# Distributed SVM

Luojie Xiang, Shunrang Cao

## I. Introduction

Support Vector Machines (SVM) is an important machine learning algorithm. The basic SVM is a binary classifier which seeks to find a hyperplane that separates data points into two classes. SVM is a max-margin classifier which tries to maximize the distance of the hyperplane to the boundary points of each class. The boundary points are called support vectors and thus the name support vector machine.

SVM has many advantages over other machine learning algorithms. Its decision relies only on support vectors. This means, once trained, it make future predictions very quickly due to the few amount of support vectors. The training itself is a well-formed convex optimization problem so that convergence is guaranteed.

The idea of the project is to design and implement a distributed SVM with the following two requirements:

- Faster training than standard SVM
- Better resiliency towards malicious attack

This work focus on malicious attack since adversarial machine learning has been a field of increasing interest. It addresses the question of learning in the presense of malicious noise. The practical scenario is that, nowadays many machine learning algorithms are used in practical online systems to provide a service or be a part of a larger system. For example, Topsy lab used machine learning algorithm in sentiment analysis to successfully predict a Netflix stock price drop based on twitter. Others have machine learning algorithm in election prediction, failure prediction in distributed systems etc. All these applications have two characteristics in common:

- Publicly available training set.
  Many applications draw training set from social media. Text analysis is a mature area and social media forms a very rich corpus that can support many data mining needs.
- Online stream training/prediction.
  At the very beginning, machine learning algorithms are mostly trained offline, once and for all. Then the model is used for prediction. This is found insufficient now since corpus change as people would talk about different things at different times on twitter. One single fixed training set is very hard to capture these changing trends. Therefore, most system now collects training set (from social media) frequently to update their model so that the model would not be obsolete and could still capture the important statistics in the current corpus.

The above characteristics raises a vulnerability for such a system since the public accessibility of the training set (drawed from social media). Adversary can craft malicious data points and post it to social media. They can almost always be sure to have their victim obtain their malicious point if they know what their victim is doing. For example, to confuse a classifier about Netfilx's sentiment, just stack a lot of positive words like happy with the word Netflix, and then label the data point with some frowny faces (many sentiment analysis system use such a distant supervised method to label their dataset, though noisy but provide good result anyways).

For the victim to avoid being attacked, he must use something better than a naive training algorithm. This is the purpose of this work to look into a defensive algorithm without introducing so much overhead. Of course besides malicious attack, crash-stop failures are an important problem. We don't focus on this failure model since the proposed algorithm can be implemented on other platforms that supports some fault tolerance such as Hadoop.

The attack model and attack algorithms are described in next two sections.

## II. Attack Model

This work assumes all computers can be trusted. The attack comes from the training dataset. The adversary has the following capabilities:

- Know the victim's dataset.
- Has computing power to create data points.
- Can insert data points into the victim's dataset.

This is a practical assumption. The difference of attackers comes in the algorithm they use to create data points. Different algorithms may result in different damage power on the victim's SVM accuracy.

Next session looks into attack algorithms.

## III. Attack Algorithm

Label flip attacks choose data points from the victim's benign training set, flip their labels and then insert them back into the victim's training set. It is a very popular attack and is well studied in literature [8].

This research focus on label flip attacks for the following reason:

- Label flip attack can be a very strong attack when the data points whose labels are to be flipped is chosen carefully.
- Label flip attack requires few computational power. Adversary only needs to choose data points and flip their label. However, huge damage can be done to victim's learner by this small amount of computation.
- Label flip attack represents a broad family of attacks that tries to put data points of a certain class into the other class's region. Optimization attacks are one example. It requires solving an optimization over a objective function

(e.g. maximize the loss function). It will result in a larger damage but require much more computation power and could be easily filtered out since they're normally very far away from the benign data points.

Our work picks three label flip attacks from a previous work with different strategy in terms of how points are picked. Difference in strategy results in the attack being stronger or weaker than others.

- Nearest-First Flip attack - train an SVM on victim's data set and pick the points nearest to the decision plane to flip label [8].
- Random Flip attack - randomly pick data points from victim's benign training set, and flip their labels [8].
- Furthest-First Flip attack - train an SVM on victim's data set and pick the furthest data point from the decision plane and flip its label [8].

### A. Nearest-First Flip attack

This attack picks the nearst points to the decision plane. Despite the fact that this is the weakest attack of the three, this attack hides the malicious points very close the the benign points. This tests the defense algorithm's robustness.

The attack works as follows:

- Step 1 - Train an SVM on benign data set.
- Step 2 - Apply the decision function of SVM on all data points of the benign data set.
- Step 3 - Sort the decision values and find the minimum in absolute value.
- Step 4 - Flip the chosen point and insert it back into the benign data set.

### B. Random Flip Attack

This attack picks a random point to flip label. It is stronger the the Nearest-First Flip attack but weaker than the Furthest-First Flip attack. The pro of this attack is that there's no need to train SVM on victim's dataset. A random pick is all computation that needs to be done. It is the fastest attack of the three and yield a good damage on victim's learner.

### C. Furthest-First Flip Attack

This attack picks the data point that is furthest from the SVM decision plane and flip its label. It is shown to generate a near-optimal attack effect on SVM [8]. Another benefit of this attack over the optimal attack is, it works on discrete feature space, since optimal attack involves solving convex optimization over loss function which requires a differentiable field. This limits optimal attacks to only continuous feature space. This attack is the strongest of the three.

The attack works as follows:

- Step 1 - Train an SVM on benign data set.
- Step 2 - Apply the decision function of SVM on all data points of the benign data set.
- Step 3 - Sort the decision values and find the maximum in absolute value.
- Step 4 - Flip the chosen point and insert it back into the benign data set.

## IV. SVM Training Algorithm

The design of SVM training algorithm should be parallelizable so that the training algorithm could be faster and has better tolerance against malicious attacks than standard SVM algorithm.

**1. Baseline: Ensemble Learning**

Ensemble Learning is a classic idea of improving any machine learning algorithm's resilliency against noise in the training dataset [1], [3]. It has also been used as a defense strategy against malicious attacks [2], [4].

Ensemble learning is used as our baseline, in which the training happens as follows:

- Step 1 - Create $N$ samples out of the training set by sampling with replacement.
- Step 2 - Scatter $N$ samples to $N$ machines.
- Step 3 - Train an SVM on each machine.
- Step 4 - All $N$ SVMs vote on new data points during prediction.

**2. A better Algorithm: Modified Adaptive Cluster**

Adaptive Cluster was proposed for reducing training set size for SVM [5], [6], based on the idea that, SVM's decision plane relies only on support vectors [7]. The process is,

- Step 1 - Cluster the training set.
- Step 2 - Train an initial SVM based on the representatives of the clusters (such as centroid).
- Step 3 - Train a final SVM using the clusters that contain support vectors in the initial SVM.

This method is modified to include resiliency against malicious attacks.

- Change 1 - After step 1 of original Adaptive Cluster algorithm, add a sanitization step, using either distance filtering, or active learning (explained later).
- Change 2 - Use the Adaptive Cluster algorithm in the ensemble framework. Step 3 in ensemble learning algorithm is changed to Adaptive Cluster algorithm above.

*Distance filtering* - Malicious data points will be far away from benign data points. For example, label flip attack [8] picks benign data, say positive class, flip the label to negative class and insert it back to the training set. These data points will reside in the positive area except now they all have negative label. Therefore, the malicious points (with negative label) will be very far away with the benign negative points. Thus, they're very likely to be put into a separate cluster if the negative points are clustered and this cluster is very likely to be very far away from the other negative clusters. Using some distance filtering, such as probability distribution threshold will identify the malicious cluster.

*Active Learning* - Active learning have machine learning algorithms summarizes the dataset and present to human experts the data points that it is most not sure of [9]. Human experts will label these very concise summaries and guide the learning process. Since in Adaptive Cluster algorithm, the representatives of each cluster is found, they can be convieniently used as the summary. Human expert will tell the learning algorithm which representatives are malicious.

The cluster whose representative is labeled malicious by the expert will be thrown away.

In this work, we only experiment with distance filtering since automated method is more desired (if it does work, and that is the case for this work).

## V. Experiment setup

This section first describes the dataset used and then the experiments.

### A. Dataset

A lot of datasets are made publicly available thanks to competitions such as KDD CUP, Kaggle and research groups such as UCI Machine Learning Repository, LIBSVM [12], [13], [14], [15]. The dataset described in Table V-A is selected. It is a realistic dataset (not synthesized). Therefore it has many undesired features such as imbalance between positive and negative class etc. However, the proposed algorithm should work with these features.

### B. Compare speed of standard SVM and ensemble SVM

The size of training data begins from 1000 and increases all the way to 30,000 with the step width of 1000. The time for training and f1 score are recorded for different size of training data.

### C. Evaluate attack resilience of standard SVM and ensemble SVM

The size of training data is fixed at 20,000, which only contains clean data. Attack points are added to clean training data. The number of attack points increases from 1000 to 20,000 with the step width of 1000. Both the f1 score and training time are recorded. The attacks are applied to both standard SVM and ensemble SVM. Three types of attacks are used: NFF, RFF, FFF.

### D. Evaluae attack relilience of adaptive cluster

The setting of training data and attack are the same as the previous experiment. They are applied to the method of adaptive cluster method. Both the f1 score and training time are recorded.

## VI. Results

This section presents the result of the experiments. The first subsection shows that of the ensemble method, which serves as our baseline. The second subsection shows that of the adaptive cluster method.

### A. Results of Ensemble Method

Fig. VI-A and Fig. VI-A shows the comparison between standard SVM and ensemble SVM training time and F1-score. We can conclude that, when dataset is very small, ensemble method has a lower F1-score but it achieves similar score when the dataset is large enough. Ensemble method can always have a much lower time than standard SVM training.

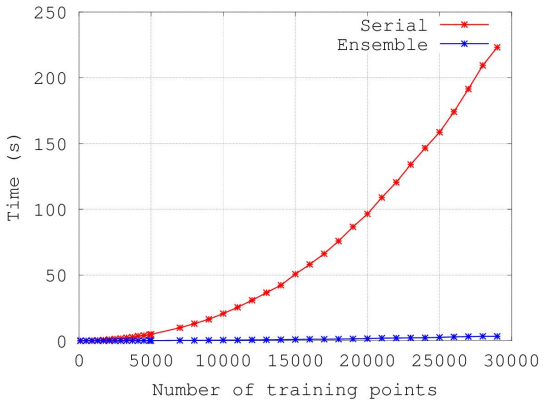


Fig. 1. Time of standard SVM and ensemble SVM training

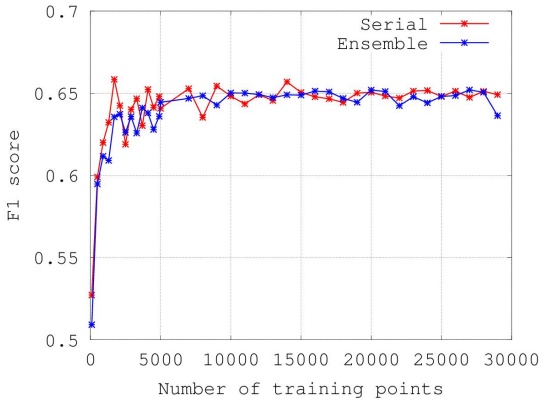


Fig. 2. F1-score of standard SVM and ensemble SVM training

### B. Standard SVM Under Attack

This section presents the result for three attacks on standard SVM to demonstrate the effect of each attack on victim's learner. Fig. VI-B shows that, FFF attack is the strongest attack and thus have the most damage on victim's learner. Following is the RF attack. The mildest is the NFF attack which barely does any damage to the victim's learner.

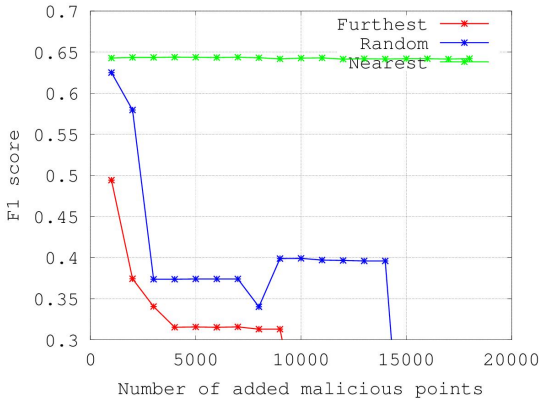


Fig. 3. F1-score of standard SVM training under Attack

TABLE I
DESCRIPTION OF DATASET

| Dataset | Source | # of classes | # of data (training/testing) | # of features (training/testing) |
|---|---|---|---|---|
| w8a | [16] | 2 | 49,749 / 14,951 | 300 / 300 |

### C. Ensemble Method Under Attack

The above section shows that, when dataset is large enough, ensemble method can achieve similar F1-score with standard SVM while have much lower training time. This section shows the results for ensemble method and standard SVM under attack.

Fig. VI-C shows the result of Nearest-First Flip attack. NFF attack is the mildest of the three and thus did not have much influence on F1-score of either standard SVM or Ensemble method. Fig. VI-C shows the result of Random-Flip attack. Fig. VI-C shows the result of Furthest-First Flip Attack. Both figure shows that ensemble has some marginal defense effect when the number of malicious point is small. However, when more malicious points are added, the defense effect disappears.

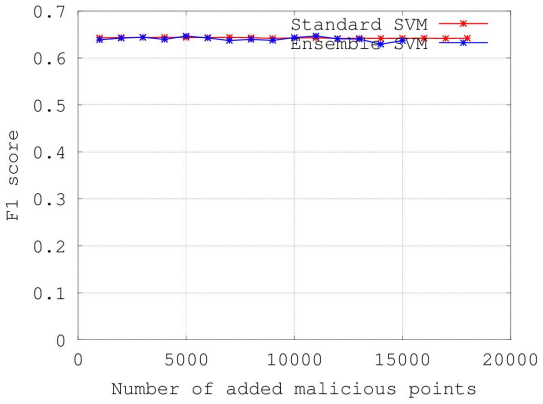


Fig. 4. F1-score of standard SVM and ensemble SVM training under NFF-Attack


Fig. 5. F1-score of standard SVM and ensemble SVM training under RF-Attack

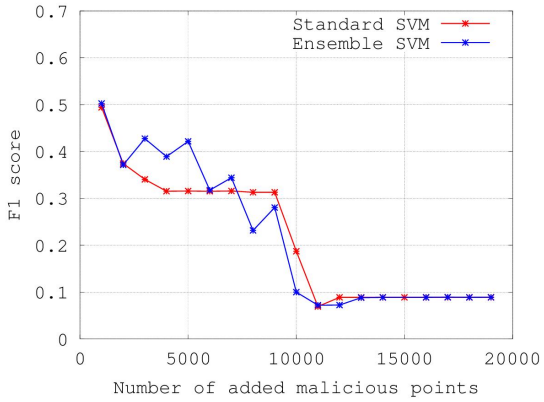


Fig. 6. F1-score of standard SVM and ensemble SVM training under FFF-Attack

### D. Adaptive Cluster Under Attack

This session shows the result of proposed method, adaptive cluster under attack, in Fig. VI-D, VI-D, VI-D. It is clear that, for both RF and FFF attack, adaptive cluster can have a good defense effect compared to both standard SVM and Ensemble SVM training. For NFF, since the attack can't do much damage on victim's learner, the defense effect is not obvious. However, when the number of malicious points is large enough, adaptive cluster's defense effect will disappear. This is expected since cluster and distance based filtering assumes that the majority is benign data points. If in an extreme context, the malicious points are the majority, clustering will filter out the benign points which would be an disaster.

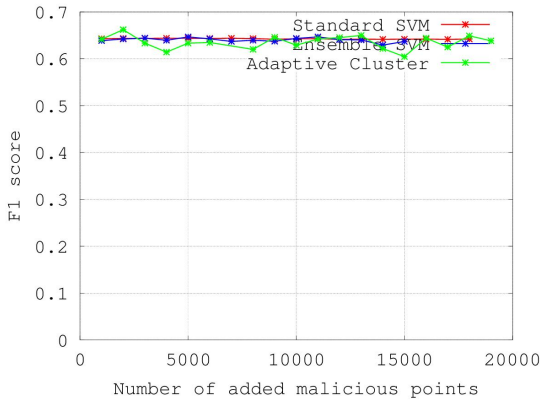


Fig. 7. F1-score of standard SVM, ensemble SVM and adaptive cluster training under NFF-Attack

Besides the good defense effect, it is important that the defense should come with a tolerable cost. Fig. VI-D shows that adaptive cluster does introduce overhead to the baseline

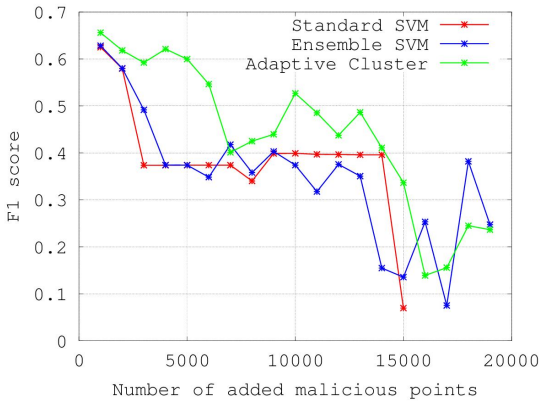


Fig. 8. F1-score of standard SVM, ensemble SVM and adaptive cluster training under RF-Attack

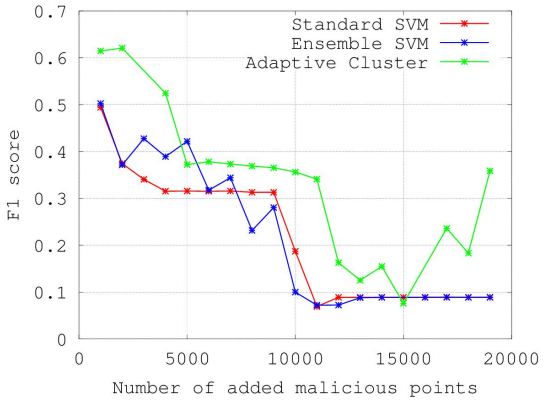


Fig. 9. F1-score of standard SVM, ensemble SVM and adaptive cluster training under FFF-Attack

ensemble method. However, with the added overhead, it still is much faster than standard SVM. Therefore, we think this overhead is tolerable.

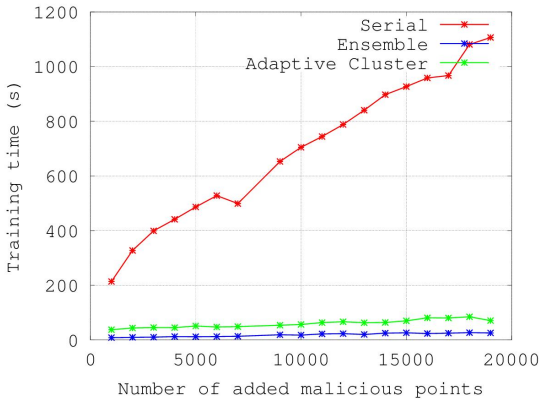


Fig. 10. Time of standard SVM, ensemble SVM and adaptive cluster training

## VII. Conclusions

This research proposed a modified adaptive cluster method to provide a defense against label flip attacks on SVM training. The modification is based on clustering and distance filtering to filter likely attack points. Through experiments, it is shown that the proposed method has good defense effect for the label flip attack. Compared with a baseline, ensemble learning, which is a proved method of improving accuracy and robustness of machine learning algorithms, it is shown that the proposed method has a much better resiliency against attack. The proposed method introduces some overhead over the baseline. However, it is still much faster than standard SVM training.

The proposed method, however, has the following limitations.

- The proposed method is only experimented on label flip attacks. There are other attacks that has a totally different mechanism with respect to label flip attack. The defense effect of the proposed method is not understood on those attacks.
- The proposed method is only a heuristically useful defense. However, no mathematical proof is given regarding the error bound for the defense. Therefore, there's no guarantee that, the proposed method will have good defense effect against label flip attack in any situation.

## References

[1] Breiman, Leo. "Bagging predictors." Machine learning 24, no. 2 (1996): 123-140.

[2] Barreno, Marco, Peter L. Bartlett, Fuching Jack Chi, Anthony D. Joseph, Blaine Nelson, Benjamin IP Rubinstein, Udam Saini, and J. Doug Tygar. "Open problems in the security of learning." In Proceedings of the 1st ACM workshop on Workshop on AISec, pp. 19-26. ACM, 2008.

[3] Dong, Yan-Shi, and Ke-Song Han. "Boosting SVM classifiers by ensemble." In Special interest tracks and posters of the 14th international conference on World Wide Web, pp. 1072-1073. ACM, 2005.

[4] Cretu, Gabriela F., Angelos Stavrou, Michael E. Locasto, Salvatore J. Stolfo, and Angelos D. Keromytis. "Casting out demons: Sanitizing training data for anomaly sensors." In Security and Privacy, 2008. SP 2008. IEEE Symposium on, pp. 81-95. IEEE, 2008.

[5] Boley, Daniel, and Dongwei Cao. "Training Support Vector Machines Using Adaptive Clustering." In SDM. 2004.

[6] Yu, Hwanjo, Jiong Yang, and Jiawei Han. "Classifying large data sets using SVMs with hierarchical clusters." In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 306-315. ACM, 2003.

[7] Koggalage, Ravindra, and Saman Halgamuge. "Reducing the number of training samples for fast support vector machine classification." Neural Information Processing-Letters and Reviews 2, no. 3 (2004): 57-65.

[8] Xiao, Han, Huang Xiao, and Claudia Eckert. "Adversarial Label Flips Attack on Support Vector Machines." In ECAI, pp. 870-875. 2012.

[9] Raghavan, Hema, Omid Madani, and Rosie Jones. "Active learning with feedback on features and instances." The Journal of Machine Learning Research 7 (2006): 1655-1686.

[10] Battista Biggio, Blaine Nelson, Pavel Laskov: Poisoning Attacks against Support Vector Machines. ICML 2012

[11] Newsome, James, Brad Karp, and Dawn Song. "Paragraph: Thwarting signature learning by training maliciously." In Recent advances in intrusion detection, pp. 81-105. Springer Berlin Heidelberg, 2006.

[12] http://www.sigkdd.org/kddcup/index.php

[13] http://www.kaggle.com/

[14] http://archive.ics.uci.edu/ml/

[15] http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

[16] John C. Platt. ”Fast training of support vector machines using sequential minimal optimization”. In Bernhard Schlkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, Advances in Kernel Methods - Support Vector Learning, Cambridge, MA, 1998. MIT Press.