Cost-Effective Malware Classification Based on Deep Active Learning

Qian Qiang[1,2,3(✉)], Yige Chen[1,2], Yang Hu[4], Tianning Zang[1,2], Mian Cheng[3], Quanbo Pan[1,2], Yu Ding[1,2], and Zisen Qi[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{qiangqian,chenyige,zangtianning,panquanbo,dingyu,qizisen}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] China National Computer Network Emergency Response Technical Team/Coordination Center, Beijing, China
chengmian@cert.org.cn
[4] Haier (Beijing) IC Design Co., Ltd., Beijing, China
huy@haier-ic.com

Abstract. Malware has now grown up to be one of the most important threats to internet security. As the number of malware families has increased rapidly, a malware classification model needs to classify the samples for further analysis. Recent success in deep learning-based malware classification, however heavily relies on the large number of labeled training samples, which may require considerable human effort. In this paper, we propose a novel malware classification framework for the cost issue, which is capable of building a competitive classifier via a limited amount of labeled training instances in an incremental learning manner. A cost-effective sample selection strategy is leveraged to focus expert efforts on labeling samples that are most informative for the classifier. We first convert the malware byte sequences into fixed-size gray-scale images through data visualization. Afterward, based on the strategy designed and oriented towards informative malware acquisition, we select samples through Convolutional Neural Network (ConvNet) to query experts for annotation according to the estimated gradients towards the last linear layer. The updated labeled dataset is then fed into the network for further fine-tuning progressively. To evaluate the capability of our method for acquiring informative malware from a pool of unknown samples, we conduct a series of experiments on a benchmark dataset named BIG 2015. Compared to random selection and other existing high-performance strategies, the proposed system can achieve a promising performance rise cost-effectively with less labeling effort wasted. The effectiveness of sample selection towards different families is also analyzed and further proves the efficiency of labeling cost. Moreover, the initialization methods and the pre-defined number of samples queried are studied for practical implementation.

Supported by National Key Research and Development Project (2020YFB1820102).

F. Li et al. (Eds.): SecureComm 2022, LNICST 462, pp. 212–227, 2023.
https://doi.org/10.1007/978-3-031-25538-0_12

Keywords: Deep active learning · Malware classification · Cost-effective

1 Introduction

Currently, the volume of global threats against business endpoints has increased by more than 10% year-over-year. This emphasizes the importance of developing efficient approaches to analyze as well as classify malicious samples. Malware classification as a fundamental task has been a huge burden for analysts due to its fast-emerging speed, and deep learning (DL) methods have shown impressive performance on related tasks [1]. However, labeling samples, which is crucial for DL, is often a cost-sensitive task since it involves human experts. How to select the most informative samples that can improve the predictive capability of classifiers is an essential question that DL-based methods should focus on. A promising solution is active-learning, noted as AL, a learning protocol where samples can be selected for experts' annotation in a sequential, feedback-driven fashion. The selected samples sent to experts for labeling are defined as query samples by the algorithm. It has a great practical significance to develop a framework combining DL and AL, which can jointly learn features and classifiers from informative samples effectively.

In particular, we propose an AL-based malware classification framework using a ConvNet called cost-effective malware classification (CEMC). Different from the existing malware-related works that utilize machine learning methods, our CEMC leverages AL based on ConvNet. The contributions of our paper can be summarized below:

- We develop a cost-effective malware classification framework based on ConvNet with an AL strategy integrated, which is capable of informative sample selection according to the estimated gradients towards the last linear layer.
- The performance of CEMC is evaluated on a dataset named BIG 2015 in terms of accuracy, precision, recall, and F1-score. The experimental results demonstrate that CEMC outperforms random selection and other active learning strategies in terms of performance and stability.
- The effectiveness of CEMC is analyzed from the family perspective, which further proves that the proposed framework achieves the main goal to select the most informative samples while neglecting the less important ones for relatively higher performance. The impact of initialization and the number of query samples, termed query number is explored as well. Some interesting conclusions are drawn for future implementation.

The rest of this paper is organized as follows. Related works are illustrated in Sect. 2. In Sect. 3, the proposed framework is described. The experiments are elaborated in detail in Sect. 4. Finally, the conclusion of the whole paper and the future work are stated in Sect. 5.

2 Related Works

Traditionally, malware analysis methods can be divided into two main categories including static approaches and dynamic approaches. In static approaches, malware is checked by analyzing its executable binaries or codes without executing [2]. In contrast, dynamic approaches trace the malware process and record the behavior features, such as system calls, registry change, or traffic flows, in a controlled environment such as a virtual environment, simulator, and sandbox [3].

As the above hand-crafted malware analysis approaches need a lot of effort to extract features, researchers have introduced deep learning methods into malware classification tasks to improve efficiency. With the help of deep learning technology, features can be learned automatically and malware classification models can be built without expert knowledge.

Nataraj et al. were the first to start the research based upon digit gray-scale images converted from malware binaries [4]. Since then, classifiers trained on image-based malware data have shown to be very promising in massive research. Coull et al. [5] confirmed the effectiveness of ConvNet for malware classification and tried to find the parts which contribute most to the classification task. Yakura et al. [6] applied ConvNet with attention mechanism to images converted from binary data and generated attention maps for further analysis.

Although DL-based approaches can efficiently obtain ideal classification results, they need large amounts of labeled samples for training. Aiming at improving the existing models by incrementally selecting and annotating the most informative unlabeled samples, active learning (AL) has been well studied in the past few decades. In the AL methods, the model is first initialized with a relatively small set of labeled training samples. Then it is continuously boosted by selecting and pushing some of the most informative samples, which are called query samples, to experts for annotation. The informativeness of a sample is often measured by either the uncertainty of the model about this sample, the expected model change after training on this sample, or how representative the samples are about other unlabeled samples [7]. The key of AL is the design of query strategies and the most common strategies belong to the category called "uncertainty" which considers the most valuable sample is the one with the highest uncertainty.

Inspired by the success of AL, there are researches introduced AL to various kinds of malware tasks since then. The scholars from Microsoft Research Center designed a system called ALADIN which used active learning combined with rare class discovery and uncertainty identification to statistically train a network traffic classifier based on Logistic Regression (LR) [8]. Min Zhao et al. proposed a malware detection framework named RobotDroid using Support Vector Machine (SVM) and an active learning algorithm for smartphones [9]. Nir Nissim et al. designed "Exploitation" based on the SVM classification algorithm using the radial basis function kernel and acquired most probably malicious samples [10]. Bahman Rashidi et al. presented an malicious Android application detection framework based on SVM with 206 features and active learning technologies [11]. Chin-Wei Chen et al. proposed an approach that combines SVM and active learning by learning (ALBL) techniques for malware family classification [12] using statistical features.

As we can see, all the related works used AL strategies based on machine learning models, and most of them are SVM-based frameworks. The strategies designed for machine learning models can not be directly borrowed by DL with guaranteened performance improvement. What's more, compared to DL, machine learning methods are relatively "shallower" and can not handle features with high dimensions and feature engineering is a prerequisite for these methods.

To cope with this issue, it becomes a necessity to study the classification framework combining DL and AL.

3 Cost-Effective Malware Classification

3.1 Framework Overview

Suppose there is a training dataset of m malware families initially with no labels denoted by D. The randomly selected malware set for network initialization is identified as D_0^L. The target of CEMC is to select the most informative query samples for expert labeling and expand the labeled training set D_i^L progressively, where $i \in \{0, 1, ..., T\}$, while the unlabeled pool shrinks accordingly.

Motivated by the insights from a significant amount of previous ConvNet-based malware classification research as well as the recently proposed active learning techniques, i.e., CEAL [13], BADGE [14], we address the above-mentioned issues by AL-based ConvNet.

During the i-th round, our proposed CEMC scrutinizes the samples from the unlabeled data set D_i^U and selects the top q most informative samples, where q represents the query number per round. These samples with newly acquired labels from malware experts are then merged into the labeled dataset D_i^L as the training set for the next $(i+1)$-th round.

Figure 1 illustrates the framework of CEMC, and the details of implementation will be discussed in the following.

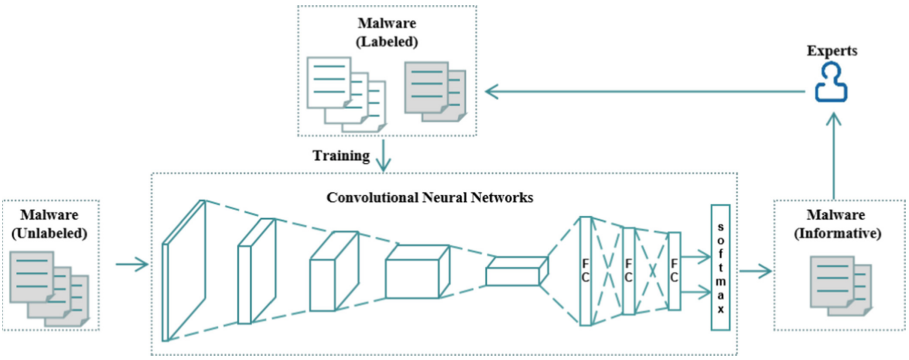


Fig. 1. The overview of the proposed cost-effective malware classification system

3.2 Malware Visualization

This part is considered as the data preprocessing module as well. Inspired by Nataraj et al. [4], gray-scale images of different malware samples appear visually similar from a given family and distinct from those belonging to different families. Based on the observation, the binaries of malware are processed line by line and the corresponding pixels are placed one after another in a row, with a width of 224. The size of the output images is fixed to 224×224 which is a routine size for image-related tasks. All the characters except bytes are ignored including newline characters and line labels. The redundant bytes are discarded and if the bytes cannot fill the entire image, black pixels are padded in the end. After this data preprocessing module, raw malware binaries can be converted into gray-scale images which are suitable for ConvNet to handle for feature extraction.

3.3 Model Initialization

The model can be initialized by D_0^L directly, and this kind of initialization is termed INIT_DIRECT. Additionally, since the raw malware samples have been transformed into gray-scale images, some research has testified transfer learning methods for malware-related tasks that borrowed knowledge from models pre-trained with the dataset of the computer vision area, noted as CV [1,15,16]. Consequently, we can also take advantage of transfer learning ahead of direct initialization and this two-stage initialization method is denoted by INIT_TRANS in the discussion below. For INIT_TRANS, D_0^L is also necessary obviously.

3.4 Model Training and Evaluation

After initialized, the model is fed with unlabeled samples remaining in D referred to as D_0^U. Query sample set X_{info} are selected for experts to annotation and the labeled traning set D_0^L is updated as well as unlabeled pool D_0^U:

$$D_0^L \cup X_{info} \rightarrow D_1^L$$

$$D_0^U \backslash X_{info} \rightarrow D_1^U$$

For the next round, the model is fine-tuned with D_1^L for pre-defined e epochs. After several rounds, the model is gradually fine-tuned for classification tasks. The trained model is evaluated on the testing samples based on four merits, including accuracy, precision, recall, and F1-score.

The strategy to select X_{info} from the unlabeled data pool is essential for the quality of fine-tuned models, which should be designed according to the characteristics of ConvNet and malware.

3.5 Informative Sample Selection

As mentioned before, the labeling cost to train a high-performance DL-based model is troublesome in practice. AL methods designed to select query samples

based on DL-based models can help to deal with this cost issue. The selection should be carried out according to the appearance of the unlabeled samples out of the neural networks, which is the estimation of the real impact with the true labels attached afterward.

Batch Active learning by Diverse Gradient Embeddings (BADGE), which is designed as a practical, general-purpose, label-efficient AL method for deep neural networks has been proposed to offer a solution for the above demand [14]. BADGE creates diverse batches of informative examples. The uncertainty is measured by the gradient magnitude concerning parameters in the final linear layer. Meanwhile, to capture the diversity, a batch of samples is collected whose gradients span a diverse set of directions.

Consider a neural network $f(., W, V)$, where $W = (W_1, ..., W_m) \in \mathbb{R}^{K \times m}$ are the weights of the last linear layer, and V consists of weights of all previous layers. For most of the classifiers, the last nonlinearity is a softmax layer, denoted by $\phi(.)$. Given a malware sample x with label y, the corresponding outputs of the network is $P = f(x, W, V) = \phi(W \cdot Z(x, V))$, where Z maps x to the output of the network's penultimate layer.

Define g_j as the gradient of the cross-entropy loss l_{CE} of x to W_j, according to the definition of cross-entropy loss, g_j can be concluded as:

$$g_j = \frac{\partial l_{CE}}{W_j} = Z(x, V)_j (P_j - I(y = j)) \tag{1}$$

Note that each component of the gradient is a scaling of the corresponding one of $Z(x, V)$, which is the output of the penultimate layer of the network with x as the input. Suppose $\hat{y} = argmax_{i \in m} p_i$, the norm and direction of $g^{\hat{y}}$ can be used to estimate the influence brought by sample x on the current model.

To fulfill the demand for diversity at the same time, k-means++ algorithm is adopted. The algorithm for CEMC is described in Algorithm 1.

Algorithm 1. CEMC: Cost-effective Malware Classification

Require:
Unlabeled pool of samples D_0^U, initialization training set D_0^L, initialized network f with INIT_TRANS or INIT_DIRECT, number of rounds T, query number q and number of epochs e for each round

for $i = 0, 1, ..., T$ **do**
  Train f for e epochs on D_i^L
  for each sample x from D_i^U **do**
    Compute its hypothetical label $\hat{y} = f(x)$
    Compute the gradient g_x based on Equa. 1
  end for
  Generate X_{info} with q samples using k-MEANS++ algorithm on $g_x : x \in D_i^U$
  Label samples in X_{info} by experts
  $D_{i+1}^U = D_i^U \backslash X_{info}$
  $D_{i+1}^L = D_i^L \cup X_{info}$
end for

4 Experiments

4.1 Malware Dataset and Experimental Setup

In this section, we evaluate the performance of CEMC on the benchmark dataset provided by Microsoft Malware Classification Challenge in 2015 which is also referred to as BIG 2015. The dataset has 10868 malware samples in the training set and 10873 in the test set both from 9 families. This is a highly unbalanced dataset in which the largest family called Kelihos_ver3 has 2942 samples and the smallest Simba family has only 42 as shown in Table 1.

Table 1. Family description of BIG 2015 traning set

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
Simba	42		

For the effectiveness of CEMC does not rely on the ConvNet architectures, we use AlexNet [17] as the structure of CEMC.

Following the settings in most DL methods, we randomly select 90% samples of each family to form the unlabeled sample pool, and the rest 10% as the test set in our experiments. It should be noted that after the split, the number of samples from Simba for test is only 4, which will lead to severe fluctuations in performance. To smooth the performance toward this family, the test samples of Simba are increased to 14 on purpose, and the training samples are decreased accordingly. We randomly select 200 samples with labels from the unlabeled pool to initialize the network and the rest are for the incremental learning process.

After several trials, the learning rate of all layers is set to 0.005, and the model is fine-tuned for 50 epochs per round. The fine-tuning is carried out for 30 rounds with 50 query samples selected for each round at first and the performance with different query numbers is also checked to measure the impact induced by query numbers. If not specified, 5-fold strategy is applied and the execution results are averaged as the final result to get rid of the influence of randomness.

To provide persuasive results, we compare CEMC to several baseline strategies borrowed directly from ML-based AL methods which are embedded into ConvNet as well, including:

Random Sampling (RS). This is not an active learning method yet, it is actually the "lower bound" of the selection methods.

LeastConfidence Sampling (LS). An uncertainty-based active learning algorithm that selects top q samples with the smallest predicted class probability, as $max_{i \in m} p_i$, just as Wang et al. [18].

Entropy Sampling (ES). An uncertainty-based strategy that selects the top q samples according to the entropy of the predictive probability distribution, defined as $H(P) = \sum_{i=1}^{m} p_i \ln 1/p_i$ [13] .

KMeans Sampling (KS). This strategy clusters the output of the last linear layer $z = Z(x, V)$ for each unlabeled sample x via K-Means into q clusters and selects the samples nearest to the center of each cluster [7]. KS is a diversity-based strategy the same as the strategy used in this paper.

The above methods share the same ConvNet architecture with CEMC on identical training sets and test sets. The only difference lies in the sample selection criteria which is the target of evaluation.

4.2 Oevrall Performance Comparison

For this comparison, the averaged performance measures are recorded with 50 query samples after INIT_TRANS initialization. The measures of performance after 15/30 rounds of all strategies are listed in Table 2.

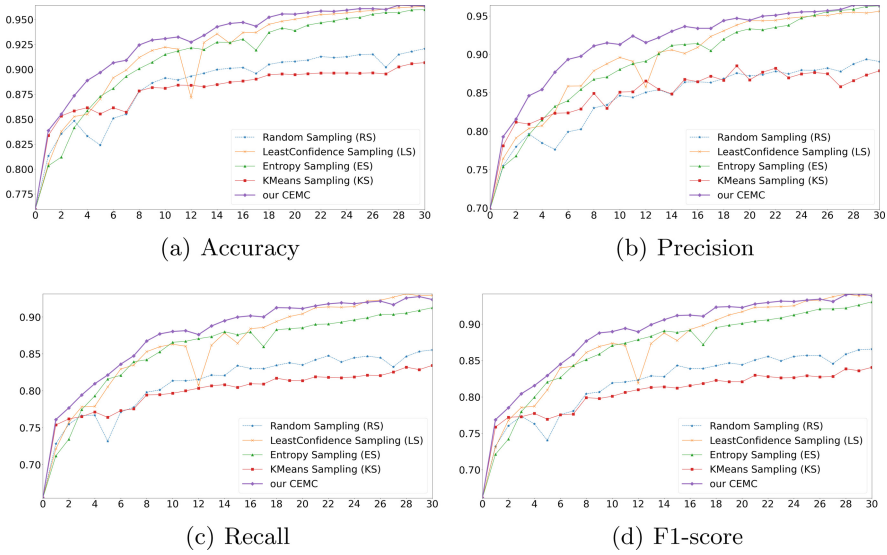
Table 2. Performance after 15- and 30-round fine-tuning of CEMC and baseline strategies

Rounds	Strategy	Accuracy	Precision	Recall	F1-score
15	RS	$90.10 \pm 0.00\%$	$86.38 \pm 0.04\%$	$83.39 \pm 0.05\%$	$84.31 \pm 0.04\%$
	KS	$88.71 \pm 0.02\%$	$86.74 \pm 0.10\%$	$80.44 \pm 0.01\%$	$81.21 \pm 0.02\%$
	ES	$92.70 \pm 0.00\%$	$91.30 \pm 0.06\%$	$87.59 \pm 0.03\%$	$88.86 \pm 0.03\%$
	LS	$92.61 \pm 0.07\%$	$90.11 \pm 0.17\%$	$86.42 \pm 0.16\%$	$87.73 \pm 0.16\%$
	CEMC	$\mathbf{94.60 \pm 0.00\%}$	$\mathbf{93.63 \pm 0.08\%}$	$\mathbf{89.98 \pm 0.01\%}$	$\mathbf{91.17 \pm 0.02\%}$
30	RS	$92.08 \pm 0.00\%$	$89.05 \pm 0.04\%$	$85.53 \pm 0.04\%$	$86.56 \pm 0.05\%$
	KS	$90.67 \pm 0.01\%$	$87.86 \pm 0.16\%$	$83.43 \pm 0.01\%$	$84.06 \pm 0.03\%$
	ES	$95.99 \pm 0.00\%$	$96.30 \pm 0.00\%$	$91.25 \pm 0.00\%$	$93.05 \pm 0.00\%$
	LS	$96.30 \pm 0.00\%$	$95.61 \pm 0.03\%$	$\mathbf{92.94 \pm 0.00\%}$	$\mathbf{93.98 \pm 0.00\%}$
	CEMC	$\mathbf{96.34 \pm 0.00\%}$	$\mathbf{96.36 \pm 0.00\%}$	$92.36 \pm 0.00\%$	$93.90 \pm 0.00\%$

It can be seen that the performance improvement of CEMC compared to the second-best strategy ES is about +2% for all measures after 15 rounds. But after 30 rounds, ES and LS show comparable performance to our CEMC.

For more detail, the performance of CEMC through 30 rounds compared to the other strategies is depicted as learning curves in Fig. 2.

The performance of these 5 methods can be separated into 2 groups in a clear margin. The first group consists of our CEMC, LS and ES, whose performance



(a) Accuracy (b) Precision

(c) Recall (d) F1-score

Fig. 2. Performance comparison through 30 rounds with 50 query samples and INIT_TRANS initialization

outperforms KS and RS in the other group. Moreover, the gap between these two groups has no trend to narrow down along with the increase of labeled samples for more rounds. This gives an assumption that the sample selection strategy is deterministic that once fixed, the upper bound of classification performance is set.

It can also be observed that not until the first 1 or 2 rounds, does CEMC stand out in terms of performance and stability. The closest competitors seem to be LS and ES. LS performs even better in terms of recall and F1-score after about 22 rounds but the difference is so small that can be negligible.

At the end of fine-tuning, the performance of CEMC, LS and ES gradually approach together, which implies the advantage obtained through careful sample selection is compensated by the growing number of labeled samples.

From the above results and analysis, it is clear that our proposed framework CEMC performs consistently better than other methods through fair comparisons and that improvement is the main goal of this study. Additionally, this also indicates that CEMC can effectively select the informative query samples for outstanding performance.

4.3 Family Perspective Performance

In this section, we compare CEMC versus other strategies based on the performance and selection process concerning all families.

First, Fig. 3 presents the confusion matrix of CEMC after 15-round fine-tuning based on one of the 5-fold datasets.

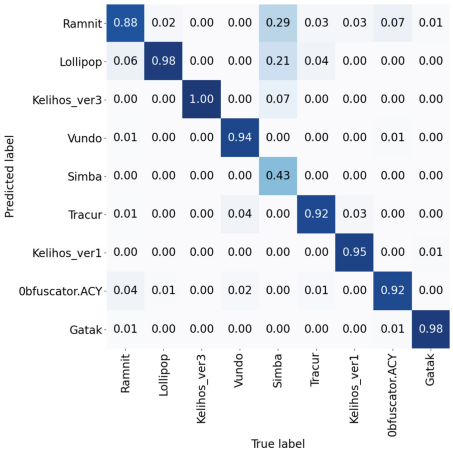


Fig. 3. Confusion matrix of CEMC after 15-round fine-tuning

Meanwhile, Fig. 4 illustrates the confusion matrix of baseline strategies with identical settings for comparison.

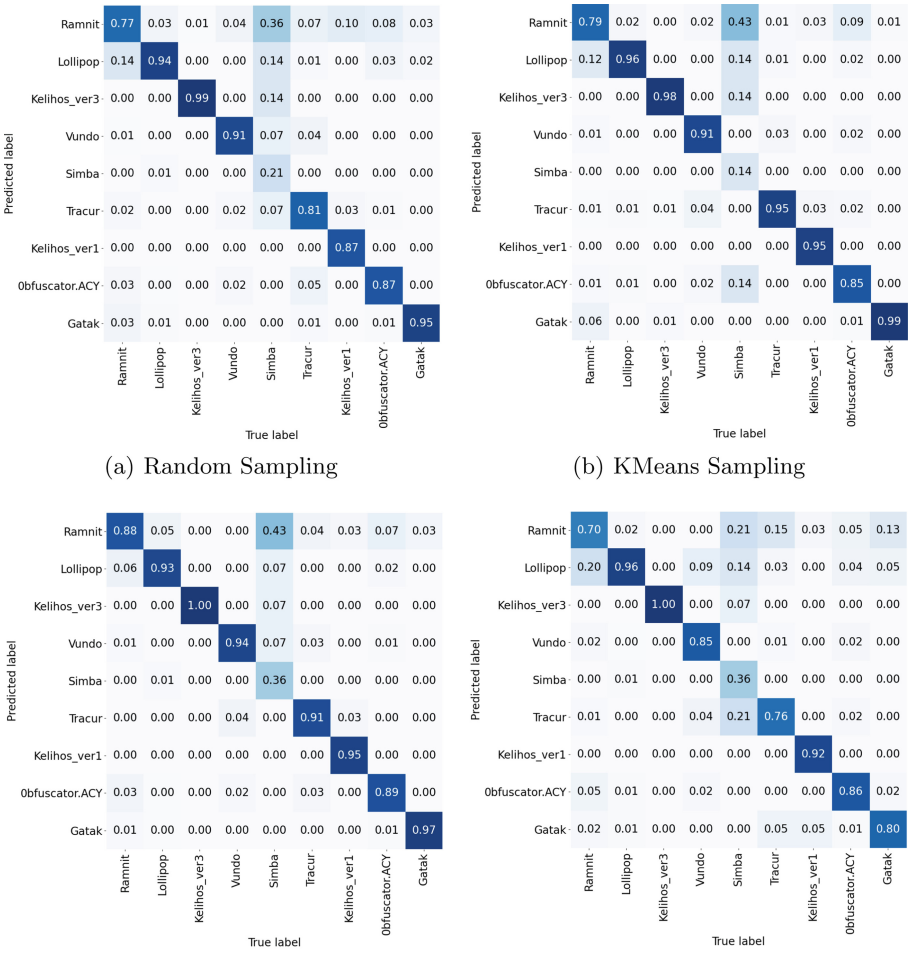
As can be seen, our method obtains the highest accuracy for 7 out of 9 families, while ES and KS both perform best for 3 out of 9. This stable and outstanding performance indicates the effectiveness of CEMC.

However, when it comes to the selection process, the situation is complex and varied for different families. They can be separated into 3 categories, denoted as SAME, MORE, and LESS, based on the number of query samples selected by CEMC compared to RS:

SAME (Lollipop, Obfuscator.ACY, Gatak and Vundo). CEMC selects almost the same number of samples from these 4 families as RS and other strategies as shown in Fig. 5. The performance of CEMC is better or at least almost the same compared to other strategies concerning these four families.

LESS (Kelihos_ver3 and Kelihos_ver1). CEMC selects much less samples from these 2 families than RS as shown in Fig. 6.

All the AL-based strategies perform perfectly with above 95% accuracy on these 2 families, while RS only recognizes 87% of the samples correctly from Kelihos_ver1 in the test set. This gives a signal that there are samples more informative than others in Kelihos_ver1. The model can yield better performance once trained with these samples selected and labeled. And this task is completed by CEMC with the least labeling cost. For Kelihos_ver3, the excellent accuracy obtained by all strategies implies high similarity among samples from this family. Acquiring samples in common is a waste of manual analysis resources and CEMC performs especially outstanding as well as its closest competitors (ES and LS) that they barely query experts for labeling samples from Kelihos_ver3.



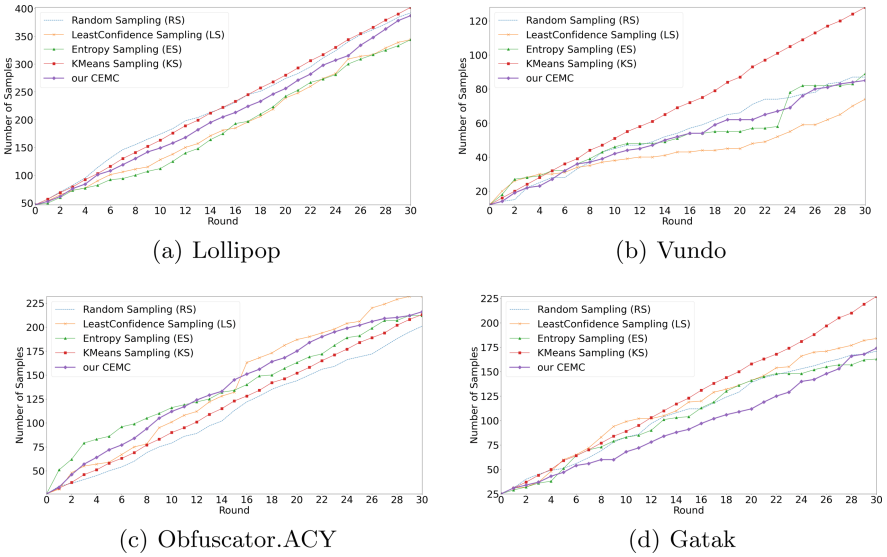
(a) Random Sampling (b) KMeans Sampling

(c) Entropy Sampling (d) LeastConfidence Sampling

Fig. 4. Confusion matrix of baseline strategies after 15-round fine-tuning

MORE (Ramnit, Tracur and Simba). CEMC selects more samples from these families than RS as in Fig. 7. The situation needs to be analyzed individually due to the complexity of situations in this category.

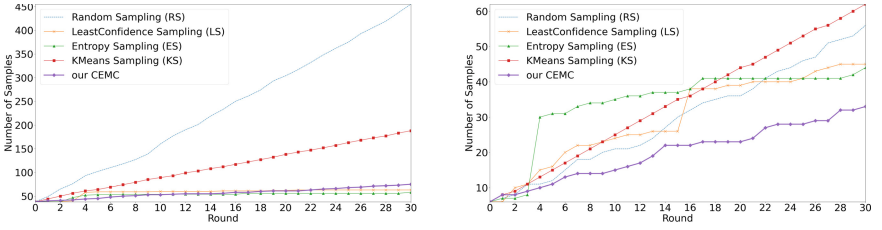
Ramnit has the third most samples among all the families in the unlabeled pool, but the best accuracy achieved by CEMC and ES is only 88% as shown in Fig. 3 and Fig. 4. This phenomenon implies that more informative samples are needed from Ramnit for a better recognition rate. Our CEMC, LS and ES are intended to fulfill this demand and continuously select almost the same amount



(a) Lollipop (b) Vundo

(c) Obfuscator.ACY (d) Gatak

Fig. 5. Number of labeled samples per family in SAME category through 30 rounds



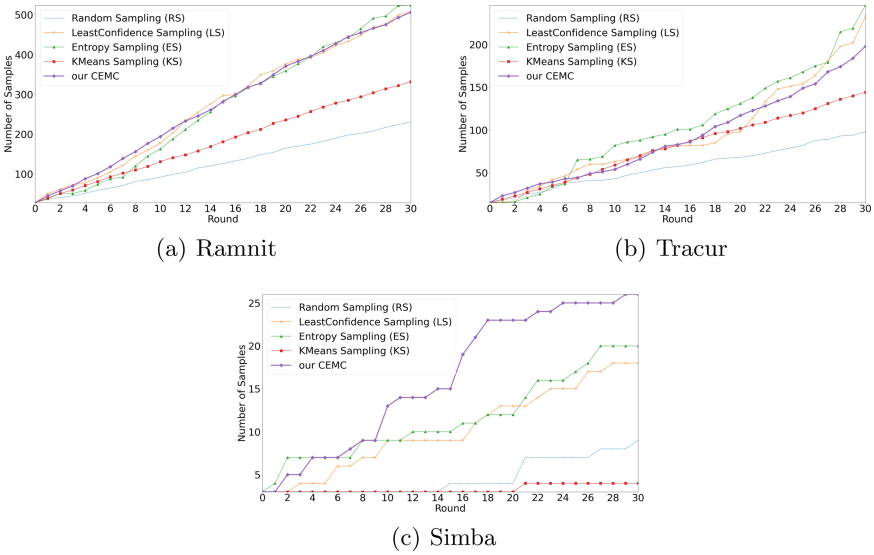
(a) Kelihos_ver3 (b) Kelihos_ver1

Fig. 6. Number of labeled samples per family in LESS category through 30 rounds

of samples through the training process. In contrast, KS selects less informative samples and RS performs the worst.

For Tracur, CEMC outperforms RS and LS, and shows comparable accuracy to KS and ES. LS does not perform well as expected and the accuracy for Tracur is only 76% which is worse than RS.

We leave Simba which has the least samples and worst performance to the last for the interesting result related to this family. There are only 25 samples from Simba in the unlabeled pool after initialization. CEMC succeeds to dig all of these 25 samples out of 9575 candidates, which is like looking for a needle in a haystack. ES and LS managed to select 22 samples, while RS and KS failed to find the informative samples belonging to Simba with crashed performance towards this specific family.



(a) Ramnit (b) Tracur

(c) Simba

Fig. 7. Number of labeled samples per family in MORE category through 30 rounds

We can draw a conclusion based on the above analysis that CEMC yields the best capability of informative sample selection for all families without exception. And this is the main goal of this study.

4.4 Query Number Study

The impact of query numbers as 50/100/200/400 is explored in this section and Fig. 8 shows the corresponding performance of CEMC.

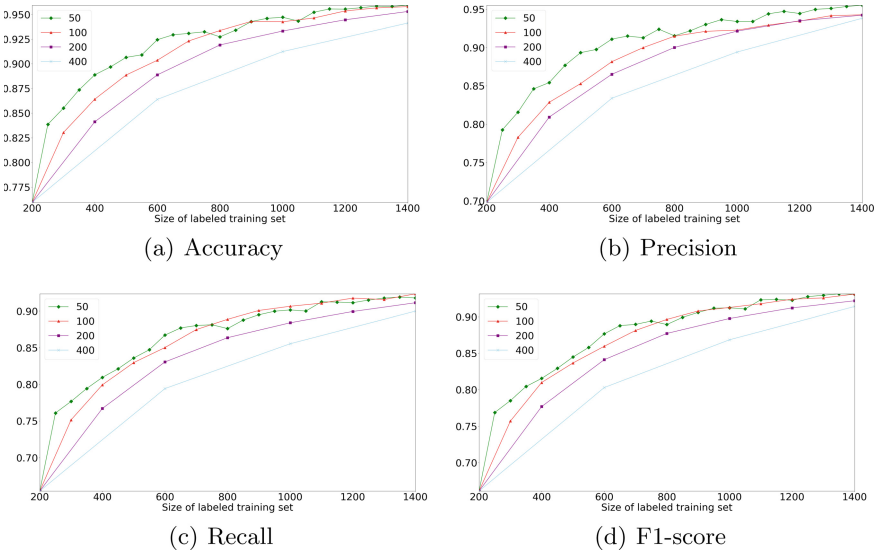
The y-axis represents the number of selected samples instead of rounds.

As can be seen, most of the time, a smaller query number shows an advantage over a larger one. It seems that identifying and acquiring more samples does not benefit the classifier consequently. This phenomenon may be due to the assumption that there exists an optimal set of informative samples and the advantage brought by this set can be disrupted by the samples out of it.

However, it cannot simply state that fewer query samples means better performance. Instead, in order to receive a maximal contribution from the suggested framework, the query number should be carefully tracked and defined. Additionally, it should be noticed that with more labeled training samples, the performance difference is tended to disappear.

4.5 Initialization Analysis

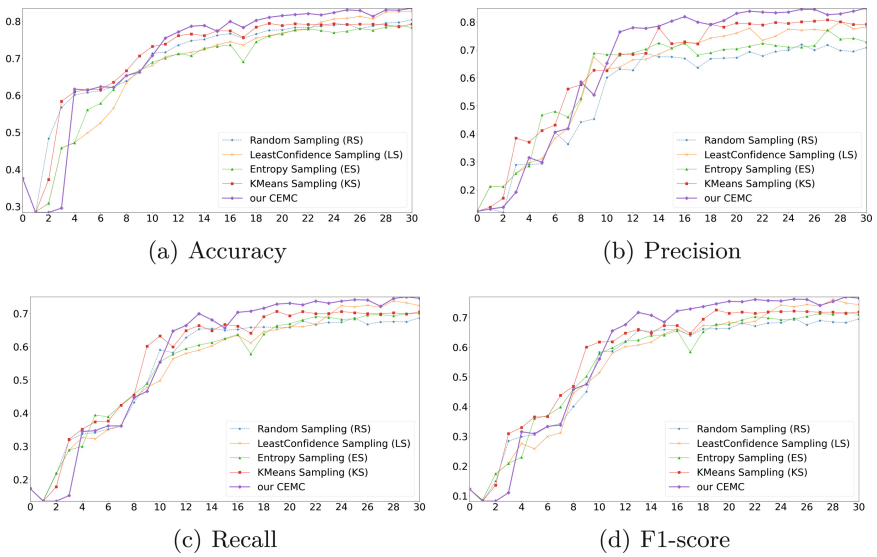
In this section, we want to discover the impact of initialization methods. As mentioned before, there are two kinds of initialization to induce the initial model, INIT_DIRECT and INIT_TRANS. Figure 9 presents the performance of INIT_DIRECT in CEMC versus baseline strategies.



(a) Accuracy (b) Precision

(c) Recall (d) F1-score

Fig. 8. Performance of CEMC with different query numbers with INIT_TRANS initialization



(a) Accuracy (b) Precision

(c) Recall (d) F1-score

Fig. 9. Performance of INIT_DIRECT comparison through 30 rounds with 50 query samples

The performance of CEMC with INIT_DIRECT shows an advantage after almost 10 rounds, which means a longer start-up window and more query samples compared to INIT_TRANS. Additionally, the advantage of CEMC is no longer as obvious as INIT_TRANS, although still can be observed after start-up.

This phenomenon illustrates the importance of high-performance initialization, and this conclusion is reasonable due to the theoretical considerations of all AL-based methods below. AL algorithms select the most informative samples based on the assumption that the prediction can estimate the real contribution introduced by the specific sample with a true label. If the classifier is not accurate enough to provide knowledge consistent with that assumption, the estimation will fail to reflect the real impact from samples and we may miss the real informative ones. Without transfer learning, the model is just initialized using a few labeled samples (200 samples in our paper), and the performance turns out to be very poor (lower than 40%), which will fail to estimate the real impact of most samples.

5 Conclusion

In this paper, we propose a malware classification framework named CEMC, which employs a well-designed, ConvNet-oriented AL strategy. This strategy can progressively select the most informative samples according to the estimated gradients towards the last linear layer. These samples are sent to experts for labeling and integrated into the training set afterward for model fine-tuning.

Comprehensive experiments are conducted to analyze the performance of CEMC versus other strategies on the benchmark dataset BIG 2015. CEMC shows a competitive advantage in the deep learning-based malware classification task and the effectiveness of selecting the most informative samples across all families is justified from the family perspective performance. Moreover, the contribution and impact of initialization and query numbers are discussed thoroughly, based on which some useful advice is given for future implementation.

However, the related works are not enough discussed yet, such as the best size of the informative sample set is not studied in this paper. In future work, we shall optimize the strategy utilized in CEMC and discover more effective strategies for further performance improvement with ConvNet as the classification algorithm. Additionally, there is no research that combines AL and RNN-based networks so far. This is also a subject for future exploration.

References

1. Vasan, D., Alazab, M., Wassan, S., Safaei, B., Zheng, Q.: Image-based malware classification using ensemble of CNN architectures (IMCEC). Comput. Secur. **92**(March), 101748 (2020)
2. Gibert, D., Mateu, C., Planes, J., Vicens, R.: Classification of malware by using structural entropy on convolutional neural networks. In: Proceedings of the 30th Innovative Applications of Artificial Intelligence Conference, IAAI 2018, pp. 7759–7764 (2018)
3. Huang, Y.-T., Chen, Y.-Y., Yang, C.-C., Sun, Y., Hsiao, S.-W., Chen, M.C.: Tagging malware intentions by using attention-based sequence-to-sequence neural network. In: Jang-Jaccard, J., Guo, F. (eds.) ACISP 2019. LNCS, vol. 11547, pp. 660–668. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21548-4_38

4. Nataraj, L., Karthikeyan, S., Jacob, G., Manjunath, B.S.: Malware images: visualization and automatic classification. In: ACM International Conference Proceeding Series (2011)
5. Coull, S.E., Gardner, C.: Activation analysis of a byte-based deep neural network for malware classification. In: Proceedings - 2019 IEEE Symposium on Security and Privacy Workshops, SPW 2019, pp. 21–27 (2019). https://doi.org/10.1109/SPW.2019.00017
6. Yakura, H., Shinozaki, S., Nishimura, R., Oyama, Y., Sakuma, J.: Malware analysis of imaged binary samples by convolutional neural network with attention mechanism. In: CODASPY 2018 - Proceedings of the 8th ACM Conference on Data and Application Security and Privacy 2018-January (3), pp. 127–134 (2018). https://doi.org/10.1145/3176258.3176335
7. Zhdanov, F.: Diverse mini-batch active learning (2019). http://arxiv.org/abs/1901.05954
8. Stokes, J.W., Platt, J.C., Kravis, J.: ALADIN: Active Learning of Anomalies to Detect Intrusion. Microsoft (2008)
9. Zhao, M., Zhang, T., Ge, F., Yuan, Z.: RobotDroid: a lightweight malware detection framework on smartphones. J. Networks **7**(4), 715–722 (2012)
10. Nissim, N., Moskovitch, R., Rokach, L., Elovici, Y.: Novel active learning methods for enhanced PC malware detection in windows OS. Expert Syst. Appl. **41**(13), 5843–5857 (2014)
11. Rashidi, B., Fung, C., Bertino, E.: Android malicious application detection using support vector machine and active learning. In: 2017 13th International Conference on Network and Service Management, CNSM 2017, 1–9 January 2018 (2017)
12. Chen, C.W., Su, C.H., Lee, K.W., Bair, P.H.: Malware family classification using active learning by learning. In: International Conference on Advanced Communication Technology, ICACT 2020, pp. 590–595 (2020)
13. Wang, K., Zhang, D., Li, Y., Zhang, R., Lin, L.: Colloids and surfaces a: physicochemical and engineering aspects **12**(d), 1–13 (2016)
14. Ash, J.T., Zhang, C., Krishnamurthy, A., Langford, J., Agarwal, A.: Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds (2019)
15. Lo, W.W., Yang, X., Wang, Y.: An xception convolutional neural network for malware classification with transfer learning. In: 2019 10th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2019 - Proceedings and Workshop, pp. 1–5 (2019). https://doi.org/10.1109/NTMS.2019.8763852
16. Bhodia, N., Prajapati, P., Di Troia, F., Stamp, M.: Transfer learning for image-based malware classification. In: ICISSP 2019 - Proceedings of the 5th International Conference on Information Systems Security and Privacy, pp. 719–726 (2019)
17. Krizhevsky, A., Sutskever, I., Hinton, G.: ImageNet classification with deep convolutional neural networks. In: Neural Information Processing Systems vol. 25 (2012). https://doi.org/10.1145/3065386
18. Wang, D., Shang, Y.: A new active labeling method for deep learning, pp. 112–119 (2014). https://doi.org/10.1109/IJCNN.2014.6889457