Anomaly Detection with Partially Observed Anomalies

Ya-Lin Zhang[†,‡], Longfei Li[‡], Jun Zhou[‡], Xiaolong Li[‡], Zhi-Hua Zhou[†]
[†]National Key Lab for Novel Software Technology, Nanjing University, China
[†]{zhangyl, zhouzh}@lamda.nju.edu.cn
[‡]Ant Financial Services Group, China
[‡]{longyao.llf, jun.zhoujun, xl.li}@antfin.com

ABSTRACT

In this paper, we consider the problem of anomaly detection. Previous studies mostly deal with this task in either supervised or unsupervised manner according to whether label information is available. However, there always exists settings which are different from the two standard manners. In this paper, we address the scenario when anomalies are partially observed, i.e., we are given a large amount of unlabeled instances as well as a handful labeled anomalies. We refer to this problem as anomaly detection with partially observed anomalies, and proposed a two-stage method **ADOA** to solve it. Firstly, by addressing the difference between the anomalies, the observed anomalies are clustered, while the unlabeled instances are filtered to get potential anomalies and reliable normal instances. Then, with the above instances, a weight is attached to each instance according to the confidence of its label, and a weighted multi-class model is built, which will be further used to distinguish different anomalies to the normal instances. Experimental results show that in the aforementioned setting, existing methods behave unsatisfactorily and the proposed method performs significantly better than all these methods, which validates the effectiveness of the proposed approach.

KEYWORDS

Anomaly Detection, Observed Anomalies, Two-Stage Method

1 INTRODUCTION

Anomaly detection [5] is a broadly used technique which aims at identifying the unexpected patterns from the usual behavior in a dataset. These unexpected patterns are always called anomalies or outliers, which are always generated by some kind of malicious purpose or illegal activity. Anomaly detection is important and can provide significant and critical help in various applications, such as intrusion detection [10], fraud detection [16], fault detection [14], suspicious transaction detection [23] and abnormal moving activity detection [11], etc.

To deal with this task, machine learning based techniques have been widely employed during the past few years, and these techniques can be roughly classified into two categories: unsupervised learning based approaches [18] and supervised learning based methods [12]. Traditionally, unsupervised learning based methods are developed, in which only unlabeled data are accessible. Distance based approaches [26], density based approaches [3] and isolation based methods [23] are typical representatives along this way.

On the other hand, if sufficient labeled data are available, supervised learning based methods are explored, in which a classification model, such as support vector machine [33], decision tree [35] and k-nearest neighbor [31], etc., can be trained to further classify unseen samples. Note that compared to unsupervised approaches, supervised methods can always provide better performance with the help of sufficient labeled data. In addition, by using both labeled and unlabeled data, semi-supervised learning based methods [30] are explored, and by combining different techniques, hybrid approaches [27] have also been developed to handle this problem.

However, there are some conditions in which adequate labeled samples are difficult to obtain, while we can access a small number of recognized anomalies, along with sufficient unlabeled samples. Let's take the task of malicious URL detection as an example, in some scenarios, apart from a large amount of unlabeled URL records, we can only obtain a handful of labeled malicious URLs, with the help of existing rule based systems. Different to supervised setting in which both positive and negative samples are provided, we only get a small set of positive (malicious) samples here, thus the supervised methods can not be directly employed. On the other hand, when compared to unsupervised learning setting, we additionally have some labeled samples, which may offer great help with proper utilization. In this paper, we refer to the this special anomaly detection setting as anomaly detection with partially observed anomalies).

There is one paradigm named PU (Positive and Unlabeled) learning [17, 19], which has seemingly similar setting with the aforementioned one. However, in PU learning, the positive samples always belong to one concept center, which means that the positive samples are similar to each other, whereas in anomaly detection, the so-called positive samples (anomalies) are usually not similar to each other, and they can be seriously disparate. In another word, we can not claim that the difference between two outliers are smaller than that of an anomaly and a non-anomaly. Thus, direct applying of PU learning based techniques for anomaly detection task may not lead to satisfactory performance.

Another paradigm called semi-supervised clustering [1, 34] deals with the cluster setting where the data are partially labeled or with other types of preliminary information, and the objective is to cluster the unlabeled samples to the appropriate clusters. It seems that semi-supervised learning deals with the similar task as we described.

WWW '18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3186580

However, just as PU learning, the samples labeled in the same cluster should be similar to each other in semi-supervised clustering, while in anomaly detection, the observed anomalies do not conform to this.

In this paper, we consider the setting of anomaly detection with partially observed anomalies, and propose a method called **ADOA** (**A**nomaly **D**etection with partial **O**bserved **A**nomalies) to solve it. **ADOA** follows a two-stage manner. In the first stage, we address that the observed anomalies should not be simply regarded into one concept center, and by assuming that the anomalies belong to k different concept centers, the anomalies are firstly clustered into k clusters. After that, both potential anomalies and reliable normal samples are selected from the unlabeled samples according to the isolation degree and the similarity to the nearest anomaly cluster center. In stage two, a weight is set to each sample according to the confidence of its attached label, and a weighted multi-class classification model is built to distinguish different anomalies from the normal samples, using original anomalies and the selected samples. Experiments on different datasets and a real application task demonstrate the effectiveness of our approach.

The rest of this paper is organized as follows. In section 2, we review the related work.In section 3, we state the problem setting and present the proposed method. In section 4, we report the experimental results on different datasets. In section 5, we apply the proposed method to the problem of malicious URL detection and validate the effectiveness of the proposed method. Finally, we conclude the paper in section 6.

2 RELATED WORK

Anomaly detection [5] deals with the task of recognizing unexpected patterns from normal behavior. The detection of anomalies has significant influence and can provide critical help in many different fields. During its development, many machine learning based methods have been proposed to handle this problem [28], and it has been widely applied in many applications, such as intrusion detection [10], fraud detection [16], fault detection [14], suspicious transaction detection [23] and abnormal moving activity detection [11], etc.

Among the developed methods, unsupervised learning based methods [4, 37] build the model with unlabeled data. To name some representative, distance based approaches [15], density based approaches [3], isolation based methods [22, 23], and so on. These methods can be widely used since there is no need for labeling of the data. However, in many application fields, the unsupervised methods may not succeed to achieve the require performance.

On the other hand, with labeled data provided, supervised learning based methods are explored. Many supervised algorithms, such as support vector machine [33], decision tree [35] and k-nearest neighbor [31] are successively adopted to the task of anomaly detection. With proper use of the label information, supervised learning based methods can always achieve better performance. Beyond these two standard paradigms, other methods, including semi-supervised learning based methods [30] and hybrid approaches [27] have also been explored based on these techniques to handle this task.

In some conditions, only the samples following the normal behavior are provided [32], while the anomalies are unseen. Methods like one-class learning [7] and support vector data description [21] are developed for this setting. These methods focused on learning the hypersphere to describe the normal samples or learning a hyperplane to divide the data points from the origin with maximum-margin.

PU (Positive and Unlabeled) learning [17] is a special case of semi-supervised learning [6, 36], which copes with the setting when only positive and unlabeled data are available, while no negative sample is labeled. During the past few years, a mass of methods have been proposed to deal with this task. Roughly speaking, these methods can be divided into three families. Two-step approaches [19, 20] try to recognize some reliable negative samples from the unlabeled data, then a traditional supervised learning or semi-supervised learning technique can be applied. Cost-sensitive learning techniques [24] for binary classification with unequal misclassification cost are also readily available for handling this problem [8]. What's more, convex methods have also been proposed to deal with this task [9]. Note that if we regard anomalies as positive samples here, PU learning is somewhat similar to anomaly detection with partially observed anomalies. However, the most striking difference is that, the positive samples in PU learning are similar to each other, thus we can find one positive concept for them, while in anomaly detection, the anomalies are always diversified, and they can rarely cluster into one concept cluster, making the standard PU learning technique not suitable to handle anomaly detection task.

Semi-supervised clustering [1] deals with the problem when the provided data are partially labeled or with other types of preliminary information, and the goal is to try to assign the unlabeled samples to the proper clusters. Many methods [2, 34] for this task are generalized from the traditional clustering algorithms, with modification to make sure that the constraints are satisfied. However, just as PU learning, the samples labeled in the same cluster should be similar to each other in semi-supervised clustering, while in anomaly detection, the observed anomalies do not conform to this.

In this paper, we focus on a special setting of anomaly detection, i.e., anomaly detection with partially observed anomalies. Different to totally unsupervised anomaly detection scenario, we have some preliminary information, i.e., the observed anomalies. Different to supervised setting, we only have a small amount of anomalies, while the other samples are totally unlabeled. Different to PU learning and semi-supervised clustering, the labeled anomalies are usually not similar to each other.

3 ANOMALY DETECTION WITH PARTIALLY OBSERVED ANOMALIES

In this section, we will first state our problem setting, and then present the proposed method **ADOA** (**A**nomaly **D**etection with partial **O**bserved **A**nomalies) .

3.1 Problem Statement and Notations

Let $\mathcal{X} = \mathbb{R}^d$ denotes the instance space and $\mathcal{Y} = \{-1, +1\}$ denotes label space, respectively. Let $y = +1$ indicates the anomalies and $y = -1$ for normal samples. We are given a data set with m training samples $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l), \boldsymbol{x}_{l+1}, \ldots, \boldsymbol{x}_m\}$, where $\boldsymbol{x}_i \in \mathcal{X}$ representing a sample. The first l samples are labeled as anomalies, which are denoted by $\mathcal{D}^l = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l)\}$,

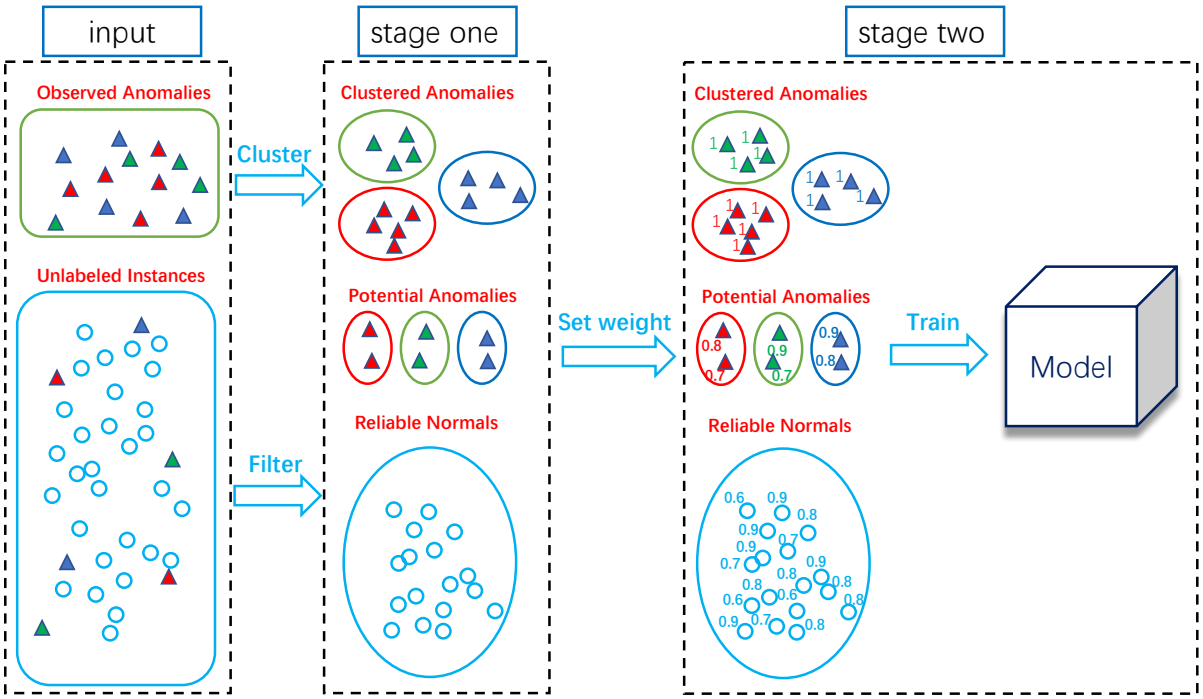

Figure 1: The overall framework of the proposed method.

and the other $m - l$ samples are unlabeled, which are denoted by $\mathcal{D}^u = \{\boldsymbol{x}_{l+1}, \ldots, \boldsymbol{x}_m\}$. Note that although the first l samples are all with the label $y = +1$, the can be totally different to each other. The goal is to build a model $f : \mathcal{X} \to \mathcal{Y}$, so that it can be further used to distinguish the diversified anomalies from the normal ones for future-coming data, thus the anomalies can be recognized.

3.2 Proposed Method

ADOA follows the two-stage manner. In the first stage, both observed anomalies and unlabeled samples are manipulated. We address that the observed anomalies are different to each other, and they should not be simply classified into one concept center. Since the anomalies are really diversified, we first try to separate them into different clusters, so that the samples in each cluster are similar to each other. For unlabeled data, we aim at sufficiently exploring the information of them. Thus, we try to filter both potential anomalies and reliable normal samples from them, with consideration of the isolation score (to be explained shortly) and their similarity score to the observed anomalies. The intuition is that, on one hand, the potential anomalies should be different to normal samples (i.e., can be easily isolated); on the other hand, they should be similar to some observed anomalies. In the second stage, we build a weighted multi-class model to distinguish different anomalies from the normal samples. For the observed anomalies, the weights are set to 1, and for the filtered samples, the weights are set according to the confidence of their attached labels. The overall procedure is shown in Figure 1, and the details of **ADOA** are presented below.

In stage one, by addressing the difference of the anomalies, we first cluster observed anomalies in $\mathcal{D}^l$ into k clusters $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$. We can employ different cluster algorithms. Here we simply run k-means algorithm (with normalization performs in advance). Specifically, the distance between each two samples is measured using squared Euclidean distance, which is as below

$$dist(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \sum_{j=1}^{d} (\boldsymbol{x}_{ij} - \boldsymbol{x}_{i'j})^2 , \tag{1}$$

in which d is the dimension of the samples. The following square error is minimized to learn the centers and the assignments of the samples,

$$E = \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in C_i} dist(\boldsymbol{x} - \boldsymbol{\mu}_i) , \tag{2}$$

in which $\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$ is the center of the i-th anomaly cluster, i.e., the anomaly concept center. Note that other cluster algorithms, such as hierarchical clustering and density-based clustering, can also be explored, and this may lead to further improvement of the performance.

For samples in unlabeled set $\mathcal{D}^u$, we select the potential anomalies and reliable normal samples based on the isolation score and the similarity score to the nearest cluster center.

Isolation Score: The concept of isolation was first proposed in [22]. They showed that an extremely random tree forest can be used for isolating samples. Each tree in the forest is built by

randomly choosing an attribute and a corresponding split value for subsequent growing at each node. Since the anomalies are few and different, they are always isolated closer to the root of the tree, whereas normal samples will go to the deeper leaf of the tree. To get the isolation score, each sample is delivered to a tree until it arrives at a leaf, the path length in each tree is then obtained, and the average path length can be calculated for the isolation forest. Based on the average path lengths on the trees, the isolation score $IS(\boldsymbol{x})$ can be calculated to describe the probability of a sample $\boldsymbol{x}$ being anomaly. Let $h(\boldsymbol{x})$ denotes the path length of a sample $\boldsymbol{x}$ on a tree, and $E(h(\boldsymbol{x}))$ indicates the average path length of a collection of isolation trees. Assume that there are n samples, and let

$$c(n) = 2H(n) - (2(n-1)/n) \tag{3}$$

denotes the average path length of unsuccessful search in Binary Search Tree, which is the same as the estimation of average $h(\boldsymbol{x})$ for external node terminations. Here $H(n)$ is the harmonic number, which can be estimated by $ln(n) + 0.5772156649$ (Euler's constant). $c(n)$ is used as the normalization parameter to calculate the isolation score $IS(\boldsymbol{x})$, which is as following:

$$IS(\boldsymbol{x}) = 2^{-\frac{E(h(\boldsymbol{x}))}{c(n)}} . \tag{4}$$

The higher is the score $IS(\boldsymbol{x})$ (close to 1), the more likely that $\boldsymbol{x}$ being an anomaly.

Similarity Score: On the other hand, it is reasonable that the closer is a sample to a known anomaly concept center, the more likely that the sample being a potential anomaly, thus we calculate the similarity score $SS(\boldsymbol{x})$ between a sample $\boldsymbol{x}$ and its nearest anomaly concept center. Specifically, $SS(\boldsymbol{x})$ is calculate as following:

$$SS(\boldsymbol{x}) = \max_{i=1}^{k} e^{-(\boldsymbol{x}-\boldsymbol{\mu}_i)^2} , \tag{5}$$

in which $\boldsymbol{\mu}_i$ denotes the i-th concept center, and k is the number of anomaly concept centers.

Total Score:To filter the potential anomalies and reliable normal samples from the unlabeled samples, we take both the isolation score and the similarity score into consideration, and the total score for an instance is denoted as

$$TS(\boldsymbol{x}) = \theta IS(\boldsymbol{x}) + (1-\theta)SS(\boldsymbol{x}) , \tag{6}$$

in which $\theta \in [0, 1]$ is a parameter to balance the importance of isolation score and similarity score.

Let

$$\alpha = \frac{1}{l}\sum_{i=1}^{l} TS(\boldsymbol{x}_i) \tag{7}$$

indicates the average score of observed anomalies. We then select the instances with

$$TS(\boldsymbol{x}) \geq \alpha \tag{8}$$

as potential anomalies, and put them into their nearest anomaly clusters. We select the instances with

$$TS(\boldsymbol{x}) \leq \beta \tag{9}$$

as reliable normal samples, where β is a predefined parameter. The smaller is β, the more reliable are the selected samples.

So far, we have not only the observed anomalies, but also the selected potential anomalies and the reliable normal samples, and the anomalies are separated into k different clusters, so that in each cluster, the anomalies are with high similarity to each other within the cluster.

In stage two,we first set weights for all selected samples and the observed anomalies. Specifically, all observed anomalies are with weight 1, and for the selected anomalies, as shown in Eq. 10, the higher is the score $TS(\boldsymbol{x})$, the more weight will the instance get.

$$w(\boldsymbol{x}) = \frac{TS(\boldsymbol{x})}{\max_x TS(\boldsymbol{x})} . \tag{10}$$

For selected reliable normal samples, the smaller is the score, the more weight it will get. The details are as below:

$$w(\boldsymbol{x}) = \frac{\max_x TS(\boldsymbol{x}) - TS(\boldsymbol{x})}{\max_x TS(\boldsymbol{x}) - \min_x TS(\boldsymbol{x})} . \tag{11}$$

With the above mentioned samples and their weights, a weighted $(k+1)$-class model can be trained to separate different anomalies to the normal samples. Particularly, each anomaly cluster is regarded as one class, so there are $k+1$ classes in all (k anomaly classes and one normal class). The following objective is minimized,

$$\sum_i w_i l(y_i, f(\boldsymbol{x}_i)) + \lambda R(\boldsymbol{w}) , \tag{12}$$

in which w_i denotes the weight of the instance $\boldsymbol{x}_i$, $l(y_i, f(\boldsymbol{x}_i))$ is the loss term, and $R(\boldsymbol{w})$ the the regularization term. In this work, support vector machine is used, so the loss term and regularization term are set to be hinge-loss and $L2$-norm.

After obtaining the multi-class model, the new-coming samples can be classified. When applying this model to an unseen instance, no matter which of the k anomaly clusters is the new-coming instance classified to, it will be regarded as an anomaly.

4 EXPERIMENTS

In this section, we run experiments on both synthetic data and real-world data to validate the performance of the proposed method. We compare the proposed method with different baselines including unsupervised approach, supervised approach and PU learning approach. Firstly, unsupervised method Isolation Forest [22] is considered, since it has been proved as a powerful method for anomaly detection. Secondly, supervised method support vector machine [13] is considered. By simply regarding all unlabeled samples as negative ones, supervised method is tested. Thirdly, PU learning based method, i.e., the cost sensitive strategy [25], is compared too, since the setting is somewhat similar to the setting of PU learning, and we want to validate whether PU learning is suitable for the problem of anomaly detection.

4.1 Experiments on Synthetic Data

We first perform experiments on synthetic dataset, to test the performance of each method in the scenario that the anomalies are diversified. To generate the dataset for anomaly detection, we first generate the normal examples, which are sampled from the multivariate Gaussian distribution $P_0 = \mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0)$, in which $\boldsymbol{\mu}_0 = [5, 5]$ is the mean vector, and $\Sigma_0 = [[5, 0], [0, 5]]$ is the covariance matrix. For anomalies, we assume that there are three different concept clusters, and they are sampled from multivariate Gaussian distribution $P_1 = \mathcal{N}(\boldsymbol{\mu}_1, \Sigma)$, $P_2 = \mathcal{N}(\boldsymbol{\mu}_2, \Sigma)$ and $P_3 = \mathcal{N}(\boldsymbol{\mu}_3, \Sigma)$, in which $\boldsymbol{\mu}_1 = [1, 1]$, $\boldsymbol{\mu}_2 = [1, 10]$, $\boldsymbol{\mu}_3 = [9, 0]$ and $\Sigma = [[0.6, 0], [0, 0.5]]$. The

Table 1: Details of the datasets information and experiments setups. 'Dimension' denotes the dimension of the datasets. 'Observed', 'Unlabeled' and 'Test' denotes the number of observed anomalies, training unlabeled samples and test samples, respectively. 'Class prior' denotes the proportion of anomalies in unlabeled and test data.

	Dimension	Observed	Unlabeled	Test	Class prior
synthetic	2	20	10000	10000	0.01
arrhythmia	274	10	200	200	0.1
vowel	12	10	500	500	0.02
letter	32	10	500	500	0.08
musk	166	10	1000	1000	0.04
thyroid	6	10	1000	1000	0.03
speech	400	10	1500	1500	0.013
satimage	36	10	2500	2500	0.01
smtp	3	10	40000	40000	0.00025
ionosphere	33	20	150	150	0.33
breastw	9	20	300	300	0.33
optdigits	64	20	2000	2000	0.025
pendigits	16	20	3000	3000	0.017
pima	8	50	300	300	0.33
cardio	21	50	500	500	0.1
mammography	6	50	5000	5000	0.02
satellite	36	100	2000	2000	0.25
annthyroid	6	100	2000	2000	0.1
shuttle	9	100	10000	10000	0.1
ForestCover	10	100	100000	100000	0.01
http	3	100	200000	200000	0.005

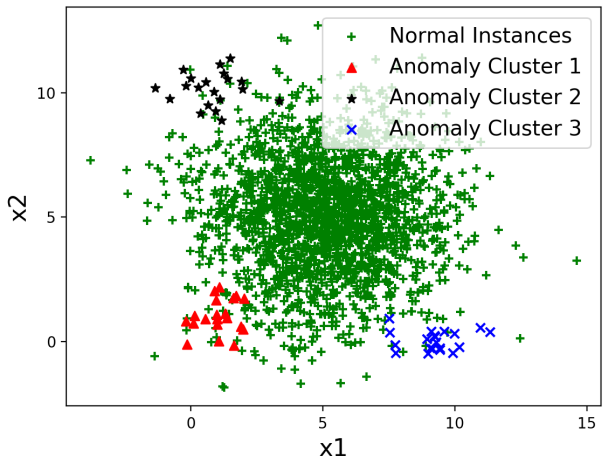

Figure 2: The sampled examples of synthetic dataset.

sampled examples are shown in Fig. 2. As we can see, the anomalies from different clusters are really different from each other, and the anomalies from the same cluster are pretty similar to each other.

To generate the setting of anomaly detection with partially observed anomalies, we randomly sample 20 examples (which may come from any of the three different clusters) from the anomalies as observed anomalies. We then sample examples to construct unlabeled training set and test set, and the number of unlabeled and test examples are both set to 10000, among which only 1% of them are anomalies. The details are shown in the first line of Table 1, with the name 'synthetic'.

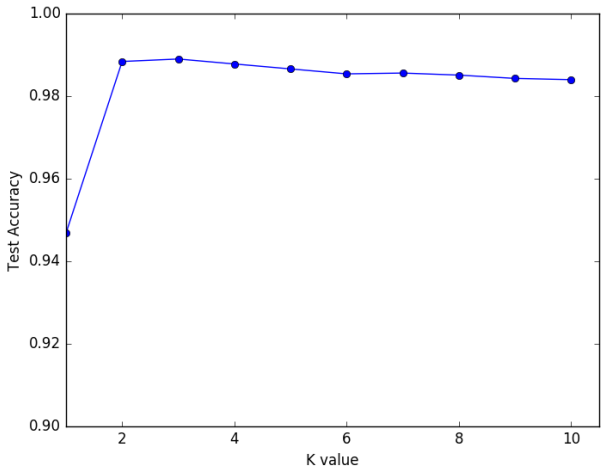

Figure 3: The accuracy with different k value.

We repeat experiments for 30 times by using the dataset generation procedure to generate observed anomalies, unlabeled training data and test sets. The AUC score is shown in the first line of Table 2. As we can see, the proposed method performs better than

Table 2: AUC score on different datasets. The highest AUC score is marked in bold.

	Proposed Method	Unsupervised Method	Supervised Method	PU Method
synthetic	**0.989**	0.954	0.944	0.978
arrhythmia	**0.840**	0.515	0.665	0.564
vowels	**0.984**	0.774	0.972	0.979
letter	**0.671**	0.625	0.633	0.535
musk	**1.000**	0.995	0.880	**1.000**
thyroid	**0.994**	0.971	0.718	0.989
speech	**0.730**	0.524	0.613	0.690
satimage	**0.992**	0.990	0.947	0.975
smtp	**0.902**	0.833	0.788	0.876
ionosphere	**0.934**	0.846	0.705	0.899
breastw	**0.993**	0.988	0.824	0.992
optdigits	**0.999**	0.811	0.972	**0.999**
pendigits	**0.996**	0.955	0.978	0.995
pima	**0.791**	0.691	0.660	0.775
cardio	**0.991**	0.892	0.970	0.987
mammography	**0.938**	0.845	0.605	0.909
satellite	**0.855**	0.694	0.735	0.838
annthyroid	**0.922**	0.775	0.642	0.850
shuttle	0.989	**0.994**	0.703	0.984
ForestCover	**0.999**	0.925	0.842	**0.999**
http	0.997	**0.999**	0.994	0.995

all other existing methods, which demonstrate the effectiveness of the proposed method.

Furthermore, we vary the value of parameter k to examine the influence of it. The results in Fig. 3 show that the behavior tends to get better as the value of k getting closed to the ground-truth. Besides, when k gets to be a little bigger, our method still works fine, which means that our method is not that sensitive with bigger k. However, when we set the value of k to 1(which means that we assume the anomaly are similar), the performance is showed to be pretty unsatisfactory, which validate the necessity of addressing the difference for the anomalies.

4.2 Experiments on Real-world Data

To explore the result on real-world data, the experiments are performed on lots of different benchmark datasets which come from different fields [29]. Note that in our setting, we are given a handful of observed anomalies, with a large amount of unlabeled samples. To construct this setting, we simply sample a small amount of anomalies as observed ones, as well as plenty of unlabeled samples. The dimension of the data, number of observed anomalies, unlabeled train samples and test samples are shown in Table 1, with 'class prior' indicating the proportion of anomalies in unlabeled and test data. As we can see, the datasets are very diversified, with different dimension, different numbers of samples and different class prior. What's more, we need to address that the number of observed anomalies are pretty small, i.e., as few as 10 for some datasets and at most 100.

For each datasets, normalization is performed, and we repeat experiments for 30 times by using the dataset generation procedure to generate observed anomalies, unlabeled training data and test sets.

The AUC scores are shown in Table 2. As we can see from the table, the proposed method performs significantly better than all other methods (wins 18 times among all 20 datasets), validating the effectiveness of the proposed method. Unsupervised method Isolation Forest reaches the first place on 'shuttle' and 'http' dataset, while on some datasets (e.g., 'arrhuthmia' and 'speech' dataset), the behavior may be pretty unsatisfactory. One interesting result is that the supervised method performs pretty awful, indicating that we should not simply regard all unlabeled samples as negative. One possible explanation is that, if we simple regard all unlabeled samples as negative, the noises will seriously deteriorates the performance.

On some datasets, the PU learning based method can also perform well, while on some dataset such as the 'arrhythmia' dataset, the performance is pretty terrible. This maybe because that, for some datasets, the anomalies are not very diversified, i.e., we can nearly find a concept center for the anomalies, making PU learning strategy feasible. However, when the anomalies get to be diversified, the PU learning based method will fail to reach the goal. Furthermore, as we can see, the proposed method never performs

Table 3: Accuracy on different datasets. The highest accuracy score is marked in bold.

	Proposed Method	Unsupervised Method	Supervised Method	PU Method
Date1	**0.878**	0.609	0.849	0.847
Date2	**0.883**	0.620	0.844	0.849
Date3	**0.891**	0.611	0.854	0.860
Date4	**0.881**	0.613	0.849	0.847
Date5	**0.876**	0.588	0.847	0.848

worse than PU learning based method. This is reasonable, because if we set the parameter k of **ADOA** to 1, it is degenerated to a special case of PU learning based strategy.

5 APPLICATION TO MALICIOUS URL DETECTION

In this section, we apply the proposed method to the problem of malicious URL detection and validate the performance of different methods on this problem.

With the fast development of Internet, more and more kinds of URL attacks have arisen, which becomes a serious threat to cyber-security. During the past years, many methods have been developed for this problem. For example, traditional techniques which based on blacklists or rule lists are first explored. However, these methods lack the ability of detecting potential attacks, making it awkward for cyber-security engineers to efficiently discover newly generated URL attacks.

Machine learning based methods are then explored to provide better generalization performance for this problem. However, as we discussed before, they are mainly focused on supervised and unsupervised setting, and when we are given a small set of recognized malicious URLs (which will be regarded as anomalies) and a large amount of unlabeled URLs, traditional approaches will fail to apply. In this section, we apply our proposed method to it.

scheme:[//[user[:password]@]host[:port]][/path][?query][#fragment]

Figure 4: The generic syntax of URLs

We first extract numerical feature from the original URLs. As shown in Fig. 4, the URLs can always be separated into different parts, including the scheme part, the authority part (user, password), the path part (host, path), the query part and fragment parts, etc. In our scenario, the first few parts are restricted, and the attacks mainly come from the malicious modification of the fragment parts. Thus, we extract feature for each URL based on the fragment parts. The fragments are always formed as '$key_1 = value_1 \& \cdots \& key_n = value_n$', and the value may be arbitrarily modified by the attackers to make an attack.

Even more specifically, given a set of URLs, we firstly divide each of them into the aforementioned parts, and then we extract the key-value pairs from the fragments of each URL. Secondly, since we are focused on discovering the trait of *malicious* URLs, we try to filter the key-value pairs and only keep the top-N keys that appear mostly in the *malicious* URLs, while the rest of the key-value pairs for each URL are collected together as one key-value pair, thus there will be at most $(N + 1)$ key-value pairs extracted from each URL. In this way, the feature vector will not get to be that tedious. Finally, based on domain knowledge, we heuristically extract eight different statistical information from each of the filtered values, including the count of *all* characters, letters, numbers, punctuations in the value, and the count of *different* characters, letters, numbers, punctuations in the value. Thus each URL will be described by a $(N + 1) * 8$ dimensional feature vector.

When running the experiments, the used data is sampled from the daily-arrived URL requests. The data mainly contains two parts: a large set of unlabeled URLs and a handful of malicious URLs which have been already marked by the existing system, and different attack types may appear among the malicious ones, including XXE (XML External Entity Injection), XSS (Cross SiteScript) and SQL injection, etc. Note that the only label information is whether a URL is recognized as a malicious one, the exact type of the attack is unknown. Since the total dataset is too large, we sample more than 10 millions of URLs from one month's requests, in which the number of observed malicious URLs by the existing system is less than 10 thousand. The model is trained using the sampled data, and will be used to predict the scores of each day's new-coming *unlabeled* URLs. When extracting key-value pairs, N is set to be 99, so that each URL is described by a 800 dimensional vector. Min-max normalization is used to process the features to same scale. What's more, all SVM classifiers are replaced by logistic regression in the experiment, since the dataset is too large.

Since we have no supervision information for the daily-arrived new URLs, we use the help of the cyber-security engineers to manually review the results and verify the effectiveness of the proposed method. It is very time-consuming to check the results, so we select the top-1000-scored potential malicious URLs from each day's data, and cyber-security engineers will manually check whether the selected URLs are malicious or benign with their domain knowledge.

Table 3 shows the accuracy of the selected potential malicious URLs on 5 different dates. As we can see, the proposed method perform significantly better than all other methods, which demonstrates the effectiveness of the proposed method on the malicious URL detection task.

6 CONCLUSION

In this paper, we address the problem of anomaly detection. Different to traditional strategies, which formalize this problem as a supervised (with labeled data provided) or unsupervised learning problem (without any labeled samples), we consider the setting when we are given a small amount of observed anomalies, as well

as plenty of unlabeled samples. We call this problem as anomaly detection with partially observed anomalies, which can not be directly handled by traditional techniques.

Previous methods are not suitable for this problem. Since no negative samples are provided, supervised learning based method is unfeasible for this task. As for unsupervised learning based methods, without using the information of observed anomalies, the performance may become pretty unsatisfactory. PU learning deal with the task where labeled positive and unlabeled samples are provided. However, the anomalies are not similar to each other, making the PU learning assumption not that suitable.

We propose a method called **ADOA** (Anomaly Detection with partial Observed Anomalies) to solve it. **ADOA** follows a two-stage manner. In stage one, we address the difference between observed anomalies, thus we first cluster the recognized anomalies into k different clusters. Later, we address that the anomalies can first be easily isolated from normal samples, at the same time, they should be similar to the observed anomalies, thus we filter potential anomalies and reliable normal samples from the unlabeled samples according to the isolation score and the similarity score to their nearest anomalies cluster centers. In stage two, we built a multi-class model to distinguish different anomalies from the normal samples. We run experiments on both synthetic and lots of different real-world datasets, which comes from diversified fields. The results on different datasets validate that existing approaches can not perform satisfactorily on this problem and the proposed method performs significantly better than existing methods. Furthermore, we apply the proposed method to the problem of malicious URL detection, the result also demonstrates the effectiveness of the method.

ACKNOWLEDGMENTS

This research was partially supported by NSFC (61333014). Most work was conducted during Ya-Lin Zhang's internship in Ant Financial.

REFERENCES

[1] Eric Bair. 2013. Semi-supervised Clustering Methods. *Wiley Interdisciplinary Reviews: Computational Statistics* 5, 5 (2013), 349–361.
[2] PS Bradley, KP Bennett, and Ayhan Demiriz. 2000. Constrained k-means Clustering. *Microsoft Research, Redmond* (2000), 1–8.
[3] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: Identifying Density-based Local Outliers. In *ACM Sigmod Record*, Vol. 29. ACM, 93–104.
[4] Guilherme O Campos, Arthur Zimek, Jörg Sander, Ricardo JGB Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E Houle. 2016. On the Evaluation of Unsupervised Outlier Detection: Measures, Datasets, and an Empirical Study. *Data Mining and Knowledge Discovery* 30, 4 (2016), 891–927.
[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly Detection: A Survey. *ACM Computing Surveys (CSUR)* 41, 3 (2009), 15.
[6] O Chapelle, B Schölkopf, and A Zien. 2006. Semi-supervised Learning. (2006).
[7] Yunqiang Chen, Xiang Sean Zhou, and Thomas S Huang. 2001. One-class SVM for Learning in Image Retrieval. In *Proceeding of 2001 International Conference on Image Processing*, Vol. 1. IEEE, 34–37.
[8] Marthinus C du Plessis, Gang Niu, and Masashi Sugiyama. 2014. Analysis of Learning From Positive and Unlabeled Data. In *Advances in Neural Information Processing Systems*. 703–711.
[9] Marthinus C du Plessis, Gang Niu, and Masashi Sugiyama. 2015. Convex Formulation for Learning from Positive and Unlabeled Data. In *Proceeding of the 32nd International Conference on Machine Learning*. 1386–1394.
[10] Eleazar Eskin, Andrew Arnold, Michael Prerau, Leonid Portnoy, and Sal Stolfo. 2002. A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. *Applications of Data Mining in Computer Security* 6 (2002), 77–102.
[11] Yong Ge, Hui Xiong, Zhi-hua Zhou, Hasan Ozdemir, Jannite Yu, and Kuo Chu Lee. 2010. Top-eye: Top-k Evolving Trajectory Outlier Detection. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. ACM, 1733–1736.
[12] Nico Görnitz, Marius Micha Kloft, Konrad Rieck, and Ulf Brefeld. 2013. Toward Supervised Anomaly Detection. *Journal of Artificial Intelligence Research* (2013).
[13] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support Vector Machines. *IEEE Intelligent Systems and Their Applications* 13, 4 (1998), 18–28.
[14] Rolf Isermann and Peter Balle. 1997. Trends in the Application of Model-based Fault Detection and Diagnosis of Technical Processes. *Control Engineering Practice* 5, 5 (1997), 709–719.
[15] Edwin M Knorr, Raymond T Ng, and Vladimir Tucakov. 2000. Distance-based Outliers: Algorithms and Applications. *the International Journal on Very Large Data Bases* 8, 3-4 (2000), 237–253.
[16] Yufeng Kou, Chang-Tien Lu, Sirirat Sirwongwattana, and Yo-Ping Huang. 2004. Survey of Fraud Detection Techniques. In *Proceeding of the 11st IEEE International Conference on Networking, Sensing and Control*, Vol. 2. IEEE, 749–754.
[17] Wee Sun Lee and Bing Liu. 2003. Learning with Positive and Unlabeled Examples using Weighted Logistic Regression. In *Proceeding of the 20th International Conference on Machine Learning*, Vol. 3. 448–455.
[18] Elizabeth Leon, Olfa Nasraoui, and Jonatan Gomez. 2004. Anomaly Detection based on Unsupervised Niche Clustering with Application to Network Intrusion Detection. In *IEEE Conference on Evolutionary Computation*, Vol. 1. IEEE, 502–508.
[19] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. 2003. Building Text Classifiers Using Positive and Unlabeled Examples. In *Proceeding of the 3rd IEEE International Conference on Data Mining*. IEEE, 179–186.
[20] Bing Liu, Wee Sun Lee, Philip S Yu, and Xiaoli Li. 2002. Partially Supervised Classification of Text Documents. In *Proceeding of the 19th International Conference on Machine Learning*, Vol. 2. 387–394.
[21] Bo Liu, Yanshan Xiao, Longbing Cao, Zhifeng Hao, and Feiqi Deng. 2013. SVDD-based Outlier Detection on Uncertain Data. *Knowledge and Information Systems* (2013), 1–22.
[22] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation Forest. In *Proceeding of the 8th IEEE International Conference on Data Mining*. IEEE, 413–422.
[23] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2012. Isolation-based Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 6, 1 (2012), 3.
[24] Xu-Ying Liu and Zhi-Hua Zhou. 2011. Towards Cost-sensitive Learning for Real-world Applications. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 494–505.
[25] Zhigang Liu, Wenzhong Shi, Deren Li, and Qianqing Qin. 2006. Partially Supervised Classification: Based on Weighted Unlabeled Samples Support Vector Machine. *International Journal of Data Warehousing and Mining (IJDWM)* 2, 3 (2006), 42–56.
[26] Gerhard Münz, Sa Li, and Georg Carle. 2007. Traffic Anomaly Detection using k-means Clustering. In *GI/ITG Workshop MMBnet*.
[27] Hoang Vu Nguyen, Hock Hee Ang, and Vivekanand Gopalkrishnan. 2010. Mining Outliers with Ensemble of Heterogeneous Detectors on Random Subspaces. In *International Conference on Database Systems for Advanced Applications*. Springer, 368–383.
[28] Salima Omar, Asri Ngadi, and Hamid H Jebur. 2013. Machine Learning Techniques for Anomaly Detection: an Overview. *International Journal of Computer Applications* 79, 2 (2013).
[29] Shebuti Rayana. 2016. ODDS Library. (2016). http://odds.cs.stonybrook.edu
[30] Rowland R Sillito and Robert B Fisher. 2008. Semi-supervised Learning for Anomalous Trajectory Detection.. In *the British Machine Vision Conference (BMVC)*, Vol. 27. 1025–1044.
[31] Ming-Yang Su. 2011. Real-time Anomaly Detection Systems for Denial-of-Service Attacks by Weighted k-nearest-neighbor Classifiers. *Expert Systems with Applications* 38, 4 (2011), 3492–3498.
[32] Swee Chuan Tan, Kai Ming Ting, and Tony Fei Liu. 2011. Fast Anomaly Detection for Streaming Data. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, Vol. 22. 1511.
[33] Hua Tang and Zhuolin Cao. 2009. Machine Learning-based Intrusion Detection Algorithm. *Journal of Computational Information Systems* 5, 6 (2009), 1825–1831.
[34] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, and others. 2001. Constrained k-means Clustering with Background Knowledge. In *Proceeding of the 18th International Conference on Machine Learning*, Vol. 1. 577–584.
[35] Su-Yun Wu and Ester Yen. 2009. Data Mining-based Intrusion Detectors. *Expert Systems with Applications* 36, 3 (2009), 5605–5612.
[36] Xiaojin Zhu. 2006. Semi-supervised Learning Literature Survey. *Computer Science, University of Wisconsin-Madison* 2, 3 (2006), 4.
[37] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. 2012. A Survey on Unsupervised Outlier Detection in High-dimensional Numerical Data. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 5, 5 (2012), 363–387.