An Efficient Unsupervised Domain Adaptation Deep Learning Model for Unknown Malware Detection

Fangwei Wang[1,2], Guofang Chai[2], Qingru Li[1,2], and Changguang Wang[1,2(✉)]

[1] Key Lab of Network and Information Security of Hebei Province, Hebei Normal University, Shijiazhuang 050024, China
{fw_wang,wangcg}@hebtu.edu.cn

[2] College of Computer and Cyberspace Security, Hebei Normal University, Shijiazhuang 050024, China

Abstract. Emerging malware and zero-day vulnerabilities present new challenges to malware detection. Currently, numerous proposed malware detection approaches are based on supervised learning. However, these methods rely on a large amount of labeled data, which is usually difficult to obtain. Moreover, since the newly emerging malware has a different data distribution from the original training samples, the detection performance of the model will degrade when facing new malware. To solve the problems mentioned above, this paper proposes an unsupervised domain adaptation-based malware detection method to align the joint distribution of known malware and unknown malware. First, the distribution divergence between the source and target domain is minimized by adversarial learning to learn shared feature representations. Second, to further obtain semantic information of unlabeled target domain data, this paper reduces the class-level distribution divergence by aligning the class centers of labeled source data and pseudo-labeled target data. To improve the ability of the model for extracting feature information, this paper mainly uses a residual network with a self-attention mechanism as a pre-trained model. Extensive experiments are conducted on two datasets. The experimental results illustrate that the proposed method outperforms the state-of-art detection methods with an accuracy of 95.63% and a recall of 95.30% in detecting unknown malware.

Keywords: Deep transfer learning · Malware detection · Domain adaptation · Self-attention module

1 Introduction

With the development of 5G technology and big data, IoT is changing every aspect of our daily lives. However, while IoT brings convenience to our lives, it is also facing many malware attacks [1]. Zscaler reported that more than 575 million device transactions and 300,000 IoT-specific malware attacks were blocked during two weeks in December

W. Shi et al. (Eds.): SPNCE 2021, LNICST 423, pp. 64–76, 2022.
https://doi.org/10.1007/978-3-030-96791-8_5

2020, a 700% increase by the pre-epidemic. These attacks targeted 553 different types of devices, including printers, digital signage, and smart TVs.

Malware is one of the most common security risks to the Internet. They are massive and have complex polymorphisms. According to the AV-TEST organization, thousands of malware are born every day. Up to 2021, the total of malware has increased to more than 1.2 billion, which is an increase of 1800% compared to that of 2011. So the complexity of the IoT hardware and software environment provides more attack opportunities for attackers. Moreover, malware poses a huge threat to all our industries. Therefore, how to detect malware quickly and accurately has become one of the most important topics of current research.

Recently, machine learning is being used to improve the accuracy of detecting malware. Supervised malware detection [2] requires a large amount of labeled data to train the model. Moreover, when that model encounters a new unknown sample, the detection performance will decline. Transfer learning [3] applies the knowledge gained from solving one problem to another different but related problem. Fine-tuning [4], a branch of transfer learning, focuses on how to finely tune the model by labeled data. However, fine-tuning does not take into account the discrepancy of data distribution among different datasets. So it still does not achieve excellent results for detecting unknown malware.

All of the aforementioned methods have obvious disadvantages when detecting unknown malware. To improve the detection performance of unknown malware, this paper proposes a detection method based on unsupervised domain adaptation. The unsupervised domain adaptation uses the labeled source domain data to predict the label of target domain data under the condition that the source and target domains have different distribution [5]. Therefore, we use the malware dataset with labels as the source domain and the malware without labels as the target domain. We then transfer the knowledge learned from the source domain to the target domain through domain adaptation to detect unknown malware. The main contributions of this paper are summarized as follows:

1) We use a deep residual network with a self-attention mechanism and a discriminator network to extract features from multi-channels.
2) We adopt the joint distribution alignment approach to reduce the distribution discrepancy. Firstly, inter-domain distribution discrepancy is reduced by adversarial learning. Then, class-level alignment can be achieved by optimizing the semantic alignment loss functions. Eventually, we can achieve intra-class sample compactness and inter-class sample separation.
3) By our proposed method, we perform massive experiments on two datasets. The results show that the method can correctly classify unknown malware and outperforms the state-of-the-art detection methods.

The remainder of this paper is organized as follows: Sect. 2 describes the related work. Section 3 presents the background knowledge and the proposed method. Section 4 describes the experimental details and the related results. Finally, Sect. 5 concludes this paper.

2 Related Work

2.1 Machine Learning-Based Malware Detection

Machine learning has been widely used in the field of malware detection and is divided into two major categories: supervised learning and unsupervised learning. Supervised malware detection methods depend on large amounts of labeled data to train the model. The most common method is to convert malware raw files into grayscale images and train convolutional neural network models to detect the malware. Nataraj et al. firstly transformed malware into images to extract their GIST features and used K-nearest neighbor classification [6]. Jinpei et al. used deep neural networks to automatically extract features to reduce the expense of feature engineering [7]. Unsupervised malware classification, on the contrary, clusters samples based on their similarity [8]. Pitolli et al. proposed an online clustering algorithm to identify malware families [9]. The algorithm classified malware among existing families when running and can identify new families. However, all of these supervised learning methods rely on labeled data to train the model. When facing unknown samples, the detection capability of the model will drop. All these unsupervised detection methods must depend on a large amount of data to reach high accuracy.

2.2 Transfer Learning-Based Malware Detection

Currently, transfer learning has been extensively applied to various fields [10–15]. For example, image classification [10], semantic segmentation [12], robot recognition [15], and medical areas [13]. Transfer learning has also been utilized to detect malware. Vasan et al. proposed an image-based malware classification [16]. They improved the classification accuracy by fine-tuning the neural network as well as data augmentation. Besides fine-tuning, domain adaptation is also a branch of transfer learning. Bartos et al. proposed a new method that computed domain-invariant feature representations from network traffic and learned to identify malicious behavior [17]. Li et al. proposed a method of aligning the feature distributions of different domains [18]. This method can reduce the discrepancy of marginal distribution between the source domain and target domain. However, it does not consider the difference in class-level distribution in the source domain and target domain. Rong et al. used deep transfer learning to detect unknown malware variants [3]. The author transformed malware traffic data into RGB images and proposed a TransNet framework to solve the problem of data distribution discrepancy among different datasets. They replaced the batch normalization layer with a transfer batch normalization layer in the deep neural network to achieve data distribution alignment. However, they did not consider the class-level alignment of different domains.

In the area of computer vision, there are two main approaches to achieve global domain alignment: non-adversarial domain alignment and adversarial domain alignment. Non-adversarial domain alignment minimizes the global distribution discrepancy of domains by different metrics such as Maximum Mean Discrepancy (MMD) [19], KL [20], CORAL [21], etc. In contrast, adversarial domain adaptation methods use the

adversarial learning idea from Generative Adversarial Networks (GAN) to learn domain invariant features. Currently, domain adaptation methods used for malware classification have two main disadvantages: some do not consider the joint distribution alignment between different domains, and some use raw PE files to extract features increasing feature engineering cost and domain expert knowledge. Inspired by the application of domain adaptation in computer vision, we convert malware files into gray images and narrow down the distribution discrepancy between source and target domains by domain adaption.

3 Proposed Methodology

3.1 Overview

The unsupervised domain adaptation aims to improve the model generalization ability in the target domain by minimizing the distribution discrepancy between the source and target domains. In this paper, we use $D_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}, D_t = \{(x_j^t\}_{j=1}^{n_t}$ to denote the source with labels and target domains without labels, respectively, where n_s is the number of samples in the source domain; y_i^s is the label corresponding to the source domain sample x_i^s. D_s and D_t have a different distribution. In this paper, our goal is to transfer knowledge from a labeled source domain to an unlabeled target domain and train a transferable target classifier to predict the labels of the target domain samples.

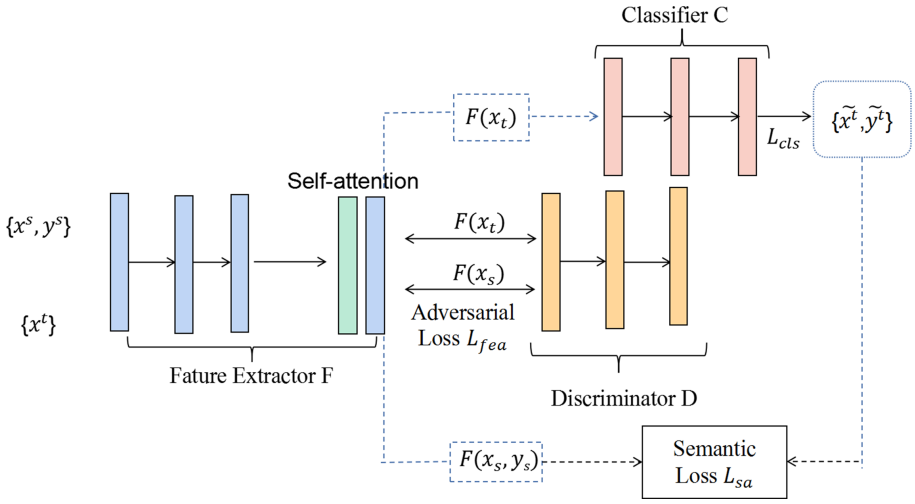

Fig. 1. Overview of the model structure.

The overall framework of the approach in this paper is shown in Fig. 1. It mainly consists of three components: feature extractor F_φ, classifier C_ϕ, and domain discriminators

D_ω, where φ, ϕ, ω are learnable parameters. The feature extractor and domain discriminator achieve global alignment by adversarial learning using labeled source domain, unlabeled target domain. To make the intra-class data more compact and inter-class data more separated, class-center alignment is achieved by source domain labeled data and target domain data with pseudo-labels. Therefore, the model needs to jointly optimize supervised classification loss L_{cls} and global domain adversarial loss L_{fea} as well as class-level semantic alignment loss functions L_{sa}. The overall optimization objectives are as follows:

$$L = L_{cls} + \alpha L_{fea} + \beta L_{sa}, \tag{1}$$

where the hyperparameters α, β are the influence factors of global alignment and semantic alignment, respectively. The supervised classification uses the Cross Entropy loss function to measure the classification error:

$$L_{cls} = E_{(x^s, y^s)\sim D_s}\left(C_\phi\left(F_\varphi(x)\right), y^s\right). \tag{2}$$

3.2 Self-attention Module

We insert a self-attention module [22] in the feature extractor. This module enables each pixel to associate with other pixels. As a result, it can solve the long-distance dependence problem in common convolutional structures. Moreover, this module achieves a better balance between improving the perceptual field and reducing the number of parameters. So as a complementary for convolution, we introduce a self-attention module to get long-term, multi-level dependencies across the image region, as shown in Fig. 2.

The workflow of the Self-Attention module is as follows. Firstly, we can get the feature map x from the first few layers of the residual network by convolution. Three feature maps $q(x)$, $k(x)$, $v(x)$ are obtained by three 1×1 convolutions, respectively. In which the feature-map dimensions of $q(x)$ and $k(x)$ remain the same, and only the number of channels is changed, and $v(x)$ keeps the output channels unchanged. Then, transpose $q(x)$ and multiply it with $k(x)$. The attention map of $[H \times W, H \times W]$ is obtained by normalizing each row by Softmax. So the model pays attention to the i-th, location when synthesizing the j-th region. Thus, we get the attention map:

$$\rho_{j,i} = softmax\left(q(x_i)^T k(x_j)\right). \tag{3}$$

Multiplying the attention map $\rho_{j,i}$ by $v(x)$, we can obtain a feature map $[H \times W, C]$. After a 1×1 convolution h, the output is then reconstructed as $[H \times W \times C]$, and the feature map O is obtained. For a feature map with N location in C channels, the output of the attention layer is $O = (o_1, o_2, \ldots, o_N) \in \mathbb{R}^{C \times N}$, and o is calculated by the following:

$$o = v \otimes \left(\sum\nolimits_{i=1}^{N} \rho_{j,i} h(x)\right). \tag{4}$$

In addition, we further multiply the output of the attention layer by a parameter θ which is learned in the self-attention layer and initialize it to 0, indicating that the self-attention module has not worked yet. As training proceeds, the network slowly begins to learn more features by the residual network with the self-attention module. Thus, the final output is:

$$f = \theta o_i + x_i. \tag{5}$$

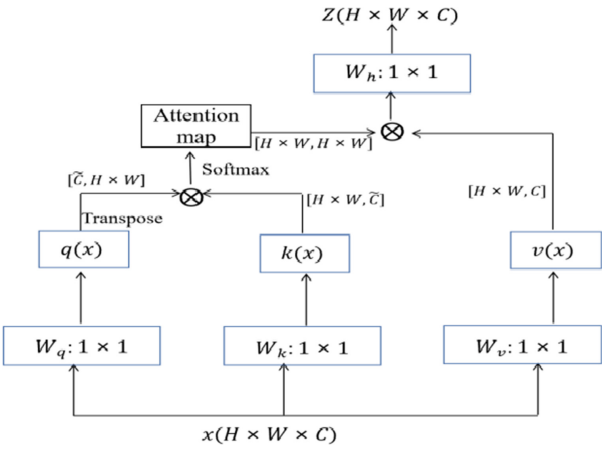

Fig. 2. The workflow of the self-attention module.

We place the self-attentive mechanism at the fourth block of the residual network due to two reasons. Firstly, we extract effectively local features and reduce the resolution by convolution. Secondly, the self-attention module is used to aggregate the global information of the feature. Therefore, the model ability to extract features is improved.

3.3 Global Domain Alignment

The global domain alignment aims to reduce the discrepancy in cross-domain feature distribution. In this paper, we align the feature distributions of two domains by minimizing the global domain adversarial loss function L_{fea} in an adversarial learning manner. The method employs the idea of generative adversarial networks, in which the feature extractor F_φ plays with the domain discriminator D_ω during the training process. The domain discriminator D_ω is trained to minimize the domain loss L_{fea} to distinguish between source and target domain features while the feature extractor F_φ obfuscates the domain discriminator D_ω by maximizing the domain loss to learn a domain-variant feature representation. When the training ends, the network can obtain domain-variant feature representation. The global domain-aligned adversarial training loss can be expressed as follows:

$$L_{fea}\left(F_\varphi, D_\omega\right) = -E_{x\sim D_t} log\left[1 - D_\omega\left(F_\varphi(x_s)\right)\right] - E_{x\sim D_s}\left[D_\omega\left(F_\varphi(x_t)\right)\right]. \tag{6}$$

3.4 Semantic Alignment

Global alignment alone does not achieve precise alignment. To make the model have higher classification ability and learn more differentiated depth features, we propose class-center discriminable feature learning. By this way, we can make similar data more compact and dissimilar data more discriminable in the feature space. Weston et al. [23] proposed that similar samples should be closer to each other and different samples should be further away from each other by judging the manifolds of the samples in the feature space. Wen et al. [24] measured the similarity of samples by the minimum distance between each sample and its corresponding class center. In this paper, we measure the distance between samples by the similarity of labels. Considering the high computational overhead of calculating the distance between each pair of samples, we calculate the distance from each sample to its class center through the class center loss function motivated by [24]. Then the semantic learning loss function L_{sa} is defined as follows:

$$L_{sa} = \sum_{i=1}^{n_s} max\big(0, \|x_i^s - c_{y_i}\| - r_1\big) + \gamma \sum_{i,j=1}^{m} max\big(0, r_2 - \|c_i - c_j\|\big), \tag{7}$$

where γ is a trade-off parameter, n_s is the number of samples in a batch. m is the number of classes. c_{y_i} is the class center and $y_i \in \{1, 2, \cdots, m\}$. r_1, r_2 are two thresholds. To update the class center, each epoch is based on a min-batch rather than all samples. Therefore, during the process of updating class centers in a mini-batch, if there is a sample corresponding to the class center of the batch, the class center is updated. otherwise, it is not updated. In each iteration, the class center is updated according to the following way:

$$\Delta c_j = \begin{cases} \frac{\sum_{i=1}^{n} c_j - x_i}{1+n}, & y_i = j, \\ 0, & y_i \neq j, \end{cases} \tag{8}$$

$$c_j^{t+1} = c_j^t - \Delta c_j^t, \tag{9}$$

where n is the number of samples in each batch and ε is the learning rate. In class-center alignment, we need to get the labels of the target domain data, so we use the pseudo-labels. We get the pseudo labels by the following steps. Firstly, we train the model with the labeled source domain data. Then, the trained model is used to predict the labels of the target domain. The detailed process is: 1) Define $\left\{p_c\left(x_i^t\right)|_{c=1}^m\right\}$ as the output probability of the classifier. $p_c\left(x_i^t\right)$ indicates the probability that x_i^t belongs to the c-th class. m is the number of samples. 2) The pseudo-label corresponding to sample x_i^t is calculated according to $\tilde{y}_i^t = argmax_c p_c(x_i^t)$. 3) The label of the class with the highest probability is selected as the class label of x_i^t. We achieve intra-class compactness and inter-class separation by minimizing the distance between the samples and the corresponding class centers.

3.5 Model Training

The whole training process of the proposed method is listed in Algorithm1. Firstly, we minimize the classification loss by standard supervised learning using labeled source domain samples, so we get a pre-trained feature extractor F_φ and classifier C_ϕ. Second, the pre-trained model is used to assign pseudo-labels to the unlabeled target domain data according to $\tilde{y}_i^t = argmax_c p_c(x_i^t)$. Then, we use the pre-trained model to initialize the feature extractor and classifier of the target model. The global domain adversarial loss L_{fea} is minimized by domain adversarial learning with labeled source data and unlabeled target domain data. Thus, we can achieve a global alignment of the source and target domains. Meanwhile, class-center alignment is implemented by minimizing the semantic alignment loss function L_{sa} using source domain samples and pseudo-label target domain samples. Therefore, we need jointly optimize supervised classification loss L_{cls}, global domain adversarial loss L_{fea}, and semantic alignment loss L_{sa}. Then the overall training loss is as follows:

$$\min_{\varphi,\omega,\varnothing} L_{total}\left(X^S, Y^S, X^T, \overset{\sim}{X^T}\right) = L_{cls}\left(X^S, Y^S; \varphi, \emptyset\right) + \alpha A + \beta B, \tag{10}$$

where $A = L_{fea}(X^S, X^T; \varphi, \omega)$, $B = L_{sa}(X^S, \overset{\sim}{X^T})$.

Algorithm1 Training of our model

Input: Source domain: $D_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$, Target domain: $D_t = \{x_j^t\}_{j=1}^{n_t}$
Pseudo-label: $\tilde{y}_i^t = argmax_c p_c(x_i^t)$

Initialize: parameter of target model $\Omega = \{\varphi, \phi, \omega, \alpha, \beta, \varepsilon\}$

While not done **do**
  Sample mini batch d_s ,d_t data in D_S, D_t do
  for t=1 to batchsize **do**
    Use d_s compute source domain class Center c_s and $c_t \leftarrow c_s$.
    Compute Δc_j by Eq. (8) and update class $c_j^{t+1} = c_j^t - \varepsilon \Delta c_j^t$
    Compute semantic alignment loss L_{sa} by Eq. (7).
    Compute jointly loss function $L_{total}\left(X^S, Y^S, X^T, \widetilde{X^T}; \varphi, \phi, \omega\right)$
    Back propagation L_{total} to get the gradient value of each parameter
    Parameter Ω is updated by gradient descent with *Adam* optimizer
  end for
  Calculate mean loss and mean accuracy
end while
Output: $F_\varphi, C_\phi, D_\omega$

4 Experiment and Result Analysis

4.1 Experimental Settings

1) *Dataset*: We evaluate the proposed approach on two windows malware datasets and a benign dataset selected from the Playdrone dataset [29]. BIG-2015 is a publicly

available dataset from Microsoft on the Kaggle platform, which includes 10,868 training samples and 10,873 test samples from 9 malware families. In this experiment, we only use the training samples. The Mailing dataset includes 9339 malicious samples from 25 malware families. The BIG-2015 malware family samples are converted to grayscale images as network input, and images are normalized to a fixed size of 196 * 196 pixels. The original sample of the Malimg dataset is the grayscale image, so we only transform it to a fixed size of 196 * 196 pixels. Additionally, we use 2280 benign samples in our experiment, where 1140 benign samples are included in the source and target domains, respectively. According to the experimental setup, we use one of the families of the malware dataset as the target domain and the remaining as the source domain, and both source domain and target domain includes benign software. For example, in the BIG-2015 dataset, the Ramnit family is selected as the unlabeled target domain data and the remaining families as the source domain data. The benign data samples are different in the source and target domains. The purpose of this setting is to simulate the ability of the model to detect unknown malware.

2) *Implementation details:* In this paper, we use a residual network as the feature extractor and classifier. To extract the precise feature information of the image, the self-attention module is introduced to associate each pixel with other pixels. For the discriminator, which has the same structure as the classifier, the convolution outputs are *x*-2048-4096-1. Moreover, we use dropout to reduce overfitting by ignoring several neurons with some probability in each training batch. The entire training process uses the Adam optimizer. The pre-training source domain classification loss uses a cross-entropy loss function. During the training of the target model, we simultaneously optimize the cross-entropy loss function, the domain adversarial loss function, and the semantic alignment loss function. The learning rate is set to 1e-4 and the parameters $\alpha = 0.1$ and $\beta = 0.1$, respectively. The learning rate at the local class center update is set to 0.5. The two thresholds r_1, r_2 used in the semantic alignment loss function are set to 0 and 100 respectively. The batch size is set to 32. All experiments are performed in the PyTorch framework. The experimental equipment is as follows: CPU: Intel(R) Core(TM) i5-4590 CPU @ 3.30 GHz 3.30 GH; RAM: 4 GB; OS: Windows 7.
3) *Evaluation metrics:* We evaluate the detection performance of our method by four metrics: accuracy, precision, recall, and F1-score. These evaluation metrics have been extensively applied to various research areas, such as machine learning, natural language processing.

4.2 Performance Comparison of Different Models

In this paper, we have run extensive experiments and calculated the detection accuracy and recall when each class acts as unknown malware (target domain) on the two datasets. We also take into consideration that one dataset (e.g. BIG-2015) acts as the source domain and the other (e.g., Malimg) acts as the target domain for malware detection. In the experiments, the benign software used in the source domain and target domain are

also different. The experimental results are shown in Fig. 3 and Fig. 4. We can see from the figures that the average accuracy and recall on the BIG-2015 dataset are 95.04% and 94.25%, respectively. Moreover, the average accuracy and recall on the Malimg dataset are 95.63% and 95.30%, respectively. In addition, this paper compares our work with traditional machine learning and the state-of-the-art malware detection method, as shown in Table 1.

From Table 1, it can be seen that our method outperforms the other methods. The first two rows are traditional machine learning methods for unknown malware detection. We can see that their accuracy and recall are generally poor. Traditional machine learning is based on feature extraction of known malware to detect unknown malware. Moreover, features are extracted based on feature engineering. Therefore, the unknown malware causes the distribution of training and test samples different. So it decreases the accuracy of model classification. To verify the effectivity of our method, we compare it with the state-of-the-art malware detection methods such as GAA-ADS [27], DART [18], BIRCH [9], and RCNN + transfer learning [28]. From Table 1, it can be seen that our method achieves higher accuracy and recall than the GAA-ADS, which used the clustering method, and RCNN + transfer learning. Moreover, it is also higher than the DART which uses the distribution alignments. DART mitigates the domain discrepancy by jointly optimizing the marginal and manifold distributions of the source and target domains. However, they process the raw data to extract the artificially designed features by traditional machine learning techniques. In this paper, we transform the raw files into grayscale images before feeding them into the neural network. Moreover, our method achieves a higher accuracy and recall by jointly aligning the global and class distributions. It shows that the approach considering class information and semantic alignment has a better performance than the previous approach only considering global domain adaptation. Moreover, it also confirms that class information helps to capture more structural information of the data.

Table 1. Comparison of the detection performance of different methods

Method	Accuracy	Recall	Precision	F1-score
SVM [25]	74.6%	84%	–	75%
SSL [26]	88.54%	90.96%	–	–
GAA-ADS [27]	92.8%	91.3%	–	–
DART [18]	93.9%	91.2%	–	90%
BIRCH [9]	95.02%	90.2%	95.2%	92.3%
RCNN + Transfer Learning [28]	92.8%	–	95.6%	–
OURS	95.63	95.30%	95.34%	93.65%

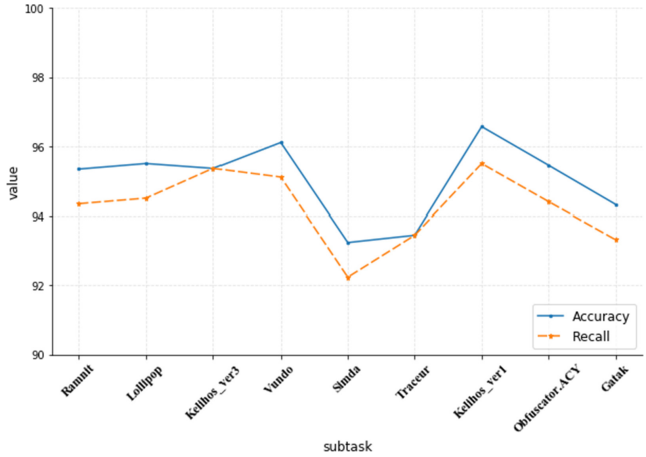

Fig. 3. Detection performance of unknown malware on the BIG-2015 dataset.

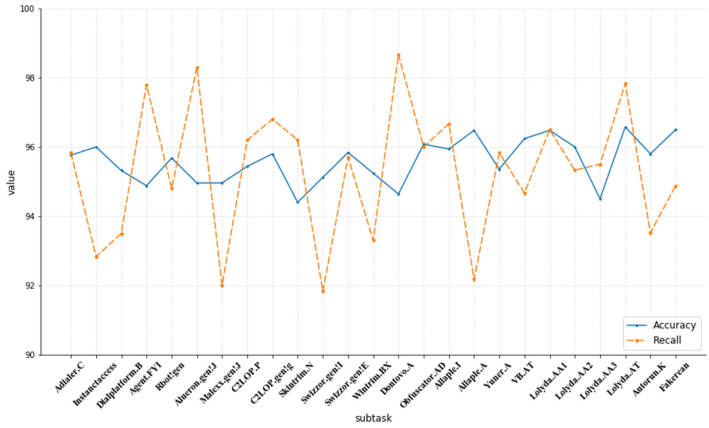

Fig. 4. Detection performance of unknown malware on the Malimg dataset.

5 Conclusion and Future Work

In this paper, we propose an unsupervised domain adaptation approach to detect unknown malware by aligning the joint distribution (global distribution and semantic distribution) of known malware and unknown malware. First, we minimize the discrepancy of the distribution between the source and target domains and learn shared feature representation by adversarial learning. Then, to further obtain the semantic information of unlabeled data, we minimize the distance from the labeled source and pseudo-labeled target domain samples to the class center. Finally, to improve the extraction capability of the model, this paper uses a residual network with a self-attention mechanism as pre-trained model. Extensive experiments are conducted on two datasets, and the results illustrate that the proposed method outperforms the state-of-art detection methods in detecting unknown

malware. In the future, we will work on a more fine-grained domain adaptation approach to the classification of malware families.

Acknowledgement. This work was supported by the National Natural Science Foundation of China under Grants No. 61572170, Natural Science Foundation of Hebei Province of China under Grant No.F2019205163 and No. F2021205004, Science and Technology Foundation Project of Hebei Normal University under Grant No. L2021K06, Science Foundation of Returned Overseas of Hebei Province of China Under Grant No. C2020342, Science Foundation of Department of Human Resources and Social Security of Hebei Province under Grant No. 201901028 and No. ZD2021062, and Natural Science Foundation of Hebei Normal University under Grant No. L072018Z10.

References

1. Li, Z., Wang, D., Morais, E.: Quantum-safe round-optimal password authentication for mobile devices. IEEE Trans. Dependable Secure Comput. (2020). https://doi.org/10.1109/TDSC.2020.3040776
2. Vasan, D., Alazab, M., Wassan, S., et al.: IMCFN: image-based malware classification using fine-tuned convolutional neural network architecture. Comput. Netw. **171**(4), 107–138 (2020)
3. Rong, C., Gou, G., Cui, M., Xiong, G., Li, Z., Guo, L.: TransNet: unseen malware variants detection using deep transfer learning. In: Park, N., Sun, K., Foresti, S., Butler, K., Saxena, N. (eds.) SecureComm 2020. LNICSSITE, vol. 336, pp. 84–101. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-63095-9_5
4. Kumar, S.: MCFT-CNN: malware classification with fine-tune convolution neural networks using traditional and transfer learning in internet of things. Futur. Gener. Comput. Syst. **125**(12), 334–351 (2021)
5. Wilson, G., Cook, D.J.: A survey of unsupervised deep domain adaptation. ACM Trans. Intell. Syst. Technol. **11**(5), 1–46 (2020)
6. Nataraj, L., Yegneswaran, V., Porras, P., et al.: A comparative assessment of malware classification using binary texture analysis and dynamic analysis. In: Proceedings of the ACM Conference on Computer and Communications Security, pp. 21–30. Association for Computing Machinery, New York (2011)
7. Yan, J., Qi, Y., Rao, Q.: Detecting malware with an ensemble method based on deep neural network. Secur. Commun. Netw. **2018**(7247095), 1–16 (2018)
8. Alom, M.Z., Taha, T.M.: Network intrusion detection for cyber security using unsupervised deep learning approaches. In: 2017 IEEE National Aerospace and Electronics Conference (NAECON), pp. 63–69. IEEE, Dayton (2017)
9. Pitolli, G., Laurenza, G., Aniello, L., Querzoni, L., Baldoni, R.: MalFamAware: automatic family identification and malware classification through online clustering. Int. J. Inf. Secur. **20**(3), 371–386 (2020). https://doi.org/10.1007/s10207-020-00509-4
10. Rezende, E., Ruppert, G., Carvalho, T., Theophilo, A., Ramos, F., Geus, P.: Malicious software classification using VGG16 deep neural network's bottleneck features. In: Latifi, S. (ed.) Information Technology New Generations 2018. Advances in Intelligent Systems and Computing, vol. 738, pp. 51–59. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77028-4_9
11. Qiu, S., Wang, D., Xu, G., Kumari, S.: Practical and provably secure three-factor authentication protocol based on extended chaotic-maps for mobile lightweight devices. IEEE Trans. Dependable Secure Comput. (2020). https://doi.org/10.1109/TDSC.2020.3022797

12. Cui, B., Chen, X., Lu, Y.: Semantic segmentation of remote sensing images using transfer learning and deep convolutional neural network with dense connection. IEEE Access **8**(8), 116744–116755 (2020)
13. Pathak, Y., Shukla, P.K., Tiwari, A., et al.: Deep transfer learning based classification model for COVID-19 disease. Innov. Res. BioMed. Eng. (2020). https://doi.org/10.1016/j.irbm.2020.05.003
14. Wang, C., Wang, D., Xu, G., He, D.: Efficient privacy-preserving user authentication scheme with forward secrecy for industry 4.0. Sci. China Inf. Sci. **65**(1), 1–15 (2021). https://doi.org/10.1007/s11432-020-2975-6
15. Sorocky, M.J., Zhou, S., Schoellig, A.P.: Experience selection using dynamics similarity for efficient multi-source transfer learning between robots. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 2739–2745. IEEE, Paris (2020)
16. Vasan, D., Alazab, M., Wassan, S., et al.: Image-based malware classification using ensemble of CNN architectures (IMCEC). Comput. Secur. **92**(5), Article ID: 101748 (2020). https://doi.org/10.1016/j.cose.2020.101748
17. Bartos, K., Sofka, M., Franc, V.: Optimized invariant representation of network traffic for detecting unseen malware variants. In: 25th USENIX Security Symposium, pp. 807–822. USENIX, Austin (2016)
18. Li, H., Chen, Z., Spolaor, R.: DART: detecting unseen malware variants using adaptation regularization transfer learning. In: ICC 2019–2019 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE, Shanghai (2019)
19. Zhu, Y., Zhuang, F., Wang, J., et al.: Multi-representation adaptation network for cross-domain image classification. Neural Netw. **119**(8), 214–221 (2019)
20. Zhuang, F., Cheng, X., Luo, P., et al.: Supervised representation learning: Transfer learning with deep autoencoders. In: Twenty-Fourth International Joint Conference on Artificial Intelligence, pp. 4119–4125. AAAI, Palo Alto (2015)
21. Sun, B., Feng, J., Saenko, K.: Return of frustratingly easy domain adaptation. In: Proceedings of the Thirtieth Conference on Artificial Intelligence, pp. 2058–2065. AAAI, Phoenix (2016)
22. Zhang, H., Goodfellow, I., Metaxas, D., et al.: Self-attention generative adversarial networks. In: International Conference on Machine Learning, pp. 7354–7363. PMLR (2019)
23. Weston, J., Ratle, F., Mobahi, H., Collobert, R.: Deep learning via semi-supervised embedding. In: Montavon, G., Orr, G.B., Müller, K.R. (eds.) Neural Networks: Tricks of the Trade 2012. Lecture Notes in Computer Science, vol. 7700, pp. 639–655. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35289-8_34
24. Wen, Y., Zhang, K., Li, Z., Qiao, Y.: A discriminative feature learning approach for deep face recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9911, pp. 499–515. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_31
25. Sanjaa, B., Chuluun, E.: Malware detection using linear SVM. In: The 8th International Forum on Strategic Technology (IFOST), pp. 136–138. IEEE, Ulaanbaatar (2013)
26. Comar, P. M., Liu, L., Saha, S., et al.: Combining supervised and unsupervised learning for zero-day malware detection. In: 2013 Proceedings IEEE INFOCOM, pp. 2022–2030. IEEE, Turin (2013)
27. Moustafa, N., Slay, J., Creech, G.: Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks. IEEE Trans. Big Data **5**(4), 481–494 (2017)
28. Zhao, Y., Cui, W., Geng, S., et al.: A malware detection method of code texture visualization based on an improved faster RCNN combining transfer learning. IEEE Access **8**(1), 166630–166641 (2020)
29. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: Proceedings of Conference on Computer Vision and Pattern Recognition, pp. 2962–2971. IEEE, Honolulu (2017)