A Scalable Mixture Model Based Defense Against Data Poisoning Attacks on Classifiers*

Xi Li, David J. Miller, Zhen Xiang, and George Kesidis

School of EECS, The Pennsylvania State University
University Park, PA, 16802
{xzl45,djm25,zux49,gik2}@psu.edu

Abstract. Classifiers, e.g., those based on Naive Bayes, a support vector machine, or even a neural network, are highly susceptible to a data-poisoning attack. The attack objective is to degrade classification accuracy by covertly embedding malicious (labeled) samples into the training set. Such attacks can be mounted by an insider, through an outsourcing process (for data acquisition or training), or conceivably during active learning. In some cases, a very small amount of poisoning can result in dramatic reduction in classification accuracy. Data poisoning attacks are successful mainly because the malicious injected samples significantly skew the data distribution of the corrupted class. Such attack samples are generally data outliers and in principle separable from the clean samples. We propose a defense that: 1) uses a mixture model both to well-fit the (potentially multi-modal) data and to give potential to isolate attack samples in a small subset of the mixture components; 2) performs hypothesis testing to decide both which components and which samples within those components are poisoned, with the identified poisoned ones purged from the training set. Our approach addresses the attack scenario where adversarial samples are an unknown subset embedded in the initial training set, and can be used to perform data sanitization as a precursor to the training of any type of classifier. The promising results for experiments on the TREC05 spam corpus and Amazon reviews polarity dataset demonstrate the effectiveness of our defense strategy.

Keywords: Adversarial learning · Data poisoning attack · Mixture modeling · Parsimonious mixtures · Spam filter · Sentiment analysis.

1 Introduction

Machine learning systems are vulnerable to inputs crafted by an adversary. Interest in adversarial learning (AL) has grown dramatically in recent years, focusing on devising attacks against machine learning models and defenses against such attacks. Three important types of AL attacks [9] are: data poisoning (*e*.g., [22,2,6,21]), test-time evasion (*e*.g., [1,15,18,11]) and reverse engineering (*e*.g.,

* This research is supported in part by an AFOSR DDDAS grant and a Cisco Systems URP gift.

[19,14,20]). In this work, we address data poisoning attacks on classifiers, *e*.g., those based on a support vector machine (SVM), a Bayesian network, or neural network, that seek to degrade classification accuracy. The proposed defense can be widely deployed to sanitize the training set pre-training.

Data poisoning (DP) is an effective attack to mislead a classifier. DP is able to dramatically degrade accuracy by adding relatively few maliciously crafted data points to a training set. To expand on this point, first note that a DP attack may be done before data collection to form the training set. For the example of a spam filter, emails sent from known malicious IP addresses are regarded as spam, and an attacker may generate and send emails that are more representative of ham via blacklisted IP addresses to pollute the spam training set. For a sentiment analysis problem, an attacker could insert positively labeled samples with words conveying very negative sentiments [13] to misguide the model of positive sentiment. Second, existing classifiers are highly vulnerable to data poisoning attacks. The malicious samples injected by an adversary will skew the data distributions of the corrupted class in a manner confounding the class discriminating features. Once the training set is poisoned with sufficient malicious samples, the overall classification accuracy will be significantly degraded.

Defenses against DP attacks on various systems include [17,8,12]. [17] constructs approximate upper bounds on the loss across a family of DP attacks, for defenders that first perform outlier removal followed by empirical risk minimization. [8] proposes a defender named Curie to protect an SVM from a DP attack. [8] assumes that the malicious data are crafted by maximizing the loss function of an SVM by flipping labels of legitimate samples. Therefore such data are similar to the normal points from the opposite set. If an additional dimension, the class label, is added to the feature space, the attack samples will be separated from the benign ones. [8] is not a generalized defense strategy as its assumption on malicious samples is specific to an SVM. A Reject on Negative Impact (RONI) strategy was proposed in [12] – although this work specifically focuses on spam filtering, it generalizes to defenses of classifiers on any domain. The defender in [12] rejects putative additional training samples if trial-adaptation of the spam model based on use of these samples causes degradation in classification accuracy on a held-out validation set. This strategy makes two strong assumptions: 1) that there is sufficient labeled data to have a held-out validation set; and 2) that the classifier has already been trained on "clean" data, with the attack consisting of additional labeled samples for classifier retraining. [12] is not a practical strategy when the malicious data are embedded within the original training set (with the attack samples an unknown subset). This more difficult scenario is addressed here.

A potential strategy for designing a **generalized** defender against an **embedded** data poisoning attack is to conduct "data sanitization" on the training set, *i*.e., identifying and removing the attack samples as training set outliers. While such ideas are mentioned in [7] and are related to [10], we are not aware such ideas have been practically, effectively applied. Again, a main reason why the victim classifiers are susceptible to data poisoning attacks is that, under the corrupted class, the training/model estimation is degraded in an unimpeded fashion by the planted malicious "outliers". **Hence, we propose a mixture based outlier detection method, both to well-fit the (potentially multi-modal) data and to allow potential concentration and isolation of poisoned samples in a small subset of the mixture components.** The validity of our defense is demonstrated by experimental results on the TREC05 spam corpus and Amazon reviews polarity dataset.

This paper is organized as follows. We first define the threat model of data poisoning attacks in Section 2. In Section 3, we propose our mixture based defense against data poisoning attacks on generative classifiers. Experimental results are presented in Section 4. Finally, we conclude our work and discuss future work in Section 5.

2 Threat Model of Data Poisoning Attack

In this section, we build the threat model of data poisoning attacks against statistical classifiers, including the knowledge and objective of attacker and defender, as well as the plausible attack scenarios.

2.1 Notation

We consider a D-dimensional feature space, with a sample represented as a vector $\underline{x} = (x_1, x_2, \ldots, x_D)$, where x_d can be discrete valued, such as word counts if $\underline{x}$ is text data, following a multinomial distribution, or can be continuous valued, *e*.g., $\underline{x}$ is generated via a multivariate Gaussian distribution. The dataset is then represented by a fixed (high) dimensional (possibly highly sparse) feature matrix – e.g., many words may have zero occurrences in a given document, in which case the vector will be sparse. Let $\mathcal{X}_p = \{\underline{x}_i^p, i = 1, \ldots, N_p\}$ be a given training set of positive samples used to build a multi-component *positive-class* model. Likewise, let $\mathcal{X}_n = \{\underline{x}_i^n, i = 1, \ldots, N_n\}$ be the negative training set for building a multi-component *negative-class* model. For concreteness, we consider binary classification here. It is possible to extend our work to a multi-class (> 2 classes) classifier.

2.2 Threat Model

Attacker's assumption and goal: We assume that an attacker: 1) has full knowledge of the learning approach and classification framework, *e*.g., a standard naive Bayes (NB) classifier; 2) does not need access to the clean training set as poisoning is done before data collection to form the training set; 3) only pollutes the negative training set, *e*.g., polluting spam set of a spam filter by sending ham emails via known blacklisted IP addresses, or poisoning negative set in sentimental analysis on product reviews by verbally recommending a product but giving it a low numerical rating; 4) is unaware of any deployed defense.

The **goal** of an attacker is to decrease classification accuracy with as few attack samples as possible.

Defender's assumption and goal: The defender presumes that: 1) The positive set is untouched (clean); 2) it is unknown whether the negative set is corrupted and if so, which is the attack subset of samples. The defender **aims** to: 1) identify and remove attack samples, as many as possible, before classifier training/retraining; 2) maintain the classification accuracy as high as that of a classifier trained without data poisoning.

Attack scenarios: Our approach is applicable to two attack scenarios: 1) classifier retraining and 2) classifier training, where the latter is more challenging. In the retraining scenario, one can initially build clean positive and negative models (those uncorrupted by attack) using $\mathcal{X}_p$ and $\mathcal{X}_n$, respectively. Let us denote a batch of additional samples that are treated as labeled negative by $\tilde{\mathcal{X}_n} = \{\tilde{\underline{x}_i}, i = 1, \ldots, \tilde{N_n}\}$. In the retraining case, the learner pools $\tilde{\mathcal{X}_n}$ with $\mathcal{X}_n$, retraining the negative model using the combined data pool $\mathcal{X}_{nc} = \{\mathcal{X}_n, \tilde{\mathcal{X}_n}\}$. Note that $\tilde{\mathcal{X}_n}$ may consist of legitimate negative samples, attack samples, or some combinations of the two. If one can utilize a separate, uncorrupted, held-out validation set, the approach in [12] can effectively mitigate an attacking $\tilde{\mathcal{X}_n}$. However, consider the other scenario – the training scenario. Unlike retraining, where the subset $\tilde{\mathcal{X}_n}$ is known to the learner, in the training scenario the attack samples are embedded amongst the clean negative samples. The learner does not know whether an attack is present and if so, which is the attack sample subset. Again the learner uses $\mathcal{X}_{nc}$, but in this case to perform the inaugural learning of the negative model, not model retraining. In the sequel, we develop a mixture based defense strategy, which effectively defeats the attack under the more challenging classifier training scenario.

3 Mixture Model Based Defense Against Data Poisoning Attack

3.1 Mixture Modeling

To defend a data poisoning attack, a "data sanitization" strategy will be applied, *i*.e., identifying and removing attack samples as training set "outliers". Such outliers are conjectured to form disjoint subpopulations from the normal negative samples in the feature space. Mixture modeling is a sound approach for seeking to concentrate (and thus isolate) poisoned samples in a few components, which can assist in accurately identifying and removing them. Accordingly, we build a mixture model based defender to distinguish the distribution of attack samples from benign ones.

The mixture representation of $\underline{x}$ for an M_k-component mixture model is given by (1), where $k \in \{p, nc\}$ represents positive/negative class, M_k is the number of components, and $\Theta(M_k) = \{\{\alpha_{j|k}\}, \{\theta_{j|k}\}\}$ denotes the parameters at model order M_k.

$$P[\underline{x}|\Theta(M_k)] = \sum_{j=1}^{M_k} \alpha_{j|k} P[\underline{x}|\theta_{j|k}] \quad (1)$$

Here, $\{\alpha_{j|k}\}$ are component masses which satisfy $0 \leq \alpha_{j|k} \leq 1$ and $\sum_{j=1}^{M_k} \alpha_{j|k} = 1$, and $\theta_{j|k}$ is the set of parameters specifying the joint probability mass function (PMF) for component j under model k. Model order M_k is selected by minimizing the Bayesian Information Criterion (BIC) cost [16], and the model parameters $\Theta(M_k)$ are estimated by the Expectation-Maximization (EM) algorithm [4].

Detection Inference: To significantly mislead a classifier with fewest attack samples, the adversary skews the distribution of the negative samples in a manner confounding the positive and negative discriminating features. Hence, the attack samples should be more representative of positive samples and present **atypicality** with respect to the negative set. Log mixture likelihood, $\log(P[\underline{x}|\Theta(M_k)])$, measures how typical a sample is to class k and therefore is used as the statistic for detection inference. (To avoid underflow, we use log-likelihood rather than likelihood.) Herein, we propose the null hypothesis of our detection inference, that the negative training set is not poisoned, *i*.e., all negative training samples are generated according to the null distribution. Alternatively, if the data poisoning attack exists, the negative training set is generated by an alternative model, which is a mixture of negative and positive distributions. The test statistic is to check on which side of the positive-negative boundary, $\log(P[\underline{x}|\Theta(M_p)]) = \log(P[\underline{x}|\Theta(M_{nc})])$, a negative sample $\underline{x}$ resides. If $\log(P[\underline{x}|\Theta(M_p)]) > \log(P[\underline{x}|\Theta(M_{nc})])$, then $\underline{x}$ is better explained by the positive model. Thus, we reject the null hypothesis, and the negative training set is deemed contaminated (poisoned). Otherwise, we accept the null hypothesis.

Implementation: We first separately apply mixture modeling to both the positive set $\mathcal{X}_p$ and negative set $\mathcal{X}_{nc}$ to compute the model parameters $\Theta(M_p)$ and $\Theta(M_{nc})$. Then, we do detection inference on negative samples for a given model component. The component label on a negative sample $\underline{x}$, $j^* = \arg\max_j P[j|\underline{x}, \Theta(M_{nc})]$, can be obtained from the E-step of EM-learning, and is fixed during detection.

To weaken the impact brought by the attack components on likelihood evaluation, suspect components in the negative model are pruned one by one. The components are traversed by the component score, which is the average sample log-likelihood under the positive model. For example, the score of component i with samples $X_i \subset \mathcal{X}_{nc}$ (those samples MAP-assigned to component i) is defined as $\frac{1}{|X_i|}\log(P[X_i|\Theta(M_p)])$. If a negative component is fundamentally formed by attack samples, it gets a higher score as its samples are more positive-representative. Otherwise, it gets a lower score. Then, from the highest to the lowest-scored negative component, we perform the following detection and response:

1. A sample $\underline{x}$ in the current component i is rejected if it is more likely positive than negative with margin of m. Here $m \in [0, 1]$ is a hyper-parameter for

relaxing this decision, because benign samples may be close to the positive-negative boundary. In other words, the suspicious subset $\tilde{X}_i = \{\underline{x} \in X_i | \log(P[\underline{x}|\Theta(M_p)]) > m \log(P[\underline{x}|\Theta(M_{nc}-1)])\}$ is removed from X_i (and also from $\mathcal{X}_{nc}$). To avoid bias if component i is contaminated, the model used to evaluate negative likelihood is the re-weighted mixture excluding component i, i.e., $\Theta(M_{nc}-1) = \{\{\frac{\alpha_{j|nc}}{\sum_{j' \neq i} \alpha_{j'|nc}}\}, \{\theta_{j|nc}\} | \forall j \neq i\}$.

2. Denoting the surviving samples of component i as $X'_i = X_i \backslash \tilde{X}_i$, the component parameters $\theta_{i|nc} = \arg\max_\theta \log(P[X'_i|\theta])$ are re-estimated, and updated component weights become $\alpha_{j|nc} = |X'_j|/|\mathcal{X}_{nc}|$ for $j = 1...M_{nc}$. Here, $X'_j = X_j$ for the negative components j not yet visited.
3. Evaluate the BIC cost of the current negative mixture, $BIC(\Theta(M_{nc}), \mathcal{X}_{nc})$, and that of the re-weighted negative mixture with component i pruned, $BIC(\Theta(M_{nc}-1), \mathcal{X}_{nc})$.
4. If the BIC cost decreases $(BIC(\Theta(M_{nc}), \mathcal{X}_{nc}) > BIC(\Theta(M_{nc}-1), \mathcal{X}_{nc}))$, we prune component i and re-weight the remaining components, i.e., update $\Theta(M_{nc})$ by $\Theta(M_{nc}-1)$ and the optimal model order M_{nc} by $M_{nc}-1$.

As mentioned previously, we use log-likelihood rather than likelihood to avoid underflow. However, underflow still exists if we simply take the log-likelihood of sample $\underline{x}$ as $\log(\sum_{j=1}^{M_k} \alpha_{j|k} P[\underline{x}|\theta_{j|k}])$. Thus, we compute log mixture likelihood by (2), where it is the sum of the expected complete data log likelihood and entropy of soft component assignments $(P[j|\underline{x}, \Theta(M_k)])$ [24]:

$$\begin{aligned} \log P[\underline{x}|\Theta(M_k)] = &\sum_{j=1}^{M_k} P[j|\underline{x}, \Theta(M_k)] \log\left(\alpha_{j|k} P[\underline{x}|\theta_{j|k}]\right) \\ &- \sum_{j=1}^{M_k} P[j|\underline{x}, \Theta(M_k)] \log\left(P[j|\underline{x}, \Theta(M_k)]\right) \\ &\forall \underline{x} \in \mathcal{X}_{nc}, k \in \{p, nc\} \end{aligned} \tag{2}$$

3.2 Parsimonious Mixture Model (PMM) Framework

The standard mixture model is not a feasible solution for a high dimensional feature space. It maintains an independent parameter for every feature in each component, which causes the dominance of model complexity in the BIC cost. This leads to gross underestimation of the model order for high D [5], with a standard mixture model choosing a single component for a high dimensional dataset. PMMs [5] solve this fundamental problem by introducing shared parameters which represent feature distributions common to all components. Model complexity is then determined by the number of components and the number of unique parameters in each component. Therefore model complexity does not dominate the BIC cost any more, and it is possible for a PMM to select a proper model order for a high dimensional dataset.

Maximum likelihood estimation of the parsimonious multi-component mixture can be performed via a generalized application of the EM algorithm (GEM)

[5]. With the assumption on independence of features conditioned on the component of origin, we rewrite the likelihood under the parsimonious mixture as

$$P[\underline{x}|\Theta(M)] = \sum_{j=1}^{M} \alpha_j \prod_{d=1}^{D} P[x_d|\theta_j]^{v_{jd}} P[x_d|\theta_s]^{(1-v_{jd})} \tag{3}$$

where $\Theta(M) = \{\{\alpha_j\}, \{\theta_j\}, \theta_s, \{v_{jd}\}\}$ is the model parameters at order M: M is the number of components, $\{\alpha_j\}$ are component masses satisfying $0 \le \alpha_j \le 1$ and $\sum_{j=1}^{M} \alpha_j = 1$, $P[x_d|\theta_j]$ and $P[x_d|\theta_s]$ are component-specific and shared distributions, respectively, and $v_{jd} \in \{0, 1\}$ is the switch between component-specific and shared distribution for feature d. The model is initialized with $M = M_{\max}$ (chosen to overestimate the true number of clusters), and the component with smallest mass is pruned in each iteration until $M = 1$. For each model order we perform GEM learning to optimize model parameters. The one which yields the least BIC cost is the optimal model order M, and the parameters associated with M are the optimal parameters. For brevity, we omit the detailed EM learning process and derivation of model parameters, which can be found in [5].

Component pruning and component parameter re-estimation in PMMs is similar to the method of the previous subsection. For example, to prune component i, the model order M is decremented to $M-1$, and $\Theta(M) = \{\{\frac{\alpha_j}{\sum_{j' \neq i} \alpha_{j'}}\}, \{\theta_j\}, \theta_s, \{v_{jd}\} | \forall j \neq i\}$. For re-estimating parameters of component i by its surviving samples, we only update component-specific parameters $\theta_i = \arg\max_\theta \log(P[X_i'|\theta])$, and the other parameters (θ_s) are untouched. PMMs are used in all our experiments.

4 Experiments

In this section we provide convincing experiments for our proposed defense. We first introduce the datasets and target classifier applied, and then demonstrate the effectiveness of an embedded data poisoning attack – "pure-positive" poisoning attack. Finally we present and analyze the performance of our defender against such adversarial attacks.

4.1 Experiment Setup and Evaluation Criterion

Datasets: The experiments were conducted on two datasets, TREC 2005 spam corpus (TREC05) [3] and Amazon reviews polarity dataset (Amazon Reviews) [23]. We choose these datasets since spam filters and sentimental analysis are common victims of DP attacks. TREC05 includes real ham and spam emails which are labeled based on the sender/receiver relationship. The Amazon Reviews dataset contains product reviews spanning 18 years which are labeled positive/negative sentiment by associated user ratings. For TREC05, the training set contains 8651 ham and 8835 spam emails, and the (exclusive) test set consists of 2861 ham and 2968 spam emails. The dictionary, following case normalization, stop word removal, stemming and low-frequency word filtering, has around 30000 unique words. As for the Amazon Reviews, the training set contains 50000 positive reviews and 50000 negative reviews, and the (exclusive) test set consists of 10000 positive and negative reviews, respectively. The dictionary, following the same text preprocessing procedure, has roughly 11000 distinct features.

Target classifier: Since standard Naive Bayes is effective in spam filtering and sentimental analysis, we choose it as the target classifier of a data poisoning attack. It distinguishes positive samples from negative ones. In our experiment, positive samples are ham emails (positive reviews) in TREC05 (Amazon Reviews) and, accordingly, negative samples are spam emails (negative reviews).

Attack: As a reasonable and potent embedded data poisoning attack, we launch a "pure-positive" poisoning attack on the target classifier, where real positive samples are added into the negative training set with various attack strengths, *i*.e., the number of injected positive samples.

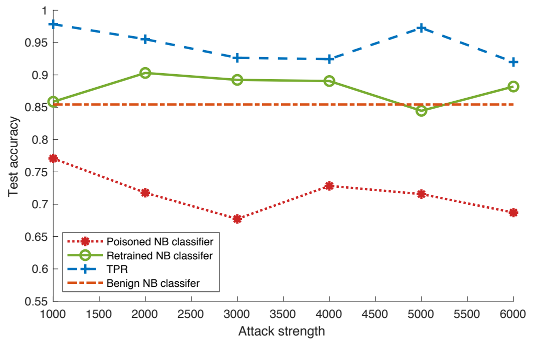
Evaluation criterion: Under both datasets, we first train a benign NB classifier, then poison the negative set with various strengths and validate the impact on test accuracy brought by the "pure-positive" attack. We next deploy our mixture based defender on the corrupted training set and measure its performance by 1) improvement in classification accuracy after retraining, 2) true positive rate (TPR) — the fraction of poisoned samples that are detected, and 3) false positive rate (FPR) – the fraction of non-poisoned samples falsely detected.

4.2 "Pure-positive" Poisoning Attack

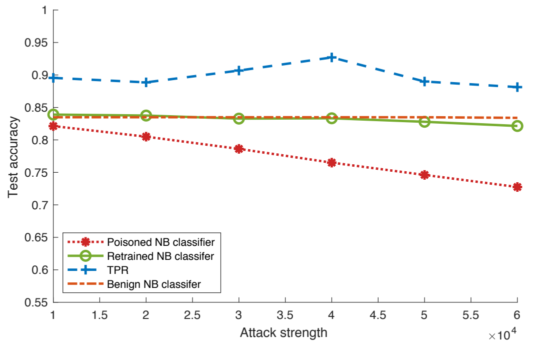
We conduct 6 "pure-positive" poisoning attacks on TREC05 with strength from 1000 to 6000, and on Amazon Reviews with strength from 10000 to 60000, respectively. The resulting test accuracies are plotted in Figures 1a and 1b as "poisoned NB classifier". Initially, when there is no attack, the benign NB classifier has test accuracy of 0.85 and 0.83 on TREC05 and Amazon Reviews, respectively. On TREC05, as the attack is strengthened to 3000 ham emails, the test accuracy drops rapidly – from 0.85 to 0.67. Similarly, on poisoned Amazon Reviews, the classification accuracy falls continuously (from 0.83 to 0.727) as the number of adversarial reviews is increased to 60000. In both cases, roughly half of the positive test samples are misclassified as negative. Thus, embedding "pure-positive" samples into the negative set is indeed a strong attack on a standard NB classifier.

4.3 Mixture Model Based Defense

Following the detection methodology proposed in the last section, our defender locates and purges all suspicious data in both datasets, with m set to 0.9 on TREC05 and 0.95 on Amazon Reviews. Then the NB classifier is retrained on the purged datasets, and the corresponding performance is shown as the solid line in Figure 1a and 1b. As expected, the retrained classifier performs well and stable under all attack cases, with accuracy of around 0.89 on TREC05 and 0.83 on Amazon Reviews. The reason why the classifier retrained on purged TREC05 outperforms the one trained on the *clean* dataset is discussed below. Besides, the average true positive rates of the defender in both experiments, 0.946 in TREC05 and 0.898 in Amazon Reviews, are relatively high, which demonstrates the validity of our detection method in practice.



(a)



(b)

Fig. 1: Performance of mixture-based defender against “pure-positive” attacks on (a) TREC05 and (b) Amazon Reviews

Although our defense strategy succeeds in defending against a data poisoning attack, making the retrained NB classifier achieve high classification accuracy – at least comparable to the benign classifier – the false positive rate is relatively poor – **in the range of 0.12-0.37**. Moreover, as mentioned before, the standard NB retrained on purged TREC05 outperforms the one trained on the clean dataset. Given these observations, for different attack strengths, we train a standard NB classifier on the “clean” datasets where false detected training spam (*i*.e., samples which have ground truth labels of spam but are detected as ham) are kept, and training ham are untouched. In each case, there are roughly

4000 training spam and 8000 training ham. Not surprisingly, on the same test set, this "benign" classifier only achieves an average test accuracy of 0.72, with the averaging over all considered attack strengths. Nearly half of the test ham are misclassified as spam and 90% of test spam are classified correctly, indicating that the false detected samples (real spam) have similar distributions as ham emails. Otherwise more test data will be classified as ham considering the greater apriori amount of ham than spam. **Hence, our method performs effective defense against "pure-positive" poisoning attacks, as the unsatisfying false positive rate is actually rooted in the inherent impurity of TREC05.**

5 Conclusions and Future Work

In this work, we proposed a mixture model based defense against data poisoning attacks against classifiers where attack samples are an unknown subset embedded in training set. We successfully launched defenses against "pure-positive" attacks on datasets TREC05 and Amazon Reviews. The experiments on two completely different datasets demonstrate the effectiveness of our defender under strong attacks. Our approach is a generalized defense strategy that is applicable to defend against embedded data poisoning attacks on any classifier – it can remove malicious data before they corrupt (discriminative) training of a deep neural network or support vector machine based classifier.

We have considered attacks which corrupt the negative set. Our approach could also be applied if the attack targets positive data, rather than negative data. However, our assumption towards the adversary is strong – we assume the attacker only pollutes one of the training sets, either positive or negative. In general, the attacker may simultaneously poison both positive and negative sets with different strengths. Defending against such attacks is a good subject for future work.

References

1. Biggio, B., Corona, I., Maiorca, D., Nelson, B., Srndic, N., Laskov, P., Giacinto, G., Roli, F.: Evasion attacks against machine learning at test time. In: Machine Learning and Knowledge Discovery in Databases - European Conference,ECML PKDD (2013)
2. Chen, X., Liu, C., Li, B., Lu, K., Song, D.: Targeted backdoor attacks on deep learning systems using data poisoning. Arxiv (2017), http://arxiv.org/abs/1712.05526
3. Cormack, G.V., Lynam, T.R.: Trec 2005 spam public corpora. https://plg.uwaterloo.ca/~gvcormac/trecspamtrack05 (2005)
4. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological) (1977)
5. Graham, M.W., Miller, D.J.: Unsupervised learning of parsimonious mixtures on large spaces with integrated feature and component selection. IEEE Trans. Signal Processing (2006)

6. Gu, T., Liu, K., Dolan-Gavitt, B., Garg, S.: Badnets: Evaluating backdooring attacks on deep neural networks. IEEE Access **7**, 47230–47244 (2019)
7. Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I.P., Tygar, J.D.: Adversarial machine learning. In: Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence (2011)
8. Laishram, R., Phoha, V.V.: Curie: A method for protecting SVM classifier from poisoning attack. Arxiv (2016), http://arxiv.org/abs/1606.01584
9. Miller, D.J., Xiang, Z., Kesidis, G.: Adversarial learning targeting deep neural network classification: A comprehensive review of defenses against attacks. Proceedings of the IEEE **108**(3), 402–433 (2020)
10. Miller, D.J., Browning, J.: A mixture model and em-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. IEEE Transactions on Pattern Analysis and Machine Intelligence (2003)
11. Miller, D.J., Wang, Y., Kesidis, G.: When not to classify: Anomaly detection of attacks (ADA) on DNN classifiers at test time. Neural Computation (2019)
12. Nelson, B., Barreno, M., Jack Chi, F., Joseph, A.D., Rubinstein, B.I.P., Saini, U., Sutton, C., Tygar, J.D., Xia, K.: Misleading learners: Co-opting your spam filter. In: Machine Learning in Cyber Trust: Security, Privacy, and Reliability. Springer, Boston (2009)
13. Newell, A., Potharaju, R., Xiang, L., Nita-Rotaru, C.: On the practicality of integrity attacks on document-level sentiment analysis. In: Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop, AISec (2014)
14. Oh, S.J., Augustin, M., Fritz, M., Schiele, B.: Towards reverse-engineering black-box neural networks. In: 6th International Conference on Learning Representations, ICLR (2018)
15. Papernot, N., McDaniel, P.D., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: IEEE European Symposium on Security and Privacy, EuroS&P (2016)
16. Schwarz, G.: Estimating the dimension of a model. The Annals of Statistics (1978)
17. Steinhardt, J., Koh, P.W., Liang, P.: Certified defenses for data poisoning attacks. In: Conference on Neural Information Processing Systems (2017)
18. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I.J., Fergus, R.: Intriguing properties of neural networks. In: 2nd International Conference on Learning Representations, ICLR (2014)
19. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction apis. In: 25th USENIX Security Symposium, USENIX (2016)
20. Wang, Y., Miller, D.J., Kesidis, G.: When not to classify: Detection of reverse engineering attacks on DNN image classifiers. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP (2019)
21. Xiang, Z., Miller, D.J., Kesidis, G.: A benchmark study of backdoor data poisoning defenses for deep neural network classifiers and A novel defense. In: 29th IEEE International Workshop on Machine Learning for Signal Processing, MLSP (2019)
22. Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., Roli, F.: Support vector machines under adversarial label contamination. Neurocomputing (2015)
23. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Proceedings of the 28th International Conference on Neural Information Processing Systems (2015)
24. Zhao, Q., Miller, D.J.: Mixture modeling with pairwise, instance-level class constraints. Neural Computation (2005)