in time, the learning results would be biased.

To enhance the stability of active learning techniques, Hsu W-N and Lin H-T [5] proposed Active Learning by Learning (ALBL), which is based on the pool-based active learning. The framework of the single learning strategy was changed to multiple learning strategies through the integration of multi-armed bandit theory in learning techniques. Then the model can automatically adjust the learning strategy according to the characteristics of the data structure, pursue the maximum effect in limited resources, thereby enhancing the practicality of the model.

III. PROPOSED APPROACH

This section could be separated to feature engineering, learning algorithms, and modeling. Figure 1 provides the architecture framework of the methodology. We extract features from malware files into *.bytes* and *.asm* format and take these features as input into pre-defined learning algorithms to select feedback data, which is then labeled by cybersecurity experts. By this methodology, the quality of the labeled data, as well as the performance of the models, could be continuously improved and the limitations of lacking in labeled data could be mitigated.

The essence of ALBL techniques is to automatically select samples requiring experts' feedback. According to the architecture, active learning techniques can be divided into two types: stream-based active learning and pool-based active learning. Stream-based active learning selects samples, which meet the predetermined otherwise, it will directly be discarded. Pool-based active learning selects samples with the most information. The ALBL techniques used in this paper belong to the latter one. It automatically selects the most valuable malware from several unlabeled malware. The cybersecurity experts consequently review and provide comments to update the original model.

A. Feature Engineering

Malware files should be decompiled and converted into *.bytes* and *.asm* files before analyzing them using machine learning techniques so as to observe the properties of different

malware families via exploratory data analysis and extract features for classification. Malware files in *.bytes* format could be viewed as a hexadecimal representations of binary files while malware files in *.asm* format are the metadata that contains some important information such as strings and function calls. Figure 2 provides an example of malware file in *.bytes* and *.asm* format which are obtained by using the disassembly tool IDA Pro.

In general, some of the properties of different malware families could be observed in *.bytes* and *.asm* format of malware files. Therefore, in order to effectively classify various kinds of malware families, in addition to some general statistical features such as occurrence and frequency, we utilize other techniques such as natural language processing and data visualization to obtain other informative features.

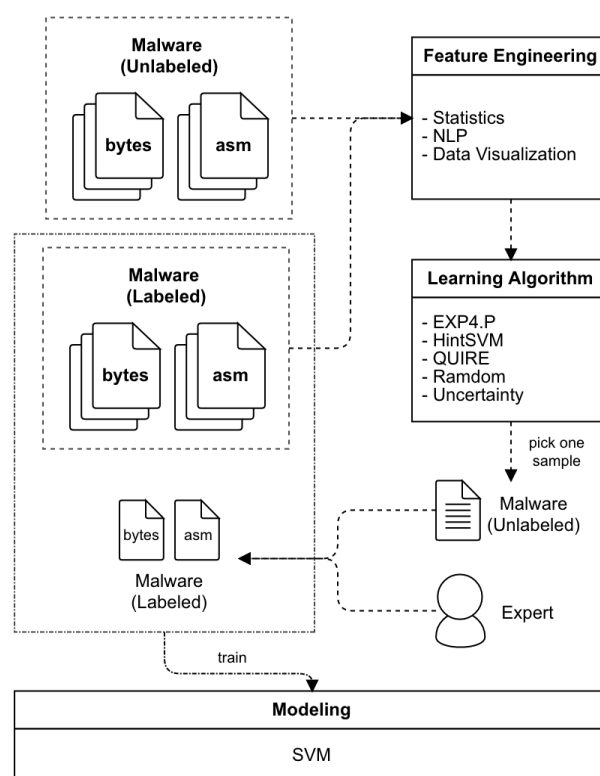


Figure 1. Architecture Framework

1) **Statistics:** In terms of the properties of different malware families, the common instructions of each corresponding family would vary. Hence, we attain some statistical features [18] based on occurrence and frequency of the instructions in *.bytes* and *.asm* files. In addition, to avoid sparse features and reduce feature space, we filter out low frequency instructions to strengthen the effectiveness of features. The features pertaining to instruction occurrence and frequency are as follows:

- Single Byte/API Frequency
- Instruction Count/Proportion
- Symbol Occurrence
- The Frequency of Register Usage
- Predefined Sections Count/Proportion

```

00401000 56 8D 44 24 08 50 8B F1 E8 1C 1B 00 00 C7 06 08
00401010 BB 42 00 8B C6 5E C2 04 00 CC CC CC CC CC CC
00401020 C7 01 08 BB 42 00 E9 26 1C 00 00 CC CC CC CC CC
00401030 56 8B F1 C7 06 08 BB 42 00 E8 13 1C 00 00 F6 44
00401040 24 08 01 74 09 56 E8 6C 1E 00 00 83 C4 04 8B C6
00401050 5E C2 04 00 CC CC CC CC CC CC CC CC CC CC CC CC
00401060 8B 44 24 08 8A 08 8B 54 24 04 88 0A C3 CC CC CC
00401070 8B 44 24 04 8D 50 01 8A 08 40 84 C9 75 F9 2B C2
00401080 C3 CC CC CC CC CC CC CC CC CC CC CC CC CC CC
00401090 8B 44 24 10 8B 4C 24 0C 8B 54 24 08 56 8B 74 24

```

```

.text:004014D2 8B 7C 24 0C      mov     edi, [esp+0Ch]
.text:004014D6 85 FF      test    edi, edi
.text:004014D8 74 2C      jz      short loc_401506
.text:004014DA 8B 71 18    mov     esi, [ecx+18h]
.text:004014DD 8D 41 04    lea     eax, [ecx+4]
.text:004014E0 83 FE 10    cmp     esi, 10h
.text:004014E3 72 04      jb      short loc_4014E9
.text:004014E5 8B 10      mov     edx, [eax]
.text:004014E7 EB 02      jmp     short loc_4014EB

```

Figure 2. An example of malware in bytes and asm format

2) Natural Language Processing:

The instructions of malware vary from one to another according to their purposes; they consequently lead to different relationships between the sequence of instructions. Therefore, natural language processing technique such as n-grams is able to effectively capture some of the characteristics of different malware families. According to Ahmadi M's experiment [1], 1-gram feature stands out as the third place in terms of the feature importance of all features experimented. It is also worth noticing that the effectiveness of identifying malware would be increased with larger N, but at the cost of memory usage.

3) Data Visualization: Given the fact that malware files in .bytes format could be viewed as hexadecimal representation of binary files, we convert hexadecimal to decimal so that the range of value would be 0 to 255. By doing so we can observe these files through grayscale images. Wang X, Liu J and Chen X [19] provides an example of grayscale images representing a file in .bytes and .asm format. It can be observed that the former one is more suitable for this method. Furthermore, we found that different malware families have various kinds of properties as the image representation changes. Ahmadi M, Ulyanov D, Semenov S, et al. [1] observed that the same malware family might share similar image representations. However, it cannot be surely stated that different malware families are entirely different in image representation. Also, interesting figures can be observed in grayscale images of malware.

B. Learning Algorithm

Practically, people try to maximize their expected gain with limited resources. This type of problem is called multi-armed bandit problem. Similarly, suitable machine learning algorithms should be utilized in order to reduce the costs entailed as well as meeting certain conditions. With techniques of ALBL, when it comes to selecting feedback data, typical learning strategies consist of five types.

1) Random: A plain strategy is to randomly select a specific sample from the whole data while the importance of each unlabeled sample is considered equally.

2) Uncertainty: Using SVM classifier, an unlabeled sample with the highest uncertainty among all samples can be selected. To that end, the property of highly uncertain data can be labeled by cybersecurity experts and in turn provide more information which would increase the performance of the model [16].

3) HintSVM: In comparison with SVM, in addition to labeled data, HintSVM [8] takes the property of unlabeled data into account. In other words, the decision boundary of HintSVM would vary considering both uncertainty and representativeness. In Figure 3, solid line only considers uncertainty while dashed line considers both uncertainty and representativeness. We can observe that unlabeled samples with less significant features in top-left cluster would be misclassified if the learning algorithm only considers uncertainty.

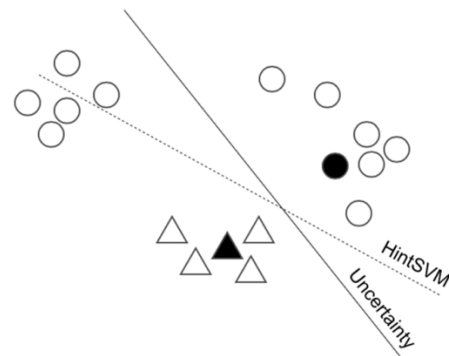


Figure 3. Illustration of the HintSVM

4) QUIRE: Huang S-J, Jin R and Zhou Z-H [6] proposed a learning algorithm, namely Querying Informative and Representative Examples (QUIRE). Similar to HintSVM, it selects samples based on uncertainty and representativeness. However, unlike HintSVM, QUIRE uses an estimator function to represent unlabeled data.

5) EXP4.P: While the four above-mentioned learning algorithms have their own advantages, to combine different learning algorithms and solve the multi-armed bandit problem, we utilize EXP4.P algorithm [2] to select samples with the highest learning value based on all four learning algorithms. The sample is consequently labeled by cybersecurity expert.

C. Modeling

In terms of the learning algorithm, we use SVM, which projects samples into high dimensional feature space and in turn makes decision boundary that separates classes as widely as possible. In addition, SVM could utilize kernel tricks to deal with data that is not linear separable in certain dimensional space.

Then, we combine with ALBL techniques, the number of required labeled samples can be reduced. First, a basic model can be established through a few labeled samples, and then human-machine collaboration can be used to enhance the model performance. With ALBL technique, the classifier will retrain after obtaining the feedback samples labeled by

cybersecurity experts; consequently, it selects other feedback samples to experts. After certain rounds of training, the model is able to predict malware families more effectively.

IV. EXPERIMENT RESULTS

In this section, we describe the effectiveness of ALBL techniques in the field of cybersecurity and conduct research based on malware analysis. The following contains can be divided into experiment data, experiment environment and experiment results.

A. Experiment Data

We used the Microsoft Malware Classification Challenge [20] competition data from Kaggle as our dataset. There were nine malware families (Ramnit, Lollipop, Kelihos_ver3, Vundo, Simda, Tracur, Kelihos_ver1, Obfuscator.ACY and Gatak) in this competition data. The training data and testing data had 10,868 and 10,873 malware samples respectively. Table 1 shows the number of malware families in the training data. The amount of Lollipop and Kelihos_ver3 is the largest and the amount of Simda is the smallest.

TABLE 1. MALWARE FAMILIES IN THE TRAINING DATA

Family Name	Count	Family Name	Count
Ramnit	1541	Tracur	751
Lollipop	2478	Kelihos_ver1	398
Kelihos_ver3	2942	Obfuscator.ACY	1228
Vundo	475	Gatak	1013
Simda	42		

B. Experiment Environment

For the experimental stage, different experimental environments were built on the Google Cloud Platform. Table 2 shows the specifications of the central processing unit (CPU), random access memory (RAM) and hard disk (HDD) used in each stage. In the experimental stage of the feature engineering, we used 1 TB HDD to store the 400GB data and the generated feature files, and the remaining space was used to store the generated feature files.

TABLE 2. SPECIFICATIONS FOR DIFFERENT EXPERIMENT STAGES

Exp. Stage	CPU	RAM	HDD
Feature Engineer	16 vCPUs	128 GB	1 TB
Learning Algorithm Modeling	4 vCPUs	16 GB	30 GB

C. Experiment 1: The effect of ALBL in the classification of malware families

To verify the effectiveness of ALBL techniques in the field of cybersecurity, which can improve the ability of the model by experts' feedbacks when using a small amount of data. We used a small amount of data in the training process, and then utilized ALBL techniques to select samples from other samples to update the model. Finally, we used the scoring mechanism in this Kaggle competition to evaluate the models' performance.

Table 3 shows the data usage of the experiment. The training data can be divided into two parts: one for the training model and the other for the evaluation model. For data used in training process 50 samples were extracted to train an initial model, and 2030 samples were regarded as unlabeled data. The information of the labeled samples would be used as the experts' feedbacks. The remaining 520 samples were used as validation data when the models were trained.

TABLE 3. SPECIFICATIONS FOR DIFFERENT EXPERIMENT STAGES

Data	Count	Description	Usage
Training data	50	Build the initial model	Training
	2030	Iteratively select malware samples and train model by ALBL	Training
		Validation data for training model	
	520	Evaluate the accuracy rate	Training
Testing data	8268	Testing data of Kaggle competition	Evaluation
	10873		

According to the experts' feedbacks, the performance of the model was recorded. The testing data would be predicted after every 100 feedbacks and be uploaded to the Kaggle competition to verify the effectiveness of the models. The results are shown in Table 4. In the first 100 experts' feedbacks, the loss value of the model is greatly reduced, indicating that the usage of the ALBL techniques can effectively improve the model.

Besides, we used the remaining 8268 samples of the training data to evaluate the performance of the final model. The results are shown in Figure 4. It can be found that the accuracy rate of the model is 99.64%, and the incorrect predictions of malware families are mainly Vundo, Simda, and Kelihos_ver1. One of the possible reasons is the characteristics of malware are similar. For example, Vundo and Tracur are both Trojans, while Kelihos_ver1 and Kelihos_ver3 both belong to Backdoor.

TABLE 4. LOGLOSS PER 100 EXPERTS' FEEDBACKS

# of Expert Feedback	Validation Loss	Public Score (Loss)	Private Score (Loss)
100	1.3153	1.3913	1.4172
200	0.1713	0.1621	0.1568
300	0.1017	0.0995	0.0959
400	0.0897	0.0763	0.0733
500	0.0591	0.0595	0.0549
600	0.0499	0.0494	0.0445
700	0.0439	0.0434	0.0394
800	0.0421	0.0411	0.0367
900	0.0427	0.0421	0.0384
1000	0.0384	0.0359	0.0304
1100	0.0368	0.0347	0.0294
1200	0.0348	0.0317	0.0266
1300	0.0344	0.0314	0.0261
1400	0.0334	0.0297	0.0240
1500	0.0341	0.0312	0.0265
1600	0.0336	0.0305	0.0263
1700	0.0323	0.0288	0.0239
1800	0.0316	0.0291	0.0237
1900	0.0319	0.0280	0.0219
2000	0.0312	0.0275	0.0216

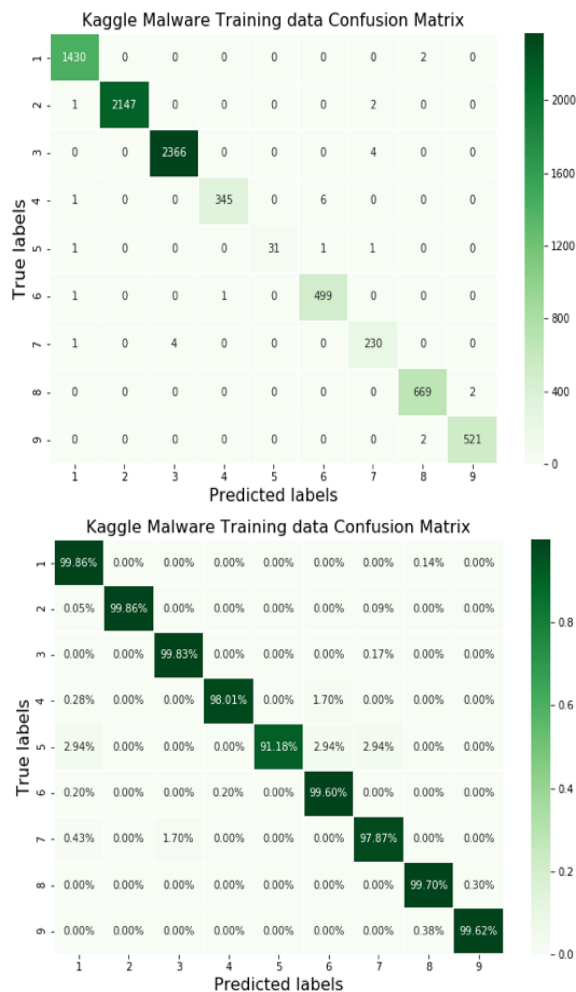


Figure 4. Confusion matrix of predictions with the remaining 8268 training data

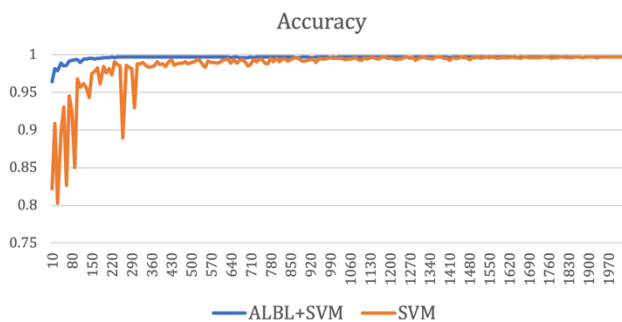


Figure 5. Accuracy comparison for ALBL+SVM and SVM model

D. Experiment 2: Differences before and after using ALBL

The experiment data was the same as that in experiment 1. In experiment 2, we compared the difference between models with ALBL techniques and SVM classifiers. Taking 10 experts' feedbacks for each round, the accuracy rate of the model with ALBL techniques and SVM classifiers is presented in Figure 5. It shows that the performance of the model with ALBL techniques is better than the SVM model from the beginning,

and its growth is relatively stable, especially when the amount of training data is small. During the experiment, the ratio of accuracy rate between the model with ALBL techniques and SVM model can be up to 121%, indicating that the ALBL technology can indeed effectively select samples and learn to achieve the purpose of saving the cost required for labeling.

V. CONCLUSION

Labeled data is often limited in the field of cybersecurity. However, it takes a considerable amount of time to manually label each of the samples. Furthermore, some rules need to be set to select samples which are given to cybersecurity experts. The technique of ALBL can effectively deal with the above scenario. The learning strategy that iteratively selects samples consequently labeled by experts is able to improve the quality of labeled data, and with such it can boost performance of machine learning models.

We propose an approach that uses ALBL techniques with which our models can be adjusted according to experts' feedbacks to deal with insufficient labeled data and reduce classification error rate. Moreover, the model will concentrate on samples with higher uncertainty as a result giving experts some insights on the causes and effects of the data. In our experiments using Kaggle competition data, we utilize ALBL as the core algorithm with which the model is iteratively trained with the appropriate feedbacks from experts. We observe that models with the ALBL techniques have better results and ALBL technique analyzes data in a multi-faceted manner by considering multiple learning strategies at the same time. Therefore, the model has a lower classification error rate and its accuracy rate is faster to converge.

How to apply ALBL techniques to different fields is a question worth considering. If it bridges the gap between theory and practice, people would gain great benefits from it. In our work combine more different training modes such as using the generative adversarial network (GAN) to figure out the problems such as diversity of dataset in the future.

ACKNOWLEDGMENT

This research is supported by industry and institute in Taiwan, and the authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the contributions of this paper.

REFERENCES

- [1] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel Feature Extraction, Selection and Fusion for Effective Malware Family Classification," *Proceedings of the Sixth ACM on Conference on Data and Application Security and Privacy - CODASPY 16*, 2016.
- [2] Alina, Langford, John, Li, Lihong, Reyzin, and R. E., "Contextual Bandit Algorithms with Supervised Learning Guarantees," *arXiv.org*, 27-Oct-2011. [Online]. Available: <https://arxiv.org/abs/1002.4058>. [Accessed: 06-Dec-2019].
- [3] "Executive Summary: 2018 Internet Security Threat Report," Mar-2018. [Online]. Available: <https://www.symantec.com/content/dam/symantec/docs/reports/istr-23-executive-summary-en.pdf>.
- [4] "Executive Summary: 2019 Internet Security Threat Report," Feb-2019. [Online]. Available: <https://www.symantec.com/content/dam/symantec/docs/reports/istr-24-executive-summary-en.pdf>.

- [5] W.-N. Hsu and H.-T. Lin, "Active Learning by Learning," 2015. [Online]. Available: <https://www.csie.ntu.edu.tw/~htlin/paper/doc/aaai15albl.pdf>.
- [6] S.-J. Huang, R. Jin, and Z.-H. Zhou, "Active Learning by Querying Informative and Representative Examples," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 10, pp. 1936–1949, 2014.
- [7] "Machine Learning for Malware Detection," 2019. [Online]. Available: <https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning.pdf>.
- [8] C.-L. Li, C.-S. Ferng, and H.-T. Lin, "Active Learning with Hinted Support Vector Machine," 2012. [Online]. Available: <http://proceedings.mlr.press/v25/li12/li12.pdf>.
- [9] McAfee Labs Threats Report August 2019 (2019) McAfee. Available from: <https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-aug-2019.pdf>.
- [10] N. Nissim, R. Moskovitch, L. Rokach, and Y. Elovici, "Novel active learning methods for enhanced PC malware detection in windows OS," *Expert Systems with Applications*, vol. 41, no. 13, pp. 5843–5857, 2014.
- [11] N. Nissim, A. Cohen, R. Moskovitch, A. Shabtai, M. Edry, O. Bar-Ad, and Y. Elovici, "ALPD: Active Learning Framework for Enhancing the Detection of Malicious PDF Files," *2014 IEEE Joint Intelligence and Security Informatics Conference*, 2014.
- [12] S. Park, I. Gondal, J. Kamruzzaman, and J. Oliver, "Generative Malware Outbreak Detection," 2019. [Online]. Available: https://documents.trendmicro.com/assets/white_papers/GenerativeMalwareOutbreakDetection.pdf.
- [13] K. Sethi, S. K. Chaudhary, B. K. Tripathy, and P. Bera, "A Novel Malware Analysis Framework for Malware Detection and Classification using Machine Learning"
- [14] A. Shabtai, R. Moskovitch, Y. Elovici, and C. Glezer, "Detection of malicious code by applying machine learning classifiers on static features: A state-of-the-art survey," *Information Security Technical Report*, vol. 14, no. 1, pp. 16–29, 2009.
- [15] J. W. Stokes, J. Platt, J. Kravis, and M. Shilman, "ALADIN: Active Learning of Anomalies to Detect Intrusions," *Microsoft Research*, 01-Mar-2008. [Online]. Available: <https://www.microsoft.com/en-us/research/publication/aladin-active-learning-of-anomalies-to-detect-intrusions/>. [Accessed: 06-Dec-2019].
- [16] S. Tong and D. Koller, "Support Vector Machine Active Learning with Applications to Text Classification," Nov-2001. [Online]. Available: <http://www.jmlr.org/papers/volume2/tong01a/tong01a.pdf>.
- [17] J. L. G. Torres, C. A. Catania, and E. Veas, "Active learning approach to label network traffic datasets," *Journal of Information Security and Applications*, vol. 49, p. 102388, 2019.
- [18] D. Ucci, L. Aniello, and R. Baldoni, "Survey of machine learning techniques for malware analysis," *Computers & Security*, vol. 81, pp. 123–147, 2019.
- [19] X. Wang, J. Liu, and X. Chen, "Microsoft MALware Classification Challenge (BIG 2015): First Place Team: Say No to Overfitting," 01-May-2015. [Online]. Available: https://github.com/xiaozhouwang/kaggle_Microsoft_Malware/blob/master/Saynotooverfitting.pdf.
- [20] Ronen, Royi, Radu, Marian, Feuerstein, Ahmadi, Mansour, and Elad, "Microsoft Malware Classification Challenge," *arXiv.org*, 22-Feb-2018. [Online]. Available: <https://arxiv.org/abs/1802.10135>. [Accessed: 06-Dec-2019].



Ching-Wei Chen received his M.S. degree in Department of Computer Science and Information Engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, R.O.C. in 2017. He is currently an engineer in Cybersecurity Technology Institute, Institute for information industry, Taiwan, R.O.C. His research interests include machine learning, data mining, deep learning and security.



Ching-Hung Su received his M.S. degree in Department of Information Management from Tamkang University, Tamsui, Taiwan, R.O.C. in 2018. He is currently an engineer in Cybersecurity Technology Institute, Institute for information industry, Taiwan, R.O.C. His research interests include machine learning, data mining, deep learning and natural language processing.



Kun-Wei Lee received his M.S. degree in statistics from National Chengchi University, Taipei, Taiwan, R.O.C. in 2017. He is currently an engineer in Cybersecurity Technology Institute, Institute for information industry, Taiwan, R.O.C. His research interests include data mining, machine learning, deep learning, generative adversarial network.



Ping Hao Bair received his B.S. degree from National Taiwan Normal University, Taipei, Taiwan, R.O.C. in 2017. He is currently an engineer at Cybersecurity Technology Institute, Institute for Information Industry, Taipei, Taiwan, R.O.C. His research interests include machine learning, deep learning, data mining, and natural language processing.