Information Sciences 548 (2021) 450–460

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Transfer learning based countermeasure against label flipping poisoning attack

Patrick P.K. Chan [a,*], Fengzhi Luo [a], Zitong Chen [a,b], Ying Shu [a], Daniel S. Yeung [c]

[a] The School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
[b] The School of Computer Science and Technology, Zhejiang University, Hangzhou, China
[c] Past President, Systems, Man, and Cybernetics Society, IEEE

ARTICLE INFO

Article history:
Received 10 July 2020
Received in revised form 7 October 2020
Accepted 10 October 2020
Available online 19 October 2020

Keywords:
Adversarial Learning
Poisoning Attack
Label Flipping Attack
Transfer Learning
Robustness

ABSTRACT

Recent studies indicate that a classifier is vulnerable in an adversarial environment. The label flipping attack aims to mislead the training process. Some countermeasures have been proposed, but are usually designed for a particular classifier only, or may cause information loss. This study aims to investigate a generic model which fully utilizes the contaminated samples in learning. We assume a small untainted dataset is obtained from an application in addition to a contaminated dataset. The adversarial learning problem is formulated as transfer learning in which the influence of contaminated samples is reduced by only extracting the information similar to the untainted samples from the contaminated set using transfer learning. Our study considers a popular method, TrAdaBoost, and indicates that its performance is closely related to the initialized weights of samples. A initialization method is devised specifically for an adversarial setting to avoid assigning a large weight to the contaminated samples. The experimental results confirm that TrAdaBoost extracts only the benign knowledge from the contaminated set successfully. Moreover, our proposed initialization method significantly enhances the robustness of the model. This study presents a promising direction using transfer learning to defend against poisoning attacks.

1. Introduction

Although machine learning techniques are now increasingly prevalent in security-related applications [1], *e.g.*, biometric identification [2], their vulnerability has yet to be thoroughly investigated. Recently, many research studies [3–6] reveal security leakages of machine learning systems in an adversarial environment, in which the samples are carefully crafted to mislead the decision of a machine learning model. Since an adversary attack violates the assumption of the same (or similar) distributions of the training and test sets made by most machine learning methods [7], the efficacy of the model drops significantly in an adversarial environment. For example, a tiny modification on a sample may mislead the decision of a Deep Neural Network [8], and one malicious training sample can significantly reduce the accuracy of a Support Vector Machine [9]. These results raise the public concern on the security issues of machine learning.

The poisoning attack is a popular type of adversarial attack in which the training samples are manipulated by injecting feature noise [10] and modifying the labels [11] to mislead a learning process. Since samples may be collected from unreliable sources, *e.g.*, Amazon's Mechanical Turk [12] and recommender systems [13], the quality of the training set can be

* Corresponding author.

https://doi.org/10.1016/j.ins.2020.10.016
0020-0255/

easily reduced by an adversary. Countermeasures against poisoning attacks are categorized into robust learning [14–16] and data sanitization [17,18]. Not only the generalization ability but also the robustness of a model are maximized in robust learning [14]. However, a robust learning method is designed for a specific learning algorithm and may not be generalized to others. On the other hand, in data sanitization methods the suspicious samples are identified according to pre-defined criteria, *e.g.*, classification accuracy [18], and data complexity [17]. The time complexity is a concern since classifier training is required in the sample evaluation. In addition, removing the suspicious samples may cause information loss in learning [2,19]. As a result, a generic method which takes full advantages of given samples is needed.

Our study aims to propose a robust learning model which is generic to any learning algorithm and utilizes the knowledge of the training samples. We investigate countermeasures to defend against poisoning attacks in the framework of transfer learning which is suitable for any classier. Transfer Learning [20] is a mechanism which aims to improve the performance on a target task by using knowledge acquired from a related task called a source task, and has been commonly used in applications without sufficient labeled training samples [21,22]. A sample in a contaminated training set may be either untainted or contaminated, which is unknown to users. We assume there is a small untainted training set collected from the same problem, which may be obtained by sanitizing a small portion of the contaminated training set manually. The untainted training set is used as target data in transfer learning since all samples in this set are clean. However, since the size of the target dataset is too small to build a satisfactory model, the contaminated set defined as the source dataset is also required. As a result, the adversarial learning problem is formulated as transfer learning, in which the similar knowledge to the untainted set is extracted from of the contaminated set to improve the learning on the target task, *i.e.* the influence of the contaminated samples to training is reduced.

TrAdaBoost [21] is considered due to its popularity and exemplary performance in many applications [23,24]. Our preliminary study indicates that TrAdaBoost highly relies on the initial weight values representing the similarity between the source samples and the target problem, and assigning an improper value to the contaminated samples causes the training to be affected significantly. Therefore, the improper weight value of a sample is initialized according to its classification accuracy of an intuitive classifier in our revised model to obtain a better initialization. Our proposed model is more flexible than most existing robust learning methods since our model can adopt any classifier. Moreover, in contrast with data sanitization, contaminated samples are evaluated iteratively to avoid information loss caused by wrongly identifying a clean sample as contaminated. In out method, samples with low reliability still contribute to training but with less importance. Our proposed model is then evaluated and compared experimentally with TrAdaBoost, AdaBoost [25], Reject On Negative Impact (RONI) [18], and SR-LSSVM [26], which are representatives of the traditional classifier, data sanitization, and robust learning methods respectively. The label flipping attack is considered in our study as it is the most common poisoning attack. The Support Vector Machine (SVM), which is a binary classification algorithm commonly used in many applications, is used as a base classifier for TrAdaBoost. Experimental results confirm that our proposed model can significantly reduce the attack success rate.

The rest of the paper is organized as follows. Section 2 provides a brief introduction on the related concepts of this study, including label flipping attacks and its countermeasures. The transfer learning-based poisoning attack defense method and its weight initialization improvement are devised in Section 3. The experimental results and discussion are given in Section 4. Finally, Section 5 concludes this study, and Section 6 discusses future study.

2. Background

The related concepts of this study are introduced in this section. The current poisoning attack methods, their countermeasures, and transfer learning are briefly introduced.

2.1. Label flipping attack

The label flipping attack, which is one common poisoning attack method, manipulates the label information of training samples. Labels of a certain number of training samples are flipped in the attack. Recently, most of the attack methods determine the adversarial samples according to the decision hyperplane of a targeted classifier [27]. Nearest and Furthest Label Flipping attacks [27] manipulate the labels of the samples closest to and furthest away from the decision plane in order to change the model. However, some studies [28,11] indicate that the decision plane may not be affected by Nearest and Furthest Label Flipping Attacks effectively. To maximize the rotation of the decision plane, the Far-Rotate Attack [11] considers not only the targeted classifier but also a a randomly generated classifier. The samples are attacked if they are furthest away from both of these decision planes. Another attack, the Adversarial Label Flipping Attack (ALFA) [11], evaluates the impact of label flipping on the accuracy of the classifier. The samples with the most impact are attacked. When no knowledge of a targeted system is obtained, *i.e.*, a black-box attack scenario, a surrogate classifier is usually used to replace the real targeted system. The poisoning attacks are illustrated in an artificial dataset which is a linearly separable classification problem with two classes shown in Fig. 1. The dots in red and blue represent the samples from two different classes. Labels of 20% of the training samples are flipped and are denoted by a dot with a circle. A SVM with the linear kernel is used as a classifier, denoted by the black lines. Since the samples are attacked randomly in Random Attack, the decision plane of SVM does not change dramatically. Nearest and Furthest Label Flipping Attack also causes a small change on the decision plane since

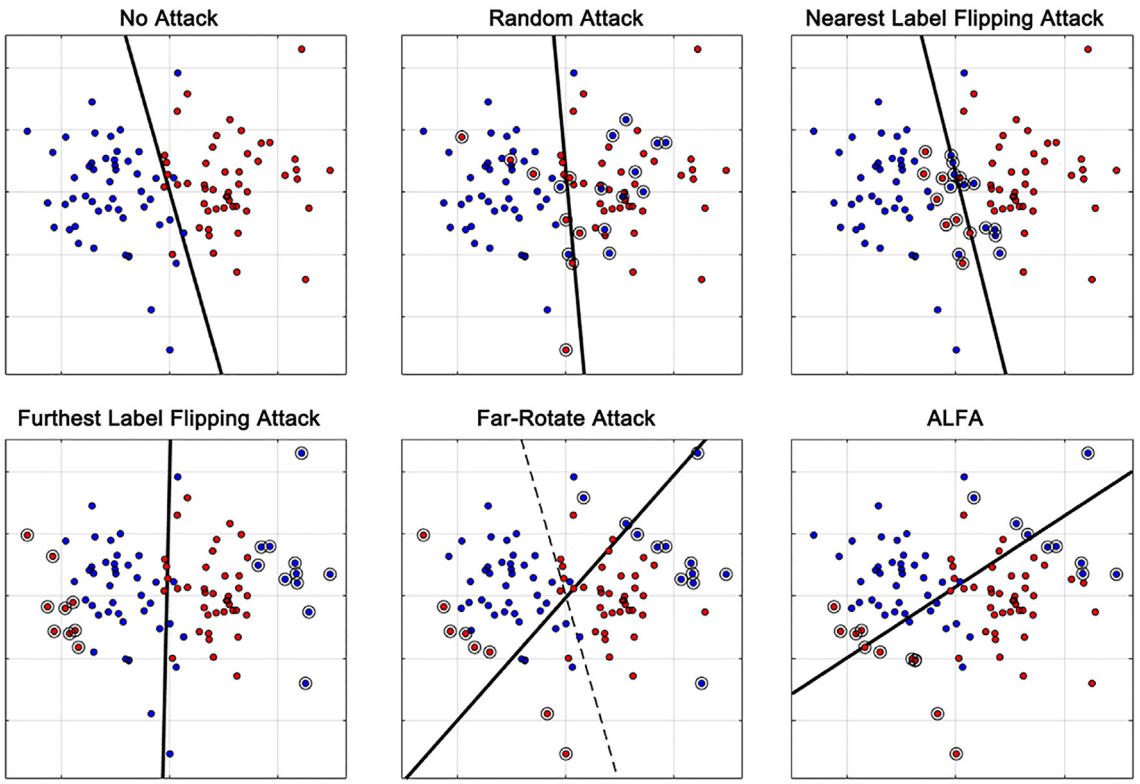

Fig. 1. Label flipping attacks, including Nearest Label Flipping Attack, Furthest Label Flipping Attack, ALFA, Far-Rotate Attack, and Random Attack, on an artificial dataset. Red and blue dots indicate the samples of two classes, and a dot in a circle denotes an attack sample. The black line denotes the SVM with the linear kernel trained on the contaminated samples.

the flipped samples are mainly symmetrical on the decision plane. Usually, the attack performance of ALFA and Far-Rotate Attack is better in comparison to other attack methods, *i.e.* their decision planes are affected more significantly.

2.2. Countermeasures of label flipping attack

The suspicious samples are filtered out from a training set before learning in data sanitization [29] to reduce the influence of poisoning attacks. There are several evaluation criteria, for example, accuracy [18] and data complexity [17]. Reject On Negative Impact (RONI) removes a contaminated sample with a significant negative impact on classification accuracy. By assuming an untainted dataset is obtained, a sample has a negative impact if the test accuracy of a classifier trained using the training set with the sample is lower than that of the classifier trained without the sample. On the other hand, the training samples are evaluated by the data complexity of a training set after removing a sample and its nearest samples [17]. Contaminated samples are then distinguished from untainted ones according to their data complexity values. However, high time complexity is one of the drawbacks since training the model is required for sample evaluation. Moreover, the detection accuracy may drop if the amount of untainted samples is not sufficient. Removing suspect samples may also cause information loss in training.

Generalization ability is an important objective in machine learning, and many studies aim to enhance [30,31] and investigate [32,33] the generalization ability of a system. However, a classifier only considering the generalization ability may be misled by a carefully crafted sample [5]. Therefore, another countermeasure is robust learning in which not only the generalization ability but also the impact of attacked samples is considered. Studies of robust learning mainly focus on Support Vector Machines (SVMs), *e.g.* Robust LS-SVM (RLS-SVM) [34] and sparse RLS-SVM (SR-LSSVM) [35] which introduces the truncated least squares loss for classification to reduce the sensitivity to outliers. Since robust learning is usually designed for a particular classifier, it may not be applied to another method. In addition, its generalization ability is generally sacrificed to increase the robustness of the classifier [36].

2.3. Transfer learning

Sufficient labeled samples play an essential role in supervised learning. However, data collection and labeling is expensive and time consuming in some applications, *e.g.* medical diagnosis [37]. As opposed to traditional machine learning approaches which require the training and test samples to follow the same (or similar) distributions, transfer learning [20] allows the domains, tasks and distributions of the training and test sets to be different. Transfer learning bridges the knowledge of applications to assist learning on a task with insufficient training data (called a target task) by using the related auxiliary data from other tasks (called a source task). Many studies [38,39,23,24] confirm that the performance on a target task is improved significantly using the knowledge of the source tasks by transfer learning in applications.

3. Transfer learning based robust learning

In transfer learning, relevant information is extracted from source tasks to assist learning on the target task, which contains inadequate samples for training a classifier with satisfactory performance. This mechanism inspires us to formulate a countermeasure for poisoning attacks in the framework of transfer learning. In a poisoning attack, a large dataset which may be contaminated is given. By assuming there is a small untainted set, which can be prepared by sanitizing manually, only useful information provided by the contaminated dataset (the source task) is extracted and transferred to the target task represented by the small untainted set via transfer learning, *i.e.* the influence of contaminated samples on training is reduced. TrAdaBoost, which is one of the most popular transfer learning methods, is used in our model. It is a general framework and any classifiers can be used as a base classifier. Moreover, the importance of each sample in the contaminated set is adjusted iteratively during training, and samples suspected by the evaluation process will not be removed immediately. The proposed framework overcomes problems of information loss in data sanitization and over generalization in robust learning respectively. Section 3.1 and 3.2 introduce the proposed algorithm in detail and discuss its time complexity.

3.1. Algorithm

A large contaminated training set Tr_A as well as a small untainted set Tr_c are obtained, where $|\mathrm{Tr}_A| = m \gg |\mathrm{Tr}_c| = n$. The aim of our model is to minimize the classification error of a classifier trained by using Tr_A and Tr_c on an untainted test set Te. Tr_c and $\mathrm{Te} \sim \mathscr{D}$, while $\mathrm{Tr}_A \sim \mathscr{D}_A$ where $\mathscr{D}$ and $\mathscr{D}_A$ are unknown distributions. As Tr_A and Te follow different distributions, a classifier trained with Tr_A may not achieve satisfactory performance on Te. As a result, our model applies transfer learning, *i.e.*, TrAdaBoost, to transfer the knowledge from Tr_A to Tr_c. We let $\mathrm{Tr} = \mathrm{Tr}_A \cup \mathrm{Tr}_c$ be the training set. $\mathrm{Tr} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n), (\mathbf{x}_{n+1}, y_{n+1}), \ldots, (\mathbf{x}_{m+n}, y_{m+n})\}$, where $(\mathbf{x}_i, y_i) \in \mathrm{Tr}_c$ when $1 \leqslant i \leqslant n$, and $(\mathbf{x}_i, y_i) \in \mathrm{Tr}_A$ when $n+1 \leqslant i \leqslant m+n$. $\mathbf{x}_i$ and y_i denote the feature vector and the desired output of the ith sample respectively. A weight value w_i^j is assigned to $\mathbf{x}_i$ to indicate its importance on training, where j denotes the number of iterations, and $j = 0$ indicate the initialized weight. $0 \leqslant w_i^j \leqslant 1$. When $w_i^j = 0$, $\mathbf{x}_i$ does not affect the jth iteration in the training progress. On the other hand, $w_i^j = 1$ means $\mathbf{x}_i$ plays an important role in training.

The flow chart of our proposed TrAdaBoost method is shown in Fig. 2. In our model, a number of base classifiers are trained using Tr. A novel weight initialization method is proposed, and the weight value is initialized according to a trivial classifier. In each iteration, a base classifier is trained on a training set generated by randomly selecting the samples based on the distribution of their weight without replacement. The weight value of a training sample is updated according to its type and whether or not the base classifier can be classified correctly. The looping stops until k base classifiers are trained. The final ensemble system consists of the last $k/2$ base classifiers. Using more base classifiers enhances the robustness of the ensemble system in an adversarial environment [15], but also increases the cost as well. Therefore, k should be determined according to the resources and limitations of an application.

This section is arranged as follows. Our proposed weight initialization method is introduced in Section 3.1.1. The iterative training and the combination of k base classifiers are shown in Section 3.1.2.

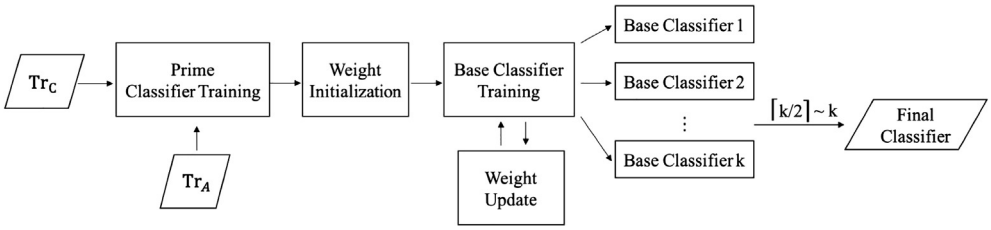

Fig. 2. Logic flow of the training of our model.

3.1.1. Weight initialization

The weight values play an important role in TrAdaBoost as they indicate the importance of a sample on training. In adversarial learning, a smaller weight should be assigned to the contaminated samples in comparison with the untainted samples. If so, the classifier will mainly learn from the untainted samples, *i.e.*, the effect of contaminated samples is reduced. However, the weight initialization of the original TrAdaboost only assigns the same weight value to all source data. The contaminated samples in the source data may influence the decision plane trained in the first iteration of the training process, which may mislead the rest of the learning process and dominate the whole training process easily. As a result, the initialization method of the original TrAdaboost may not work effectively in an adversarial setting, and a new method should be specifically designed.

Let $\mathbf{W}^i = \{w_1^i, w_2^i, \ldots, w_{n+m}^i\}$ be the set of weight values in the ith iteration, where w_j^i is the weight value for the jth sample. In the process of weight initialization, $\mathbf{W}^0$ is calculated by training a trivial classifier h_{te} which roughly determines whether a sample in Tr_A is contaminated. A smaller weight value is assigned to a suspect sample in the initialization, and vice versa. h_{te} is trained by using Tr, and a weight value w_i^{te} is assigned to $\mathbf{x}_i$ as follows:

$$w_i^{te} = \begin{cases} 1/n & i = 1, \ldots, n \\ 1/m & i = n+1, \ldots, n+m \end{cases} \tag{1}$$

Note that the first n samples are in Tr_c and the rest are in Tr_A. Since $n \ll m$, the weight values assigned to samples in Tr_c are larger than the ones in Tr_A. There is less confidence in the correctness of information provided by the samples in Tr_A. Therefore, a smaller weight is assigned to Tr_A to reduce its influence while training h_{te}. The proposed weight initialization reduces the importance of the samples in Tr_A classified incorrectly by h_{te} in terms of square error, shown in (2). A larger weight is assigned to a sample with less error in Tr_A, while a constant weight value is assigned to samples in Tr_c.

$$w_i^0 = \begin{cases} 1/n & i = 1, \ldots, n \\ ||y_i - h_{te}(\mathbf{x}_i)||_2 & i = n+1, \ldots, n+m \end{cases} \tag{2}$$

3.1.2. Training and combination

A base classifier h^t is trained iteratively according to $\mathrm{Tr}_A, \mathrm{Tr}_c$ and $\mathbf{W}^{t-1}$ in the tth iteration, where $t = 1, 2, \ldots, k$, and k is the number of base classifiers. The algorithm aims to classify all samples in Tr_c correctly since they provide correct information. When $\mathbf{x}$ is misclassified, its weight will increase to emphasize its significance in future iterations. The formula of the weight update on $\mathbf{x}_i$ in Tr_c is defined as follows:

$$w_i^{t+1} = w_i^t \beta^{\gamma |h_t(\mathbf{x}_i) - c(\mathbf{x}_i)|} \tag{3}$$

where $\beta = 1/(1+\sqrt{1\ln(n/N)}$ and $1 \leqslant i \leqslant n$. On the other hand, as the samples in Tr_A may potentially be attacked, their weights are updated differently from the ones in Tr_c. A wrong classification result on $\mathbf{x} \in \mathrm{Tr}_A$ may indicate that $\mathbf{x}$ is different from the other samples, *i.e.* $\mathbf{x}$ is likely to be an attack sample. Therefore, the weight value of $\mathbf{x}$ is reduced to lower its impact on learning. The weight update on $\mathbf{x}_i$ in Tr_A is shown in (4).

$$w_i^{t+1} = w_i^t \beta_t^{-|h_t(\mathbf{x}_i) - c(\mathbf{x}_i)|} \tag{4}$$

where $\beta_t = \epsilon_t/(1-\epsilon_t)$, ϵ_t is the error of h_t on Tr_A, and $i = n+1, n+2, \ldots, n+m$. The ensemble system only consists of the last $k/2$ trained base classifiers, shown in (5), where $\alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t})$.

$$h_f(x) = sign(\sum_{t=\lceil \frac{k}{2} \rceil}^{k} \alpha_t h_t(x)) \tag{5}$$

3.2. Time complexity

Comparing with the original TrAdaboost, our method requires additional training for a trivial classifier for weight initialization. However, the time complexity of our method is the same as the original TrAdaboost in terms of the big-O, which is in $O(k(n+m))$, where n and m denote the sample numbers in untainted and contaminated sets, and k is the number of base classifiers.

4. Experiments

The performance of our revised TrAdaBoost method is evaluated and compared with other existing methods under label flipping poisoning attacks experimentally in this section. We first introduce the datasets and the experimental settings in Section 4.1. Afterward, the experiment results are exhibited and analyzed in Section 4.2.

4.1. Dataset and experimental setting

Three security-related applications, including Letter Recognition, Spam Assassin, and Phishing Website Detection, are considered in this experiment. The Letter Recognition dataset [40] contains total of 20,000 samples with 16 features which represent 26 capital letters in the English alphabet. Only classes U and V containing 1536 samples in total are used in our experiments. The Spam Assassin dataset [41] is a 2-class classification problem containing the 5000 samples with 500 features indicating the presence of a word in an email (0 denotes absence and 1 denotes presence). A phishing website is a kind of malicious website set up for cheating users out of important private information like bank card numbers and passwords. The Phishing Website dataset [42] contains information about 5528 normal and abnormal website records denoted by 30 features. All datasets mentioned are in numerical form. The details of the datasets are shown in Table 1.

We assume the attacker obtains full knowledge of the classification strategies including the training set, the type of classifier, and its parameters. 6% of the training data is chosen as the small untainted set (Tr_c) and the rest of the set is used as the contaminated set (Tr_A), where $\mathrm{Tr}_A \cap \mathrm{Tr}_c = \varnothing$. A portion of Tr_A is attacked by four poisoning attack algorithms, including Nearest Label Flipping Attack (Nearest Attack), Furthest Label Flipping Attack (Furthest Attack), Adversarial Label Flipping Attack (ALFA) and Adversarial Label Flipping Attack (Far-Rotate Attack), separately with different contamination ratios.

Besides the traditional *TrAdaboost*, our proposed model (denoted by *our method*) is also compared with *RONI* and *SR-LSSVM*, which are the representatives of data sanitization and robust learning methods respectively. *RONI* divides the training data into three validation sets. All samples in Tr_c are randomly separated into these validation sets to verify a trained model in *RONI*. Samples in Tr_A which downgrade the trained model are regarded as vulnerable samples and are subsequently removed. Finally, AdaBoost with SVM as a base classifier is trained by using the sanitized samples obtained by *RONI*. *SR-LSSVM* is used as a base classifier in Adaboost by using both Tr_A and Tr_c in training. The traditional AdaBoost algorithm is trained by using Tr_A and Tr_c denoted as *AdaBoost*, and only using Tr_c denoted as $Adaboost_{Target}$ is also considered as the baseline. The number of base classifiers of all ensemble methods, *i.e. AdaBoost*, *TrAdaBoost* and *Our Method*, is set as 50. The Support Vector Machine (SVM) with the Gaussian kernel is used as a base classifier. The SVM regularization parameter $C \in \{2^{-5}, 2^{-4}, \ldots, 2^{3}\}$ is selected according to an additional inner 5-fold cross-validation to maximize classification accuracy, usually yielding $C = 1$. All algorithms are implemented using MATLAB 2016a in Windows 10 and are executed in a PC with Intel Core i5-6300HQ CPU. The experimental results are the average of ten independent runs.

4.2. Experimental results

The experimental results are discussed in this section. Section 4.2.1 first evaluates and compares the performance of our method, and other existing and baseline methods under different label flipping attack settings. The influence of our proposed weight initialization method on the performance in an adversarial environment is investigated in Section 4.2.2. Finally, the running times of all methods are discussed in Section 4.2.3.

4.2.1. Robustness improved TrAdaBoost-based poisoning attack defense method

The performance of *Adaboost*, $Adaboost_{Target}$, *RONI*, *SR-LSSVM*, *TrAdaboost*, and *our method* under different types of label flipping attacks is evaluated in terms of accuracy in the three datasets. The attack ratio of Tr_A varies from 5% to 25%.

Fig. 3 shows the experimental results. Accuracy of all methods drop when the attack ratio increases, except $Adaboost_{Target}$, because it only uses the clean dataset in training. On the other hand, *Adaboost* ignores the influence of the contaminated dataset, and uses both Tr_A and Tr_c to train the classifier. *Adaboost* and $Adaboost_{Target}$ are the two trivial methods to deal with the contaminated dataset and serve as a baseline for comparison.

The results show that the test accuracy of *our method* maintains at a relatively high level compared with other methods. *TrAdaboost* also achieves excellent results in most cases, especially in nearest attack and ALFA. Both *our method* and *TrAdaboost* perform better than $Adaboost_{Target}$ and *Adaboost* generally, which confirm that learning on the target task can be improved by extracting the useful knowledge from the contaminated set via transfer learning. Different from *TrAdaboost*,

Table 1
The datasets used in the experiments.

Dataset	Training Set	Test Set	Feature	Class
Letter Recognition (Only U and V)	169	1367	16	2
Spam Assassin	100	4900	500	2
Phishing Website	166	5362	30	2

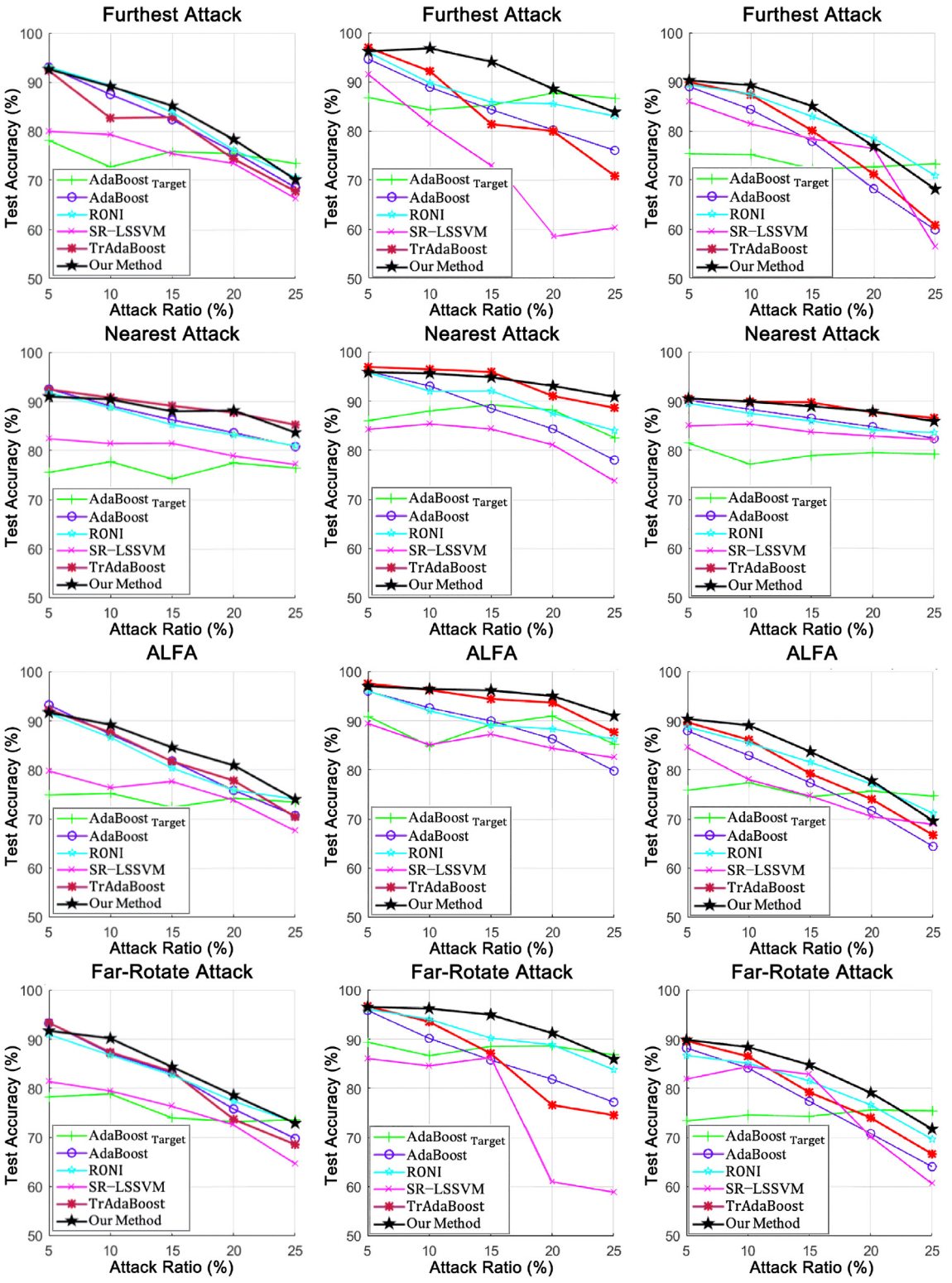

Fig. 3. Accuracy of *Adaboost*, *Adaboost*$_{Target}$, *RONI*, *SR-LSSVM*, *TrAdaboost* and *our method* under label flipping attacks with different attack ratios.

our method has stable and consistent performance in all datasets under different kinds of label flipping attacks. The performance of *TrAdaboost* drops significantly in the Spam dataset under Furthest and Far-Rotate attack, which is even worse than *Adaboost* and $Adaboost_{Target}$. The results indicate that our proposed weight initialization improves the performance of transfer learning significantly in an adversarial environment.

RONI has satisfactory performance although its accuracy is lower than our proposed method in most cases. However, *RONI* is not suitable for Nearest Attack, *i.e.*, its performance is worse than *our method* and also *TrAdaboost* obviously under Nearest Attack in all datasets. As only a few samples are evaluated each time, the attack samples of Nearest Attack located around a decision plane may not significantly change the classifiers in the evaluation of *RONI*. This may explain why *RONI* cannot identify contaminated samples generated by Nearest attack successfully. The performance of *SR-LSSVM* is not satisfactory in the experiments. Its accuracy is usually lower than the trivial methods, *i.e.*, *Adaboost* and $Adaboost_{Target}$.

4.2.2. Initialization comparison

The importance of the weight initialization to *TrAdaBoost* is investigated deeply in this section. Five initialization methods, including the standard one used in *TrAdaBoost* (*Standard*), our proposed initialization method mentioned in Section 3.1.1 (*Our Method*), random initialization (*Random*), and two ideal methods (*Ideal+* and *Ideal−*), are considered. *Random* arbitrarily chooses numbers ranging from $[0, 1]$ for each sample. *Ideal+* and *Ideal−* assume that the contaminated samples in Tr_A are known. In *Ideal+*, 0 (1) is assigned to an untainted (contaminated) sample, while *Ideal-* assigns the values in the opposite way.

The experimental results shown in Fig. 4 indicate that *Ideal+* and *Ideal−* achieve the best and the worst performance in all cases as expected. It confirms that the identification of contaminated samples plays a critical role in weight initialization. Generally, the performance of the weight initialization used in *TrAdaBoost* (*i.e.*, *Standard*) is similar to *Random*, which means the weight initialization of *TrAdaBoost* cannot work effectively under a poisoning attack. In contrast, *Our Method* achieves dramatically higher accuracy than *Standard* and *Random*, and our performance is substantially closer to *Ideal+*. The results confirm that our weight initialization method identifies the potential contaminated samples reasonably and improves the performance of transfer learning in an adversarial environment effectively.

4.2.3. Running time comparison

The training times of *RONI*, *SR-LSSVM*, *TrAdaboost*, and *our method* are discussed in this section. Table 2 shows the running times of these methods for different datasets in terms of seconds. 15% of Tr_A is attacked. The training time is independent to an attack method, and ALFA is chosen as an example for demonstration.

The result shows that the running time of *our method* is insignificantly higher than *TrAdaboost* due to the additional training on the trivial classifier used during weight initialization. *RONI* has the longest running time due to the classifier training in the evaluation process. The running time of *SR-LSSVM* is much shorter than the others because no iterative process is required and a low-rank approximation of the kernel matrix is applied to reduce the time complexity [26].

5. Conclusions

This study investigates the formulation of learning on a contaminated dataset as a transfer learning problem. TrAdaBoost is used due to its popularity and excellent performance in many applications. Our model aims to reduce the influence of the contaminated samples on training by only extracting the knowledge similar to the untainted set from the contaminated set. Different from existing robust learning methods which are designed for a specific learning algorithm, our proposed framework can be applied to any learning algorithm. In addition, the information loss caused by data sanitization can be avoided since samples in the contaminated set are evaluated iteratively during training. Moreover, our study found that improper weight initialization on the contaminated set leads to poor performance of TrAdaBoost in an adversarial environment. A weight initialization method which assigns a weight to a sample according to the correctness of its predicted value given by a trivial classifier is proposed against poisoning attacks. The experimental results indicate that TrAdaBoost achieves excellent performance in terms of stability and accuracy compared with *RONI* and *SR-LSSV*, which are the representatives of data sanitization and robust learning, under label-flip attacks. The results also confirm that our proposed weight initialization method improves the robustness of TrAdaBoost significantly in the adversarial settings. It confirms that the framework of transfer learning reduces the influence of contaminated samples on learning and extracts only useful knowledge from the contaminated set.

6. Future work

One possible future work of this study is to investigate the performance of our method under poisoning attacks with feature noise. In this kind of poisoning attack, the difference between poisoned and untainted samples is smaller than one in the label flipping attack. It may increase the difficulty of transferring knowledge from the contaminated to the untainted dataset. Careful investigation of weight adjustment is required.

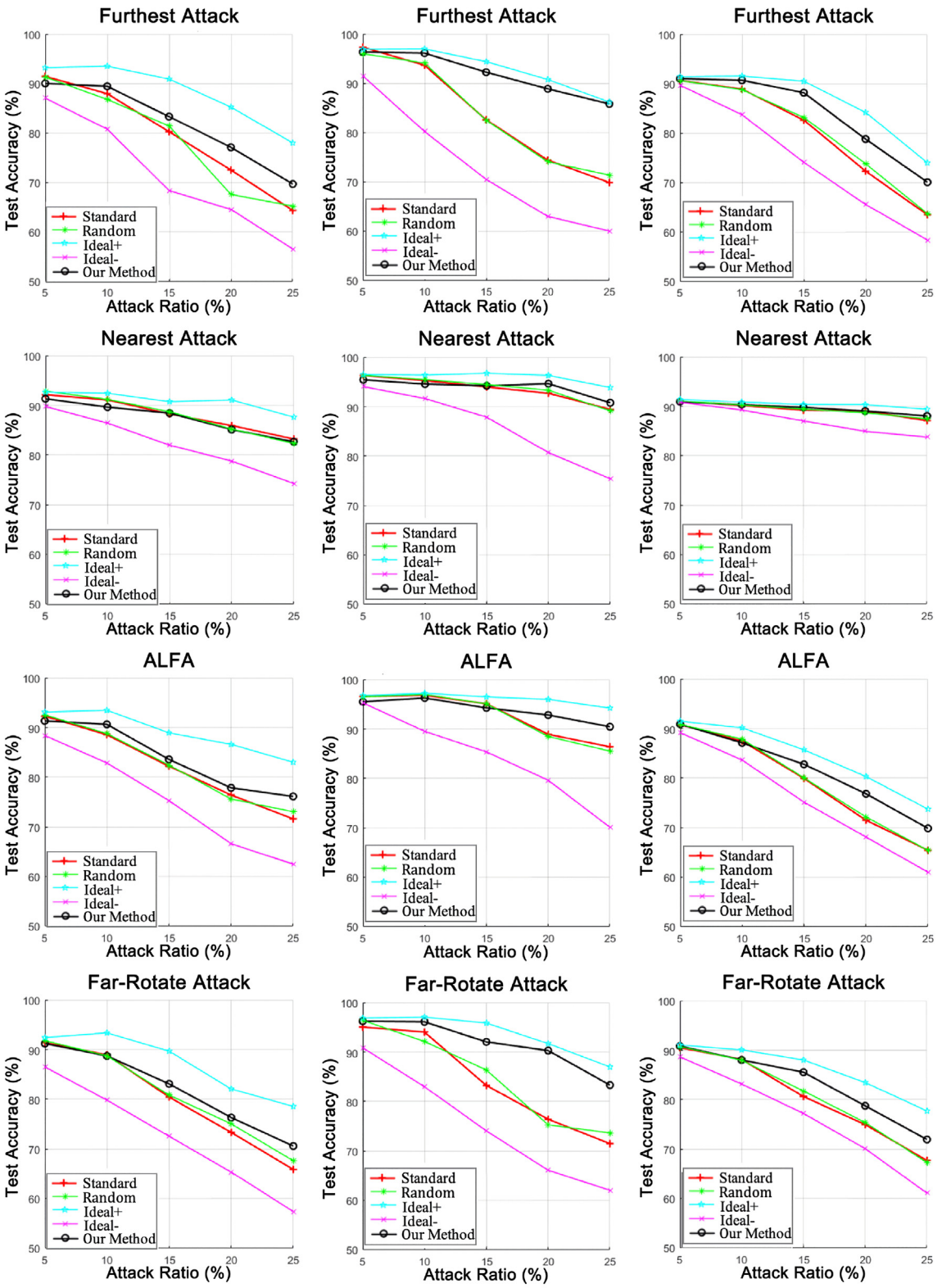

Fig. 4. *TrAdaboost* with five different initialization methods (*i.e.*, *Standard*, *Our Method*, *Random*, *Ideal+* and *Ideal−*) under label flipping attacks with different attack ratios.

Table 2
Training time (in seconds) and the ranking (the fastest (slowest) method is ranked as 1 (5)) of *RONI*, *SR-LSSVM*, *TrAdaboost* and *our method* in different datasets.

Method	RONI	SR-LSSVM	TrAdaBoost	Our Method
Letter Recognition	7.03 (4)	0.11 (1)	5.14 (2)	5.85 (3)
Spam Assassin	87.05 (4)	0.14 (1)	68.03 (2)	69.23 (3)
Phishing Website	37.26 (4)	0.13 (1)	32.83 (2)	33.40 (3)

CRediT authorship contribution statement

Patrick P.K. Chan: Conceptualization, Writing - original draft, Writing - review & editing, Funding acquisition. **Fengzhi Luo:** Writing - original draft, Software, Validation. **Zitong Chen:** Methodology, Writing - original draft. **Ying Shu:** Investigation, Methodology, Formal analysis. **Daniel S. Yeung:** Supervision, Conceptualization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported by the Natural Science Foundation of Guangdong Province, China (No. 2018A030313203), and the Fundamental Research Funds for the Central Universities (No. 2018ZD32).

References

[1] J. Martínez Torres, C. Iglesias Comesaña, P.J. García-Nieto, Review: machine learning techniques applied to cybersecurity, Int. J. Mach. Learn. Cybern. 10 (10) (2019) 2823–2836.
[2] B. Biggio, Z. Akhtar, G. Fumera, G.L. Marcialis, F. Roli, Security evaluation of biometric authentication systems under real spoofing attacks, IET Biometr. 1 (1) (2012) 11–24.
[3] P.P. Chan, Y. Wang, D.S. Yeung, Adversarial attack against deep reinforcement learning with static reward impact map, in: Proceedings of the 2020 ACM Asia Conference on Computer and Communications Security, 2020.
[4] P.P.K. Chan, W. Liu, D. Chen, D.S. Yeung, F. Zhang, X. Wang, C. Hsu, Face liveness detection using a flash against 2d spoofing attack, IEEE Trans. Inf. Forensics Secur. 13 (2) (2018) 521–534.
[5] B. Biggio, G. Fumera, F. Roli, Security evaluation of pattern classifiers under attack, arXiv: Learning..
[6] K. Chen, P.P.K. Chan, F. Zhang, Q. Li, Shilling attack based on item popularity and rated item correlation against collaborative filtering, Int. J. Mach. Learn. Cybern. 10 (7) (2019) 1833–1845, https://doi.org/10.1007/s13042-018-0861-2.
[7] M. Barreno, B. Nelson, A.D. Joseph, J.D. Tygar, The security of machine learning, Mach. Learn. 81 (2) (2010) 121–148.
[8] S.-M. Moosavi-Dezfooli, A. Fawzi, P. Frossard, Deepfool: A simple and accurate method to fool deep neural networks, IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016).
[9] B. Biggio, B. Nelson, P. Laskov, Poisoning attacks against support vector machines, in: Proceedings of International Conference on Machine Learning ICML'12, 2012, pp. 1467–1474.
[10] B. Nelson, M. Barreno, F. Jack, C. Anthony, D. Joseph, B.I.P. Rubinstein, U. Saini, C. Sutton, J.D. Tygar, K. Xia, Exploiting machine learning to subvert your spam filter, in: Proceedings of the First USENIX Workshop on LargeScale Exploits and Emergent Threats, 2008.
[11] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, F. Roli, Support vector machines under adversarial label contamination, Neurocomputing 160 (2015) 53–62.
[12] M.D. Buhrmester, T. Kwang, S.D. Gosling, Amazon's mechanical turk: a new source of inexpensive, yet high-quality, data?, Perspect. Psychol. Sci. 6 (1) (2011) 3–5.
[13] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, IEEE Trans. Knowl. Data Eng. 17 (6) (2005) 734–749.
[14] R. Huang, B. Xu, D. Schuurmans, C. Szepesv??ri, Learning with a strong adversary, CoRR..
[15] B. Biggio, G. Fumera, F. Roli, Multiple classifier systems for robust classifier design in adversarial environments, Int. J. Mach. Learn. Cybern. 1 (1) (2010) 27–41.
[16] A. Demontis, B. Biggio, G. Fumera, G. Giacinto, F. Roli, Infinity-norm support vector machines against adversarial label contamination, ITASEC (2017) 106–115.
[17] P.P.K. Chan, Z. He, H. Li, C. Hsu, Data sanitization against adversarial label contamination based on data complexity, Int. J. Mach. Learn. Cybern. 9 (6) (2018) 1039–1052.
[18] M. Barreno, B. Nelson, R. Sears, A.D. Joseph, J.D. Tygar, Can machine learning be secure (2006) 16–25.
[19] P.P.K. Chan, Z. He, X. Hu, E.C.C. Tsang, D.S. Yeung, W.W.Y. Ng, Causative label flip attack detection with data complexity measures, Int. J. Mach. Learn. Cybern. https://link.springer.com/article/10.1007/s13042-020-01159-7
[20] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[21] W. Dai, Q. Yang, G. Xue, Y. Yu, Boosting for transfer learning 227 (2007) 193–200..
[22] M. Talo, U.B. Baloglu, zal Yildirim, U. Rajendra Acharya, Application of deep transfer learning for automated brain abnormality classification using mr images, Cogn. Syst. Res. 54 (2019) 176–188.
[23] S. Jiang, H. Mao, Z. Ding, Y. Fu, Deep decision tree transfer boosting, IEEE Trans. Neural Networks Learn. Syst. 31 (2) (2020) 383–395.
[24] X. Zhang, Y. Zhuang, W. Wang, W. Pedrycz, Transfer boosting with synthetic instances for class imbalanced object recognition, IEEE Trans. Cybern. 48 (1) (2018) 357–370.
[25] Y. Freund, R.E. Schapire, Experiments with a new boosting algorithm (1996) 148–156..
[26] L. Chen, S. Zhou, Sparse algorithm for robust lssvm in primal space, Neurocomputing 275 (2018) 2880–2891.
[27] H. Xiao, H. Xiao, C. Eckert, Adversarial label flips attack on support vector machines (2012) 870–875.
[28] B. Biggio, B. Nelson, P. Laskov, Support vector machines under adversarial label noise 20 (2011) 97–112.

[29] A. Paudice, L. Munozgonzalez, E. Lupu, Label sanitization against label flipping poisoning attacks, arXiv: Machine Learning..
[30] P. Chan, D. Yeung, W. Ng, C.-M. Lin, J. Liu, Dynamic fusion method using localized generalization error model, Inf. Sci. 217 (2012) 1–20, https://doi.org/10.1016/j.ins.2012.06.026.
[31] C. Fan, L. Zeng, Y. Feng, G. Cheng, J. Huang, Z. Liu, A novel learning-based approach for efficient dismantling of networks, Int. J. Mach. Learn. Cybern. 11 (9) (2020) 2101–2111, https://doi.org/10.1007/s13042-020-01104-8.
[32] X. Wang, H. Xing, Y. Li, Q. Hua, C. Dong, W. Pedrycz, A study on relationship between generalization abilities and fuzziness of base classifiers in ensemble learning, IEEE Trans. Fuzzy Syst. 23 (5) (2015) 1638–1654.
[33] X. Wang, R. Wang, C. Xu, Discovering the relationship between generalization and uncertainty by incorporating complexity of classification, IEEE Trans. Cybern. 48 (2) (2018) 703–715.
[34] J. Valyon, G. Horvath, A robust ls-svm regression, World Acad. Sci., Eng. Technol. Int. J. Comput. Electr. Autom. Control Inf. 1 (7) (2007) 2237–2242.
[35] L. You, L. Jizhen, Q. Yaxin, A new robust least squares support vector machine for regression with outliers, Procedia Eng. 15 (2011) 1355–1360.
[36] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, A. Kurakin, On evaluating adversarial robustness, arXiv: Learning..
[37] N. Akhtar, A. Mian, Threat of adversarial attacks on deep learning in computer vision: a survey, IEEE Access 6 (2018) 14410–14430.
[38] C. Yu, J. Wang, Y. Chen, X. Qin, Transfer channel pruning for compressing deep domain adaptation models, Int. J. Mach. Learn. Cybern. 10 (11) (2019) 3129–3144.
[39] Y. Wen, Y. Qin, K. Qin, X. Lu, P. Liu, Online transfer learning with multiple decision trees, Int. J. Mach. Learn. Cybern. 10 (10) (2019) 2941–2962.
[40] D. Dua, C. Graff, UCI machine learning repository (2017). URL:http://archive.ics.uci.edu/ml..
[41] Apache spamassassin. URL: https://spamassassin.apache.org/index.html..
[42] N. Abdelhamid, A. Ayesh, F. Thabtah, Phishing detection based associative classification data mining, Expert Syst. Appl. 41 (13) (2014) 5948–5959.