# POSTER: Privacy Preserving Boosting in the Cloud with Secure Half-Space Queries

Shumin Guo, Keke Chen
Ohio Center of Excellence in Knowledge Enabled Computing
Department of Computer Science and Engineering
Wright State University, Dayton, OH 45435, USA
guo.18@wright.edu, keke.chen@wright.edu

## ABSTRACT

This poster presents a preliminary study on the PerturBoost approach that aims to provide efficient and secure classifier learning in the cloud with both data and model privacy preserved.

## Categories and Subject Descriptors

H.2.0 [**General**]: Security, integrity, and protection

## Keywords

Privacy, outsourcing data mining, cloud, RASP perturbation

## 1. INTRODUCTION

Most data mining tasks require a good understanding of the mining techniques, time-consuming parameter tuning, algorithm tweaking, and frequently algorithm innovation. They are often resource-intensive and need the expertise of applying data-mining techniques. As a result, most data owners, who have no sufficient computing resources or data-mining expertise, cannot mine their data.

The development of cloud computing and services computing enables at least two solutions. First, if the data owner has the data-mining expertise but not the computing resources, he/she can rent public cloud resources to process the data. Second, if the data owner does not have the expertise, he/she can outsource their data-mining tasks to data-mining service providers.

The Netflix prize is a successful story of outsourced data mining. The goal of the competition is to develop effective movie recommendation algorithms with the published Netflix data. Any interested person or team can attend the competition. Netflix rewards the winning teams based on the accuracy of their algorithms. In comparison, if developing in-house algorithms, Netflix may spend much more and possibly get nothing close to the winning algorithms.

In spite of all the benefits, the unprotected outsourcing approach has at least three drawbacks.

- The published data may contain private information [5], which actually forced Netflix to suspend the Netflix prize II competition[1].
- The data ownership is not protected. Once published, the dataset can be accessed by all the participants.
- The ownership of the resultant models is not protected. At least the model developer knows the model and understands how to use it.

Because the success of modern machine learning and data mining applications has largely depended on the available data, datasets are now precious properties to the data owners. With unprotected outsourcing, competitors can freely use the published data and the resultant models, and possibly derive knowledge against the data owner's interests. Data owners will soon realize that, without the protection on data and model privacy they will have to keep their data mining tasks in-house.

**Proposed Approach.** The proposed approach (PerturBoost) aims to address the above problem with the *secure half-space query* approach for classifier learning. Specifically, we will use secure half-space queries to mine a classification model from the data hosted in the cloud - the scenario is similar for using data-mining service providers.

This approach uses our previously developed RASP perturbation [1] that perturbs the data to protect the confidentiality, while still allowing users to conduct secure half-space queries. We utilize the boosting framework to build up a strong classifier with good prediction accuracy, based on a bunch of weak classifiers that have slightly better accuracy than random guess. These weak classifiers are constructed with RASP-based secure half-space queries.

In this way, we effectively address the problem of secure data mining in the cloud. (1) The data is protected with the RASP perturbation. (2) The model is protected in the form of secure half-space queries. (3) The accuracy is preserved with the boosting framework.

This approach has a couple of unique features. (1) It is very efficient, with low costs in storage, computation, and communication. (2) It provides sufficient security, if the user protects the perturbation parameters well.

The preliminary results show that the PerturBoost approach can learn models with satisfactory accuracy. An ongoing effort is to further reduce the cost and improve the accuracy of learning.

## 2. BACKGROUND

### 2.1 Classification Modeling

Classifier learning is to learn a model $y = f(x)$ from a set of training examples $\{x_i, y_i\}$, where $x_i \in \mathbb{R}^k$ is the $k$-dimensional feature vector describing an example, and $y_i$ is the label for the example - if we use '+1' and '-1' to indicate two classes, $y_i \in \{-1, +1\}$. The learning result is a function $y = f(x)$, i.e., given any known feature vector $x$, we can predict the label $y$ for the example $x$. The quality of the model is defined as the accuracy of

[1] http://blog.netflix.com/2010/03/this-is-neil-hunt-chief-product-officer.html

*CCS'12,* October 16–18, 2012, Raleigh, North Carolina, USA.
ACM 978-1-4503-1651-4/12/10.

prediction. A random guess to the two-class setting would have an accuracy around 50%.

Our approach is based on the boosting framework [2] for learning classifiers. A boosting model is a weighted summation of $n$ base classifiers, $f(x) = \sum_{i=1}^{n} \alpha_i h_i(x)$, where the base models $h_i(x)$ can be any *weak learner*, e.g., a learner with its accuracy significantly higher than 50% for two-class prediction, and $\alpha_i$ is the weight of $h_i(x)$. Both $\alpha_i$ and $h_i()$ are learning in the procedure.

Weak learner can be in any forms [4], among which a simple one is linear classifier. It can be represented as decision rules, such as:

```
if f(x) <0 then y=-1, otherwise y=1.
```

$f(x) = w^T x + b$ is a hyperplane, where $w \in \mathbb{R}^k$ and $b \in \mathbb{R}$ are to be learned from examples to achieve a good prediction accuracy. $f(x) < 0$ is also used as half-space query conditions, i.e., finding the records $x$ satisfying the condition $f(x) < 0$. This allows us to apply the RASP approach that was originally designed for secure half-space queries [1].

## 2.2 RASP perturbation

We assume that the RASP perturbation will only perturb the feature vectors $x_i$ of $\{x_i, y_i\}$, while leaving $y_i$ unchanged, which will not breach the data privacy. For each $k$-dimensional original vector $x_i$, the RASP perturbation[2] can be described in two steps.

1. The vector $x_i$ is extended to $d{+}2$ dimensions as $(x_i^T, 1, v_i)^T$, where $x_i^T$ is the transpose of $x_i$, the $(d+1)$-th dimension is always 1, and the $(d+2)$-th dimension, $v_i$, is drawn from a random number generator $RG$ that generates positive values from normal distributions.

2. The $(d+2)$-dimensional vector is further transformed to

$$p_i = RASP(x_i) = A(x_i^T, 1, v_i)^T, \qquad (1)$$

where $A$ is a $(d{+}2) \times (d{+}2)$ randomly generated invertible matrix with $a_{ij} \in \mathbb{R}$ such that there are at least two non-zero values in each row of $A$ and the last column of $A$ is non-zero.

$A$ is shared by all vectors, but $v_i$ is randomly generated for each individual vector. Note that the same $x_i$ can be mapped to different $p_i$ in the perturbed space due to the randomly chosen $v_i$, which provides necessary protection.

The RASP perturbation approach also includes a secure query transformation method to preserve half-space queries. A simple half-space query is like $X_j < a$, where $X_j$ represents the $j$-th dimension, $a$ is a constant in the domain, and ' $<'$ can be other comparison operators. It is transformed to an encrypted half-space query in the perturbed space: $p_i^T Q p_i < 0$, where $p_i$ is defined earlier as the perturbed vector. $Q$ is $(A^{-1})^T uv^T A^{-1}$, where $u$ is a vector with all entries zero except for $j$-th dimension set to 1 and $d+1$-th dimension set to $-a$ corresponding to the vector representation of the condition $X_j < a$, i.e., $p_i^T (A^{-1})^T u < 0$; $v$ is a vector with all entries zero except for $d+2$-th dimension set to 1. This quadratic query form $p_i^T Q p_i < 0$ represents the equivalent query condition $(X_j - a)V < 0$, where $V$ is the $d{+}2$-th expanded random positive dimension. For details, we refer readers to the original paper [1].

Note that the above query encoding can be extended to encode half-space queries in general form $w^T x + b < 0$. We only need to revise the $u$ vector to be $u = (w^T, -b, 0)^T$. This general form of half-space query will be used in our PerturBoost framework, as shown in Figure 1.

# 3. PERTURBOOST: PROTECTING BOTH DATA AND MODEL PRIVACY

In the PerturBoost framework, the client prepares a perturbed dataset and the parameters, and then outsources them to the cloud. The PerturBoost algorithm is invoked in the cloud to get a model for the client.

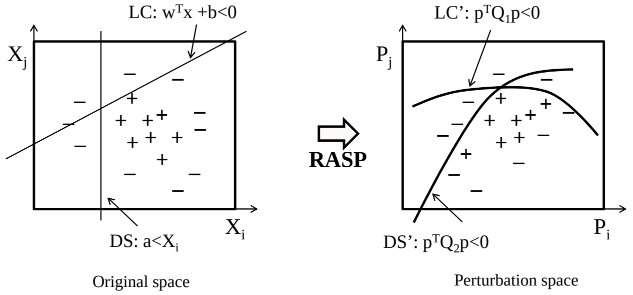

**Figure 1: Illustration of learning classifiers.**

The PerturBoost algorithm is basically an algorithm wrapping the AdaBoost algorithm [2] for processing the perturbed data. We describe the details later.

After the model is learned, it can be applied in two different settings: either transforming the model back to the unperturbed data space - the *model transformation* approach, or transforming the new feature vector data, $\{x_{new}\}$, whose labels are to be predicted, to the perturbed space - the *data transformation* approach. If the user wants to apply the model remotely in the cloud, then the data transformation approach should be used. While using the model locally in the client side, the user can choose any of the two.

## 3.1 PerturBoost Learning

The PerturBoost framework uses the AdaBoost algorithm to handle the base classifiers that adapt to the perturbed data. Algorithm 1 shows the basic structure of the PerturBoost learning algorithm.

**Algorithm 1** PerturBoost($B, Tr, Ts$)

```
Input: $B$: the type of base classifier; $Tr$: the perturbed training
    dataset; $Ts$: the perturbed testing dataset.

model ← AdaBoost(B, $Tr$, $Ts$);
return model;
```

We describe two types of RASP base classifiers.

## 3.2 RASP Base Classifiers

The RASP perturbation only preserves one type of utility: half-space queries. Thus, the applicable models are limited to linear classifiers. In the preliminary study, we test two types of *random* linear classifiers: random decision stump and random general linear classifier. Randomized classifiers increase the resilience to the attacks on model privacy. These classifiers, if applied as individual standalone classifiers, are useless because of their low accuracy. However, they are good enough to serve as weak base classifiers in the boosting framework.

**RASP random decision stump.** Random decision stumps are a straightforward translation of the simple range conditions like $X_j < a$ described in the RASP paper [1]. Note that with the decision stump form, the query parameter matrix: $\Theta = (A^{-1})^T uv^T A^{-1}$ can be simplified. Let $\alpha_j$ be the $j$-th row of $A^{-1}$. $\Theta$ is actually $(\alpha_j - a\alpha_{d+1})^T \alpha_{d+2}$, which weakens the model privacy.

[2] The full version transforms the dimensional values with order preserving encryption (OPE), before applying the described steps [1].

**RASP random linear classifier.** In the RASP query representation, we try to generate random linear classifiers in the following way. The $v$ vector keeps unchanged, while the $u$ vector is set to $(w^T, b, 0)^T$ for the original query $w^T x + b < 0$. It is easy to check that this transformation is correct. Thus, the problem is transformed to finding an appropriate setting of $w$ and $b$.

Arbitrarily generated random linear classifiers might not be very useful. It may result in a very skewed partitioning of the dataset. Instead, we use the following method to increase the chance of finding reasonable random linear classifiers. First, we normalize each dimension with the transformation $(X_j - \mu_j)/\sigma_j$, where $\mu_j$ is the mean and $\sigma_j$ is the standard deviation of the dimension $X_j$. In this way, we can reduce the differences caused by very different domains (e.g., one in the range [0,1] and the other in the range [100,200]). Then, we choose each dimension of $w$ and the constant $b$ uniformly at random from the range $[-1, 1]$. In this way of choosing $w$, the perpendicular direction of resultant hyperplane will be uniformly distributed in the unit hyper-sphere. In addition, the setting of $b$ will constrain the dimensional intercepts in the range $[-1, 1]$, forcing the plane to cut the dataset around the center of the data distribution. This minimizes the chance of generating skewed linear classifiers.

## 3.3 Discussion on Model Privacy

A potential attack can be conducted to breach the privacy of the query (i.e., the decision stump model) if a strong assumption is held that the attacker knows two pairs of input-output queries on the same dimension. We assume that the attacker knows $X_j < a_1$ and its encoded form $\Theta_1$, and $X_j < a_2$ and $\Theta_2$, respectively. Then, the $\Theta$ matrix for any value in the $X_j$ domain can be possibly enumerated. For instance, for $a_3 = (a_1 + a_2)/2$, we have the corresponding $\Theta_3 = (\Theta_1 + \Theta_2)/2$. As any value in the domain can be represented as $a_1 + \lambda(a_2 - a_1)$, $\lambda \in \mathbb{R}$, the corresponding $\Theta$ is $\Theta_1 + \lambda(\Theta_2 - \Theta_1)$. This means the model privacy is not preserved, if the attacker is equipped with such additional knowledge. We call it *the model-enumeration attack*.

Theoretically, using random linear classifiers does not avoid this attack. After all, if the attacker knows a pair of hyperplanes with parameters $u_1$ and $u_2$, and their $\Theta$s, respectively, he/she can still use the same enumeration method to derive other hyperplanes and their $\Theta$ representations. However, different from decision stumps, which have the values constrained in one dimension, this attack only covers a small vector space, i.e., the points on the line $u_1 + \lambda(u_2 - u_1)$. As a random selection of $u$ has extremely low probability falling on the line, the chance of breaching a randomly generated linear model with this amount of knowledge is negligible.

Because the random linear classifier approach makes the model-enumeration attack computationally more expensive, we believe random linear classifiers provide more model-privacy protection than decision stump classifiers. A more rigid study will be conducted for this comparison.

# 4. PRELIMINARY EXPERIMENTS

We want to understand whether the PerturBoost framework can generate classifiers with satisfactory accuracy.

**Datasets.** For easier validation and reproducibility of our results, we use a set of public data from UCI machine learning repository in experiments. For convenience we also select the datasets of only two classes. These datasets were widely applied in various classification modeling and evaluation.

In pre-processing, the missing values in some datasets (e.g., the Breast-Cancer and Ionosphere datasets) are replaced with random samples from the domain of the corresponding dimension. They are then normalized with the transformation $(v - \mu_j)/\sigma_j$, where $\mu_j$ is the mean and $\sigma_j^2$ is the variance of the dimension $j$, to remove the bias introduced by the domains. Then, the datasets are randomly shuffled and split into training data (70% of the records) and testing data (30%). Each of the datasets is also perturbed with the RASP method.

| Dataset | NoPert | | DS | LC |
|---|---|---|---|---|
| Breast-Cancer | 3.7 | | 2.3 | 2.8 |
| Credit-Australian | 13.4 | | 22.5 | 11.5 |
| Credit-German | 22.7 | | 29.3 | 22.7 |
| Diabetes | 21.6 | | 22.1 | 22.0 |
| Heart | 13.5 | | 11.2 | 12.5 |
| Hepatitis | 12.8 | | 21.2 | 14.7 |
| Ionosphere | 2.8 | | 12.1 | 10.4 |
| Spambase | 6.7 | | 17.0 | 11.1 |

**Table 1: Error-rate comparison for different models (%).**

**Implementation.** We implement the RASP perturbation based on the algorithm in the paper [1]. The Weka package [3] is used to implement the PerturBoost framework. The two base classifiers, RASP random decision stump and RASP random linear classifier, are implemented based on Weka's Java interface. The Weka package also uses the LibSVM library for SVM classifiers.

**Preliminary Results** In the following table (Table 1), "NoPert" means the best SVM classifiers on the original non-perturbed data. We test SVM classifiers with the three popular kernels: linear, radial basis function, and sigmoid function, and choose the best results. "DS" represents decision stump base classifiers are used for PerturBoost, and "LC" means general linear base classifiers.

Classifiers are trained with the training data and tested on the testing data. Table 1 shows the testing error-rates for the models. Overall, general linear base classifiers give better results than decision stump base classifiers, and the results are also close to the non-perturbed scenarios in most cases.

# 5. CONCLUSION

This poster presents a preliminary study on the PerturBoost approach that aims to provide efficient secure classifier learning in the cloud with both data and model privacy preserved, using previously studied RASP perturbation approach. The results show that PerturBoost with certain secure base classifiers can generate good models with accuracy and security guarantee.

# 6. REFERENCES

[1] Chen, K., Kavuluru, R., and Guo, S. Rasp: Efficient multidimensional range query on attack-resilient encrypted databases. In *ACM Conference on Data and Application Security and Privacy* (2011), pp. 249–260.

[2] Freund, Y., and Schapire, R. E. A short introduction to boosting. In *International Joint Conferences on Artificial Intelligence* (1999), Morgan Kaufmann, pp. 1401–1406.

[3] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The weka data mining software: An update. *SIGKDD Explorations 11*, 1 (2009).

[4] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer-Verlag, 2001.

[5] Narayanan, A., and Shmatikov, V. Robust de-anonymization of large sparse datasets. In *Proceedings of the IEEE Symposium on Security and Privacy* (2008), pp. 111–125.