2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Solving Unbalanced Problems in Similarity Learning using SVM Ensemble

Peipei Xia
School of Computer Science and Technology
& Provincial Key Laboratory for
Computer Information Processing Technology
Soochow University
Suzhou 215006, Jiangsu, China
Email: 20124227054@suda.edu.cn

Li Zhang
School of Computer Science and Technology
& Provincial Key Laboratory for
Computer Information Processing Technology
Soochow University
Suzhou 215006, Jiangsu, China
Email: zhangliml@suda.edu.cn

***Abstract*—Similarity learning is one of the most fundamental notions in machine learning and pattern recognition. In real-world problems, the number of the paired-samples in similarity set is far less than the ones in dissimilarity set. In other word, there is an unbalanced problem in the paired-samples of similarity learning. This paper presents a scheme of SVM ensemble to solve it. In our scheme, we randomly select some of samples to construct paired-samples, not producing all the paired-samples, and introduces multiple classifiers to obtain higher stability and reliability. As a result, the SVM ensemble can effectively decrease the number of paired-samples in similarity learning and solve the unbalanced data learning to some degree. In the experiments, the SVM ensemble is compared with some classic unbalanced learning algorithms. The results on classification tasks show that the SVM ensemble gains better performance.**

I. Introduction

In machine learning and pattern recognition, many classic algorithms adopt similarity or distance metric, such as K-NN (nearest neighbors) [1], SVM (Support Vector Machine) [2], [3], [4], etc. Traditionally, similarity or distance metric is preferentially appointed in these algorithms, such as Euclidean distance, which is independent of specific problems. The traditional way can not be applied to many complex tasks of information analysis, recognition, retrieval and others. Thus, similarity learning and distance metric is becoming a popular research subject in the field of machine learning, pattern recognition, image sciences, computer vision, etc.

In similarity learning, it requires to construct paired patterns (paired-samples), which are the objects for machines to learn. When solving real-world problems especially multi-class problems, it is quite obvious that most of the paired-samples are made up of dissimilar patterns, only a few ones consist of similar patterns. In other word, this is an unbalanced problem. It is well known that a balanced dataset provides improved overall classification performance compared to an unbalanced dataset for some classifiers [5], [6], [7]. Consequently, it is necessary to solve unbalanced problems in similarity learning.

There may exist two unbalanced situations in a dataset. The first type is between-class unbalance or class unbalance, in which case some classes have much more instances than others [8]. The other type is within-class unbalance or case unbalance, in which case some subsets have much fewer instances than other subsets in one class [9]. Here, by unbalanced problem, we mean the first one. By convention, in unbalanced dataset, we call the classes containing more instances the majority (common) class, while the ones containing fewer instances are called the minority (rare) classes.

Methods for solving unbalanced problems can be grouped into two categories, or the data level and algorithmic level. The main idea behind the methods at data level is to alter the distribution of data in order to provide a balanced dataset for classifiers. The methods at algorithmic level improve the performance by modifying algorithms themselves [8], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30]. We focus on the methods at data level which include some classical sampling methods [8]. Typically, sampling methods in unbalanced data learning is to modify the unbalanced dataset to get a relatively balanced distribution.

The simplest sampling methods are random over-sampling and random under-sampling. The former augments the minority class by exactly replicating the randomly selected examples from the minority class, while the latter randomly removes some examples from the majority class. The random over-sampling and under-sampling methods appear to be functionally equivalent since they both alter the size of the original dataset and can actually provide the same proportion of balance. However, this commonality is only superficial. Random over-sampling may make the decision regions of the learner smaller and more specific, leading to over-fitting [10]. Random under-sampling may cause classifiers to miss important concepts pertaining to the majority class. Thus, many modified sampling methods have been presented. Kubat et al. presented a heuristic under-sampling method which balances a dataset through just eliminating the noise and redundant examples from the majority class [11]. Synthetic sampling is another important part of sampling methods. The synthetic minority over-sampling technique (SMOTE) is a powerful method that has shown a great deal of success in various applications [12]. SMOTE generates new synthetic examples along the line between the minority examples and their randomly selected nearest neighbors. Han et al. presented two novel over-sampling methods based on SMOTE, or borderline-SMOTE1 and borderline-SMOTE2 [13]. The examples on the borderline and the ones nearby (called borderline examples in [13]) are more apt to the misclassified than the ones far from borderline, and more important for classification. Borderline-SMOTE

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

methods only strengthen the borderline minority examples. First, this method finds out the borderline minority examples; then, synthetic examples are generated from them and added to the original dataset. It shows that synthetic sampling methods are effective in dealing with unbalanced datasets. However, the data generation methods discussed above have a highly computational complexity. Noting that the essential problem of over-fitting is caused by random over-sampling, Mease et al. proposed a much simpler technique, over/under-sampling with jettering (JOUS-Boost) [10]. In each iteration of boosting, JOUS-Boost introduces independently and identically distributed (iid) noise into minority examples and generates new samples. This algorithm is relatively simple compared to synthetic sampling counterparts and also incorporates the benefits of boosted ensembles to improve performance. Based on the characteristics of the given data distribution, four KNN under-sampling methods were proposed, namely, NearMiss-1, NearMiss-2, NearMiss-3 and the "most distant" method [14].

The integration of sampling strategies with ensemble learning has also been studied. For example, the SMOTEBoost algorithm is based on the idea of integrating SMOTE with AdaBoost.M2 [15]. Specifically, SMOTEBoost introduces synthetic sampling in each boosting iteration. Another integrated approach, the DataBoost-IM method, combines the data generation techniques introduced in [17] with AdaBoost.M1 to achieve high predictive accuracy for the minority class without sacrificing accuracy on the majority class [16]. The Granular Support Vector Machines-Repetitive Under-sampling algorithm (GSVM-RU) was proposed in [18] to integrated SVM with under-sampling. The GSVM-RU method uses the SVM itself as a mechanism for under-sampling to sequentially develop multiple information granules with different information samples, which are later combined to develop a final SVM for classification.

The methods mentioned above could be directly used to solve the unbalanced problem in similarity learning. When applying these methods, it requires to generate an unbalanced problem first. In other words, all paired-samples which are unbalanced must be obtained from the original dataset, and then these methods are used for generating a balanced dataset. However, it is time-consuming for generating all paired-samples. Even if we generate all paired-samples, we still try to discard most of them by using these classical sampling methods. Thus, it costs a lot. In this paper, we apply SVM ensemble to solve unbalanced problem in similarity learning. In our scheme, we randomly select some of samples to construct paired-samples to decrease the number of paired-samples and balance the data distribution. The randomly selected samples would play an important role in the performance of SVM. To eliminate the randomness in selecting samples, ensemble learning is introduced here. Since selecting samples randomly could bring the diversity for ensemble learning, it is appropriate for introducing ensemble techniques.

The structure of this paper is organized as follows. Section II gives a brief introduction to unbalanced problems in similarity learning and reviews some classic under-sampling methods. Section III describes our proposed similarity learning method solving unbalanced problems using SVM ensemble in details. Section IV compares the proposed method, SVM ensemble, with other under-sampling methods and gives experimental results. Section V draws conclusions.

II. Related work

In this section, we have a brief review on unbalanced problems in similarity learning and some classic sampling methods in unbalanced data learning.

A. Unbalanced problem in similarity learning

Let the set of multi-class training samples be $X = \{\mathbf{x}_i,\ y_i\}_{i=1}^{l}$, where $\mathbf{x}_i \in R^d$ is the ith training sample, $y_i \in \{1, 2, \cdots, c\}$ is the label of $\mathbf{x}_i$, l is the number of training samples, and c is the number of classes. As Phillips described in [31], two sets are generated in difference space. One is the within-class differences set S, the other one is the between-class differences set D. In this way, we can formulate a multi-class classification problem into a classic binary one, which can be easily dealt with the traditional SVM. The within-class differences set S and between-class differences set D are formally described as

$$S = \{(\mathbf{x}_i - \mathbf{x}_j) | \mathbf{x}_i \sim \mathbf{x}_j, y_i = y_j, i, j = 1, \cdots, l\} \quad (1)$$

and

$$D = \{(\mathbf{x}_i - \mathbf{x}_j) \,|\mathbf{x}_i \not\sim \mathbf{x}_j, y_i \neq y_j, i, j = 1, \cdots, l\} \quad (2)$$

where $(\mathbf{x}_i - \mathbf{x}_j)$ is a paired-sample consisting of two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ in difference space. $\mathbf{x}_i \sim \mathbf{x}_j$ and $\mathbf{x}_i \not\sim \mathbf{x}_j$ indicate the two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ are in the same class (within-class) and in different class (between-class), respectively. Define that the labels of the paired-samples in the within-class differences set S are $+1$, while the labels of the paired-samples in the between-class differences set D are -1.

Obviously, the number of the generated training paired-samples is about l^2, which leads to an excessive number of training paired-samples. In addition, the number of the constructed dissimilar paired-samples is much larger than the number of the similar ones in multi-class problems. This is an unbalanced problem to be solved in similarity learning.

B. Classic sampling methods in unbalanced data learning

Sampling methods play an important role in solving unbalanced problems. Generally speaking, these methods provide a balanced distribution by using different ways. In other word, these methods add some instances into the minority class or remove some examples from the majority class according to the required proportion of balance.

Random over-sampling and random under-sampling are the simplest sampling methods. Random over-sampling augments the original minority class by exactly copying several randomly selected examples, while random under-sampling removes some instances from the majority class randomly. At the first blush, random over-sampling and under-sampling methods seem to be functionally equivalent. Nevertheless, each of them introduces its own drawbacks. In the case of random under-sampling, the disadvantage is quite obvious: removing data means missing information, maybe very important to classifiers. Thus, it is probable to degrade the performance on the majority class. In the situation of random over-sampling, the problem is not visualized enough. Several copies of certain

examples may become "tied", making the decision region smaller and more specific, eventually leading to over-fitting problems [10].

Synthetic sampling with data generation is another significant part of sampling. Synthetic minority over-sampling technique (SMOTE) is a representative one in this community [12]. The SMOTE algorithm generates new synthetic instances along the line between the minority instance and its selected nearest neighbors. The detailed steps of SMOTE are illustrated as follows. For each instance $\mathbf{x}$ in the minority class, we first find its K nearest neighbors in the same class. Then we randomly select some of the neighbors according to the amount of SMOTE. Next, for every selected neighbors $\mathbf{x}'$, we compute the difference $\mathbf{dif}$ between $\mathbf{x}$ and $\mathbf{x}'$, or $\mathbf{dif} = \mathbf{x}' - \mathbf{x}$, multiply $\mathbf{dif}$ by a random number gap between 0 and 1. Finally, we get the new synthetic example $\mathbf{Synthetic}$. The formal description is

$$\mathbf{Synthetic} = \mathbf{x} + gap \times \mathbf{dif}$$

In summary, SMOTE has overcome the drawback of random over-sampling that may cause overfitting, and also made the decision region larger and more general.

Zhang et al. proposed four informed under-sampling methods based on K-Nearest Neighbor (KNN) classifier, namely, NearMiss-1, NearMiss-2, NearMiss-3 and the "most distant" method [14]. The NearMiss-1 method selects the majority class examples that are close to *some* of the minority examples. In this method, the majority examples would be selected when their average distances to three closest minority examples are smallest. NearMiss-2 selects majority examples that are close to *all* minority examples. That is, we select the majority examples whose average distances to three farthest minority examples are smallest. In NearMiss-3, for each example in minority class, we select a given number of the closest majority examples. This method guarantees every minority example is surrounded by some majority class. Finally, the "most distant" method selects the majority class examples whose average distance to the three closest minority class examples are the largest.

These classical methods could be directly used to solve the unbalanced problem in similarity learning. However, all paired-samples which are unbalanced must be obtained from the original dataset when applying these methods. In fact, it is time-consuming for generating all paired-samples. Even if we could generate all paired-samples, we still want to discard most of them by using these classical sampling methods. Thus, it costs a lot.

III. SVM ENSEMBLE FOR UNBALANCED PROBLEM IN SIMILARITY LEARNING

In this section, we apply SVM ensemble to solve unbalanced problem in similarity learning. Typically, SVM ensemble learning consists of two sub-procedures. The first is how to generate individual SVMs. The second is how to combine the predictions into a final result. The proposed method will be described from these two parts.

A. Generating individual SVMs

As described in Section II-A, traditionally, there exists an unbalanced problem in similarity learning, which has an influence up on the performance of standard SVM. To avoid this, the paired-samples must be balanced. Our strategy is that for each example $\mathbf{x}$ in the original training set X, just pick the same amount of the examples from the same class and the different class, not all the examples in X, to construct paired-samples. In ensemble learning, individual learners should be as diverse as possible. Thus, we randomly select the examples to construct paired-samples for each individual SVM. In this way, the individual SVMs can be definitely diverse. In the same time, it guarantees that each example in the original training set is with the same weight.

Here we display the sub-procedure of generating individual SVMs in formal description. Let $X = \{\mathbf{x}_i,\ y_i\}_{i=1}^{l}$ be the original training set. For each individual SVM, it is necessary to construct a new training set of paired-samples in difference space. Assume that there is N individual SVMs totally, then N new training sets $Z_1, Z_2, \cdots, Z_N$ are required. To construct $Z_n, n = 1, 2, \cdots, N$, we first randomly select some samples $\mathbf{x}_j$ in the original training set. For each of them, then we randomly select k examples from the its class and k examples from the different class, and respectively generate the within-class difference set S_n and the between-class difference set D_n by using them. Finally, $Z_n = S_n \cup D_n$.

After gaining $Z_n, n = 1, 2, \cdots, N$, use N standard SVMs to train. We can get N train models. The standard SVM is to solve the convex quadratic programming problem as follows:

$$\begin{aligned} \max_{\alpha} \ & \sum_{i=1}^{m} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j v_i v_j k\left(\mathbf{z}_i, \mathbf{z}_j\right) \\ s.t. \ & \sum_{i=1}^{m} \alpha_i v_i = 0 \\ & 0 \le \alpha_i \le C\,, \quad i = 1, 2, \cdots, m \end{aligned} \tag{3}$$

where $\mathbf{z}_i \in R^d$ is the paired-sample, $v_i \in \{-1, +1\}$ is the label for $\mathbf{z}_i$, $k\left(\mathbf{z}_i, \mathbf{z}_j\right)$ is the kernel function, α_i is the Lagrange multiplier, m is the number of the paired-samples, and $C > 0$ is the regular factor.

In the test procedure of SVM, we input the paired-samples in test set into the training model obtained above. The discriminant function is described by

$$\hat{v} = f\left(\mathbf{z}\right) = \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i v_i k\left(\mathbf{z}_i, \mathbf{z}\right) + b\right) \tag{4}$$

where $\operatorname{sgn}(\cdot)$ is the sign function, $\mathbf{z}$ is a test paired-sample, and $\hat{v}$ is the estimated label of $\mathbf{z}$. For any $0 < \alpha_j < C$, the threshold b can be computed under KKT conditions by the following equation:

$$b = v_j - \sum_{i=1}^{n} \alpha_i v_i k\left(\mathbf{z}_i, \mathbf{z}\right) \tag{5}$$

B. Combining multiple outputs

Traditionally, a test set for similarity learning is formed in this way: for a test example $\mathbf{x}$, construct the paired-samples with each examples in the original training set. Since the scale of test set is quite large when the original training set contains too much examples, it is a big challenge in computation complexity. To speed up the algorithm, it requires to cut down the scale of test set to some degree. We randomly

select k' examples in each class from the original training set, and express the selected subset of the original training set as X'. For a test example $\mathbf{x}$, construct paired-samples with the examples in X' instead of all the examples in the original training set. Until now, the test set T is obtained.

Each individual SVM corresponds to a specific train model. For each paired-sample in T, there results in N outputs obtained from all SVMs. In the following, we describe the combination rules for these outputs.

It is well known that the test paired-sample $\mathbf{z}$ is made up of two samples, the test sample $\mathbf{x}$ and any example $\mathbf{x}_i$ in X. Also, (4) only determines these two samples are similar or not, and cannot indicate the class attribute of the unseen samples $\mathbf{x}$. For the application of classification, it needs to utilize similarity to determine the class of test samples.

To determine the class label of an unseen sample $\mathbf{x}$, we construct the set of test paired-samples $T = \{\mathbf{z'}_i\}_{i=1}^{k' \times c}$, where $\mathbf{z'}_i = (\mathbf{x} - \mathbf{x}_i) \in R^d$ constructed with the original training samples $\mathbf{x}_i$ in X'. Note that the training samples in X' are rearranged according to their labels, or the first k' samples belongs to class 1 and so on. By (4), we can obtain $\hat{v}_{ni}, n = 1, \cdots, N, i = 1, \cdots, k' \times c$. Each $\hat{v}_{ni}$ represents whether the test sample $\mathbf{x}$ is similar to the training sample $\mathbf{x}_i$ according to the nth SVM. Of course, the labels of $\mathbf{x}_i$ are known since they are training samples.

To combine the outputs of multiple SVMs, we introduce some combination rules, including "voting then classifying" (or VC) rule, the "classifying then voting" (or CV) rule, the maximum (or MAX) rule, the minimum (or MIN) rule and the average (or AVG) rule. In the following, we describe these rules.

1) VC rule: For the test paired-sample $\mathbf{z}_i$, the VC rule first combines the N similarities obtained from N SVMs. Namely,

$$\hat{v}'_i = \operatorname{sgn}\left(\frac{\sum_{n=1}^{N}(\hat{v}_{ni}+1)}{2N} - \frac{1}{2}\right) \tag{6}$$

If $\hat{v}'_i = 1$, then we think that the test sample $\mathbf{x}$ is similar to the training sample $\mathbf{x}_i$. Otherwise, they are not similar when $\hat{v}'_i = -1$. Then we compute the similarity probability of the text sample in each class, which is defined as

$$P_j(\mathbf{x}) = \frac{\sum_{i=(j-1)\times k'+1}^{j\times k'}(\hat{v}'_i + 1)}{2k'}, j = 1, \cdots, c \tag{7}$$

We classify the test sample $\mathbf{x}$ according to the maximum similarity probability. Namely,

$$\hat{y} = \arg\max_{j=1,\cdots,c} P_j(\mathbf{x}) \tag{8}$$

2) CV rule: In the CV rule, the estimated classification labels on the test sample $\mathbf{x}$ are first obtained, and then these classification results are combined by some way. The similarity probability in each class generated by the nth classifier is expressed as:

$$P_{nj}(\mathbf{x}) = \frac{\sum_{i=(j-1)\times k'+1}^{j\times k'}(\hat{v}_{ni} + 1)}{2k'}, j = 1, \cdots, c, n = 1, \cdots, N \tag{9}$$

By using these probabilities, we give the classification results

$$\hat{y}_n = \arg\max_{j=1,\cdots,c} P_{nj}(\mathbf{x}) \tag{10}$$

where $\hat{y}_n$ is the estimated classification label on $\mathbf{x}$ obtained from the nth classifier. According to the majority voting rule, these classification labels $\hat{y}_n, n = 1, \cdots, N$ determine the final estimate label for $\mathbf{x}$.

3) MAX rule: Given the similarity probability $P_{nj}, n = 1, \cdots, N, j = 1, \cdots, c$, the MAX rule is to estimate the label for $\mathbf{x}$ by

$$\hat{y} = \arg\max_{j=1,\cdots,c} \max_{n=1,\cdots,N} P_{nj} \tag{11}$$

4) MIN rule: Given the similarity probability $P_{nj}, n = 1, \cdots, N, j = 1, \cdots, c$, the MIN rule is to estimate the label for $\mathbf{x}$ by

$$\hat{y} = \arg\max_{j=1,\cdots,c} \min_{n=1,\cdots,N} P_{nj} \tag{12}$$

5) AVG rule: The AVG Rule is parallel to the MAX rule and MIN rule. The only difference is that the final similarity probability of each class is the average value of the results in individual SVMs, which can be expressed by

$$\hat{y} = \arg\max_{j=1,\cdots,c} \frac{\sum_{n=1}^{N} P_{nj}}{N} \tag{13}$$

IV. Experiments

In order to validate the effectiveness of the proposed method for solving unbalanced problems, we select some popular datasets, including Iris dataset from UCI [32], the MNIST database of handwritten digits [33] and the UMIST Face Database [34]. The compared methods are four representative under-sampling methods, which are random under-sampling, NearMiss-1, NearMiss-2, and NearMiss-3 method.

Some experiment settings are explained here, such as the data pre-processing and some parameters. First, a data normalization processing is needed. All the selected datasets are mapped into the interval $[0, 1]$. Second, the kernel function of SVM is the popular radial basis function (RBF), which is denoted by $k(\mathbf{z}, \mathbf{z}') = \exp\left\{-\gamma\|\mathbf{z} - \mathbf{z}'\|^2\right\}$, where $\gamma > 0$ is the kernel parameter. The value of γ is determined according to the training examples [36]. There is another parameter related to SVM, which is called the regular factor C. In our experiment, C is determined via 5 fold cross-validation, ranging from $\{10^{-3}, 10^{-2}, \cdots, 10^3\}$. The results of cross-validation show that the performance is best when C is equal to 10. Hence, we set $C = 10$ in the following experiments. The four sampling methods sample the data to a completed balanced data distribution.

A. Iris dataset

The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. Iris dataset is one of the most popular datasets.

The Iris dataset contains 3 classes. Each class describes a type of iris plants. Note that one class is linearly separable

from the other two, while the latter are not linearly separable from each other. 50 instances are included in each class, 150 in total. Each sample has 4 attributes. In the experiment, half of the samples are selected randomly as training samples in each class, while the rest as test samples. We perform 10 times and report the average result.

To construct the training set of paired-samples for SVMs, we first randomly select k samples in each class from the training set. For each selected sample, we randomly select k other examples in the same class and k examples in the different class. In the case of constructing test paired-samples, we randomly select k' examples from each class for a test sample. Here, let $k = k' = 10$.

We compare the performance of five combination rules when varying the number of individual SVMs N. The number of individual SVMs N takes value from the set $\{5, 10, 15, 20, 30, 40, 50, 100\}$. Figure 1 and Table I show test errors versus (vs.) different N for our method, and Table II gives test errors obtained from other four methods. The performance index "Number" in these tables means the number of paired-samples in the training set in SVM ensemble. While in the four under-sampling methods, it means the number of paired-samples after/without under-sampling.

The running time in Table I includes the time of constructing training and test paired-sample sets, the time of training all SVMs and the time of test all samples in the test set. Note that the running time in Table I includes the time of training N SVMs, while the running time in Table II just contains that of one SVM.

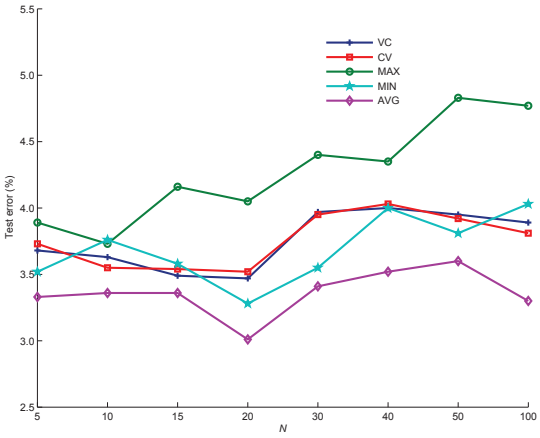

Fig. 1. Test errors versus different N on the Iris dataset

From Figure 1, it is obvious that the AVG rule is the best one with highest accuracy and strongest stability among all the five rules. From Table I, we can see that the number of SVMs has a little effect on the classification performance, but has a great effect on the running time. Table II indicates that the accuracy of the four under-sampling methods is quite high, which is compared to our methods. However, the time spent on the sampling procedure cannot be ignored, especially in NearMiss-1, NearMiss-2 and NearMiss-3. When the scale of dataset is much larger, the re-sampling time is insufferable, which will be reflected in the following experiments. When N is small enough, the running time is competitive to the other methods.

In the proposed method, the accuracy is pretty good among all the five rules when N is 20. Considering both accuracy and running time, we choose the AVG rule as the combination rule of individual SVMs, and set N to 15 in the following experiments.

B. MNIST database of handwritten digits

The MNIST database of handwritten digits is a popular database in machine learning and pattern recognition. The database has a training set of 60,000 instances and a test set of 10,000 instances, which is a subset of a larger set available from NIST. It contains 10 classes, namely class 0-9. Each instance is a gray level image with the size of 28×28 pixel. Figure 2 shows some samples from the MNIST database. In this experiment, we just select five classes, class 1, 3, 7, 8, and 9. Considering the huge expense on sampling procedure in the under-sampling methods, 50 training examples and 100 test examples are picked from each class randomly. That is, there are 250 training examples and 500 test examples in total. As the dimension is quite high, dimensionality reduction is in badly need. Influenced by the recent upsurge of compressed sensing, random projection is becoming an effective and fast dimensionality reduction method [35], [36]. Here, random projection is used on the original samples. Namely,

$$\bar{\mathbf{x}}_i = \mathbf{R}\mathbf{x}_i \tag{14}$$

where $\bar{\mathbf{x}}_i \in R^{d'}$ is the corresponding sample of $\mathbf{x}_i$ in a projected subspace, $\mathbf{R} \in R^{d' \times d}$ is a random projection matrix, and d' is the dimensionality of the projected subspace. d' is set to 50 in this experiment. Similarly, we perform 10 trials and report an average result.

Fig. 2. Some samples in the MNIST database of handwritten digits

In the experiment of the MNIST database, we set $k = 10$ and $k' = 10$. As described in Section IV-A, we select the AVG rule as the combination strategy, and set N to 15. The results are illustrated in Table III.

From Table III, it is easy to find out that the proposed method not only has a higher accuracy but also a much faster runtime. The other four methods consume too much time on sampling. Without considering the runtime of the four under-sampling methods, more examples can be included in the training set, which will improve the performance of our method.

C. UMIST face database

The UMIST face database consists of 564 images of 20 people. Individuals cover a range of race/sex/appearance. Each

individual is shown in a range of poses from profile to frontal views. Figure 3 shows some samples in the UMIST face database. Each sample is an image of size 112×92. The dimensionality is quite high, so random projection is still used here. The dimensionality of projection space is also 50. The experiments is repeated 10 times. Since the number of the instances for each person is not the same and the number of individuals is relatively large, we select less examples to construct paired-samples, $k = 5$ and $k' = 5$. The results are shown in Table IV.

Fig. 3. Some samples in the UMIST face database

From Table IV, it is obvious that the prediction of the proposed method is best. Meanwhile, random under-sampling method has done a better job in running time. The UMIST database contains images of 20 people, thus the number of paired-samples in training set after random under-sampling is not very large. While in SVM ensemble, the SVM training procedure runs 15 times. The NearMiss-1, NearMiss-2 and NearMiss-3 method get a lower accuracy while pay a huge cost on cpu time, especially in NearMiss-2.

V. Conclusion

This paper deals with similarity learning using SVM ensemble which has a great advantage in solving the unbalanced problems. This problem is caused by constructing paired-samples in traditional way. From the results on the Iris dataset, the proposed method gets a competitive accuracy compared to the other under-sampling methods while the advantage of speed is not outstanding. The reason is that the Iris dataset is in small size. Meanwhile, the AVG rule gets the best performance when varying N. On the MNIST database and UMIST face database, the proposed method outperforms the other four methods in both prediction and running time. To sum up, the proposed method is meaningful, especially when the scale of dataset is relatively larger.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 61373093, 61033013, and 61271301, by the Natural Science Foundation of Jiangsu Province of China under Grant Nos. BK2011284 and BK201222725, by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China under Grant No.13KJA520001 and by the Qing Lan Project.

References

[1] T. Cover, P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol.13, no. 1, pp. 21-27, 1967.

[2] C.J.C. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp, 121-167, 1998.

[3] N. Cristianini and J. Shawe-Taylor, "Support Vector Machine," *Cambridge University Press*, 2000.

[4] L. Zhang, "Research on Support Vector Machines and Kernel Methods," Ph.D. thesis, Xidian University, 2009.

[5] G. M. Weiss, F. Provost, "The Effect of Class Distribution on Classifier Learning: An Empirical Study," *Technical Report ML-TR-43*, Dept. of Computer Science, Rutgers Univ. ,2001.

[6] J. Laurikkala, "Improving Identification of Difficult Small Classes by Balancing Class Distribution," *Proc. Conf. AI in Medicine in Europe: Artificial Intelligence Medicine*, pp. 63-66, 2001.

[7] A. Estabrooks, T. Jo, N. Japkowicz, "A Multiple Resampling Method for Learning from Imbalanced datasets," *Computational Intelligence*, vol. 20, pp. 18-36, 2004.

[8] N. V. Chawla, N. Japkowicz, "A. Kolcz, Editorial: Special Issue on Learning from Imbalanced datasets," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 1-6, 2004.

[9] G. Weiss, "Mining with rarity: A unifying framework," *ACM SIGKDD Explorations Newsletter*, vol. 6 no. 1, pp. 7-19, 2004.

[10] D. Mease, A. J. Wyner, "A. Buja, Boosted classification trees and class probability/quantile estimation," *The Journal of Machine Learning Research*, vol. 8, pp. 409-439, 2007.

[11] M. Kubat, S. Matwin, "Addressing the Course of Imbalanced Training Sets: One-sided Selection," *International Conference on Machine Learning*, pp. 179-186, 1997.

[12] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-Sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321-357, 2002.

[13] H. Han, W.Y. Wang, B.H. Mao, "Borderline-SMOTE: A New Over-Sampling Method in Imbalanced datasets Learning," *Proc. Int'l Conf. Intelligent Computing*, pp. 878-887, 2005.

TABLE I. Test errors versus different N on the Iris dataset

N	Number	Test error(%)					Running time(s)
		VC	CV	MAX	MIN	AVG	
5	1500	3.68	3.73	3.89	3.52	3.33	1.06
10	1500	3.63	3.55	3.73	3.76	3.36	2.17
15	1500	3.49	3.54	4.16	3.58	3.36	3.12
20	1500	3.47	3.52	4.05	3.28	**3.01**	4.10
30	1500	3.97	3.95	4.40	3.55	3.41	6.36
40	1500	4.00	4.03	4.35	4.00	3.52	8.63
50	1500	3.95	3.92	4.83	3.81	3.60	10.67
100	1500	3.89	3.81	4.77	4.03	3.30	21.86

TABLE II. Test errors obtained from four under-sampling methods on the Iris dataset

Methods	Number	Test error(%)	Running time(s)
Random under-sampling	3750/5625	3.10	0.65
NearMiss-1	3750/5625	3.34	2.80
NearMiss-2	3750/5625	2.89	3.00
NearMiss-3	3750/5625	2.32	4.30

[14] J. Zhang, I. Mani, "KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction," *Proceedings of Workshop on Learning from Imbalanced Datasets*, 2003.

[15] N. V. Chawla, A. Lazarevic, L. O. Hall, K. W. Bowyer, "SMOTEBoost: Improving Prediction of the Minority Class in Boosting," *Knowledge Discovery in Databases: PKDD 2003*, Springer, Berlin Heidelberg, pp. 107-119, 2003.

[16] H. Guo, H.L. Viktor, "Learning from Imbalanced datasets with Boosting and Data Generation: The DataBoost IM Ap-proach," *ACM SIGKDD Explorations Newsletter*, vol. 6. no. 1, pp. 30-39, 2004.

[17] H. Guo, H.L. Viktor, "Boosting with Data Generation: Improving the Classification of Hard to Learn Examples," *Proc. Int'l Conf. Innovations Applied Artificial Intelligence*, pp. 1082-1091, 2004.

[18] Y. Tang and Y. Q. Zhang, "Granular SVM with Repetitive Undersampling for Highly Imbalanced Protein Homology Prediction," *Proc. Int'l Conf. Granular Computing*, pp. 457-460, 2006.

[19] M. Joshi , V. Kumar, R. Agarwal, "Evaluating Boosting Algorithms to Classify Rare Classes: Comparison and Improvements," *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on. IEEE*, pp. 257-264, 2001.

[20] G. Wu, Y. C. Edward, "Class-Boundary Alignment for Imbalanced Dataset Learning,", *ICML 2003 workshop on learning from imbalanced datasets II*, Washington, DC, pp. 49-56, 2003.

[21] B. Raskutti, A. Kowalczyk, "Extreme Re-Balancing for SVMs: A Case Study," *ACM SIGKDD Explorations Newsletter*, vol. 6. no. 1, pp. 60-69, 2004.

[22] B. Scholkopt, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, "Estimating the Support of a High-Dimensional Distribution," *Neural Computation*, vol. 13, pp. 1443-1471, 2001.

[23] L. M. Manevitz, M. Yousef, "One-Class SVMs for Document Classification," *Machine Learning Research*, vol. 2, pp. 139-154, 2001.

[24] L. Zhuang, H. Dai, "Parameter Estimation of One-Class SVM on Imbalance Text Classification,", *Lecture Notes in Artificial Intelligence*, vol. 4013, pp. 538-549, 2006.

[25] H. J. Lee, S. Cho, "The Novelty Detection Approach for Difference Degrees of Class Imbalance," *ACM SIGKDD Explorations Newsletter*, vol. 4233, pp. 21-30, 2006.

[26] L. Zhuang, H. Dai, "Parameter Optimization of Kernel-Based One-Class Classifier on Imbalance Text Learning," *ACM SIGKDD Explorations Newsletter*, vol. 4099, pp. 434-443, 2006.

[27] N. Japkowicz, "Supervised versus Unsupervised Binary-Learning by Feedforward Neural Networks," *Machine Learning*, vol. 42, pp. 97-122, 2001.

[28] L. Manevitz, M. Yousef, "One-Class Document Classification via Neural Networks," *Neurocomputing*, vol. 70, pp.1466-1481, 2007.

[29] N. Japkowicz, "Learning from Imbalanced datasets: A Comparison of Various Strategies," *AAI workshop on learning from imbalanced data setsq*, pp. 10-15, 2000.

[30] N. Japkowicz, C. Myers, M. Gluck, "A Novelty Detection Approach to Classification," *Proc. Joint Conf. Artificial Intelligence*, pp. 518-523, 1995.

[31] P. J. Phillips, "Support vector machines applied to face recognition," *Advances in Neural Information Processing Systems 11*, 1998.

[32] P. M. Murphy, D. W. Aha, "UCI machine learning repository," *http://www.ics.uci.edu/ mlearn/MLRepository.html*, 1992.

[33] C. L. Liu, K. Nakashima, H. Sako, et al, "Handwritten digit recognition using state-of-the-art techniques, Frontiers in Handwriting Recognition, 2002", it Proceedings. Eighth International Workshop on. IEEE, pp. 320-325, 2002.

[34] "UMIST face database (n.d.)." *Retrieved June 6, 2003 from http://images.ee.umist.ac.uk/danny/database.html*

[35] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 210-227 , 2009.s

[36] L. Zhang, W. D. Zhou, P.-C. Chang, J. Liu, Z. Yan, T. Wang, F.-Z. Li, "Kernel Sparse Representation-Based Classifier," *IEEE Transactions on Signal Processing*, vol. 60, pp. 1684-1695, 2012.

TABLE III. TEST RESULTS ON THE MNIST DATABASE

Methods	Number	Test error(%)	Running time(s)
SVM ensemble	5000	13.58	481.2
Random under-sampling	25000/62500	18.10	1014.8
NearMiss-1	25000/62500	36.16	5971.1
NearMiss-2	25000/62500	39.24	7885.1
NearMiss-3	25000/62500	20.98	11457.0

TABLE IV. TEST ERRORS ON THE UMIST DATABASE

Methods	Number	Test error(%)	Running time(s)
SVM ensemble	2820	5.11	292.7
Random under-sampling	8904/79524	7.50	72.5
NearMiss-1	8904/79524	6.23	1200.6
NearMiss-2	8904/79524	15.32	1645.8
NearMiss-3	8904/79524	6.83	1576.1