Using clustering analysis to improve semi-supervised classification

Haitao Gan*, Nong Sang, Rui Huang, Xiaojun Tong, Zhiping Dan

* Institute for Pattern Recognition and Artificial Intelligence, Huazhong University of Science and Technology, Wuhan 430074, China
College of Mathematics and Computer Science, Wuhan Textile University, Wuhan 430073, China

A R T I C L E   I N F O
Article history:
Received 19 December 2011
Received in revised form 30 May 2012
Accepted 7 August 2012
Communicated by Qingshan Liu
Available online 3 October 2012

Keywords:
Self-training
Semi-supervised classification
Semi-supervised clustering
Fuzzy c-means
Support vector machine

A B S T R A C T
Semi-supervised classification has become an active topic recently and a number of algorithms, such as Self-training, have been proposed to improve the performance of supervised classification using unlabeled data. In this paper, we propose a semi-supervised learning framework which combines clustering and classification. Our motivation is that clustering analysis is a powerful knowledge-discovery tool and it may reveal the underlying data space structure from unlabeled data. In our framework, semi-supervised clustering is integrated into Self-training classification to help train a better classifier. In particular, the semi-supervised fuzzy c-means algorithm and support vector machines are used for clustering and classification, respectively. Experimental results on artificial and real datasets demonstrate the advantages of the proposed framework.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

Supervised classification has been widely used in pattern recognition and computer vision, including applications such as handwritten digit recognition, document classification, face recognition, object detection and tracking, etc. [1–6], which relies on sufficient labeled data to train a good classifier (sufficient usually means that the labeled data can roughly represent the underlying structure of the entire data space). However, labeled data are usually insufficient and hard to obtain since data labeling requires extensive expert effort and is often time-consuming. Meanwhile, unlabeled data are often abundant in real world, and may be extremely valuable for improving the classifier trained only by a small amount of labeled data. Consequently, semi-supervised classification, which uses both labeled and unlabeled data to train a classifier, has become a recent topic of interest. Many researchers have been studying the idea and various approaches have been proposed in the semi-supervised learning area, such as Self-training [7], Co-training [8,9], transductive support vector machines [10], generative models [11], and graph-based methods [12–14], etc. We refer the readers to some excellent surveys [15,16] for more details.

Our research is partially inspired by Self-training, which is a technique commonly used for semi-supervised classification [7]. As the name implies, a classifier is first trained by labeled data and used to classify unlabeled data. Consequently, unlabeled data that are classified with the highest confidence (probability of belonging to a certain class) are added incrementally to the labeled dataset with their predicted labels. The procedure is repeated until convergence. Self-training has been applied to many problems, such as word sense disambiguation [17] and subjective nouns [18]. However, Self-training will fail to approach the real data space if the labeled data cannot represent the underlying structure of the particular space, because the initial trained classifier will give bad results on the unlabeled data. Recently, an improved version of Self-training, called Help-training, is proposed in [19]. Its main idea is to use a generative model to iteratively select parts of the unlabeled data to help train the main discriminative classifier. Experimental results show that Help-training outperforms the standard Self-training. However, Help-training does not fundamentally solve the problem existing in Self-training since the generative model used to help select unlabeled data for the discriminative classifier is trained only by labeled data. When the data space represented by the labeled data is not consistent with the entire data space, Help-training does not help much. Fig. 1 shows an example of Help-training for semi-supervised support vector machines. This example demonstrates what may happen if we use a generative model to select unlabeled data that are classified with high confidence in order to help train a discriminative classifier. In this case, a bad result may occur, because samples between the real decision boundary and the decision boundary found by the labeled data will be classified into the wrong class, no matter a generative or discriminative classifier is used. In fact, for most semi-supervised methods, unlabeled data are classified individually by a classifier to help approach the real data space, without revealing the intrinsic structure information implied by all
the unlabeled data. This will result in blindness to some extent in semi-supervised classification.

Our intuition is that unlabeled data may be extremely valuable to reveal data space structure and model the relationships among labeled and unlabeled data prior to the training of a classifier. Similar ideas have been studied previously. Oliveira et al. [20] first use PCA or ICA on unlabeled and labeled data (ignoring the labels), and then map labeled data onto a subspace, and finally train a classifier base on the subspace. This approach learns good functional structures using unlabeled data. Ando and Zhang [21] take this strategy as structural learning and employ the structure in the training stage. But it does not make the best use of prior knowledge in the structural learning stage. We can use semi-supervised clustering methods to learn the structure better where labeled data are used to guide the clustering process.

In this paper, we propose a framework for semi-supervised classification where clustering and classification methods are combined together. In our framework, particularly, a semi-supervised clustering process is integrated into the Self-training process to help to train a better classifier. The main advantage of the framework is that it makes use of labeled data and unlabeled data to reveal the actual data space structure through clustering analysis to compensate for the limitation of labeled data. To sum up, this paper offers the following contributions:

(1) As unlabeled data may contain crucial information about the data space, we use clustering methods to reveal the underlying data space structure to improve the training of the classifier.
(2) Labeled data are used to guide the clustering process through semi-supervised clustering methods.
(3) Newly labeled data are used not only to update the classifier (as in Self-training), but also to better guide the semi-supervised clustering methods.

The rest of the paper is organized as follows: in Section 2, we briefly review the Semi-Supervised FCM (SSFCM) algorithm; in Section 3, we describe our algorithm that integrates SSFCM and Support Vector Machines (SVMs); in Section 4, we present the experimental results on several artificial and real datasets; finally, we conclude the paper and discuss some future directions in Section 5.

2. Review of semi-supervised fuzzy c-means algorithm

Clustering analysis is an important unsupervised tool for grouping unlabeled data [22]. Informally, clustering analysis tries to partition a given dataset into clusters so that data points in the same cluster are more similar, according to some measure, than those in different clusters. Fuzzy C-Means (FCM) is one of the most popular unsupervised clustering methods. In comparison to hard clustering, FCM provides an additional conceptual enhancement by allowing a data point to be assigned to different classes to various membership degrees. In this way, the patterns can be treated more reasonable and the algorithm is capable of identifying eventual “outliers”.

Let \( X = \{x_1, x_2, \ldots, x_n\} \), where \( x_i \in \mathbb{R}^d \) be a dataset of size \( n \) and dimension \( d \). FCM partitions the patterns by minimizing an objective function. Bezdek [23] defines the objective function as

\[
\min J_m = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m|x_j - v_i|^2
\]

where \( c \) is the number of classes, \( V \) represents the set of prototypes \( \{v_i\} \) associated with classes. \( U \) is a partition matrix whose element \( u_{ij} \) indicates the membership degree of the data point \( x_j \) to class \( i \) and satisfies two conditions: \( 0 \leq u_{ij} \leq 1 \) and \( \sum_{i=1}^{c} u_{ij} = 1 \). The superscript \( m \) is the degree of fuzziness associated with the partition matrix. \( d_{ij} = ||x_j - v_i|| \) expresses the distance between \( x_j \) and \( v_i \).

The key idea of semi-supervised clustering algorithms is to take advantage of different kinds of prior knowledge to improve the performance of clustering. Earlier research works have considered prior knowledge in the form of labels [24] or constraints [25]. In this paper, we will focus on the semi-supervised model where prior knowledge is provided in the form of labels. We employ the semi-supervised FCM algorithm proposed in [26,27] which extends the objective function of the standard FCM algorithm to capture the hidden and visible data space structures. The hidden data space structure is discovered using the original FCM objective function as the first term. The second term takes into account the visible data space structure reflected by labeled data. Formally, given a labeled set \( X = \{x_1, x_2, \ldots, x_n\} \), where \( x_i \in \mathbb{R}^d \), and denoting the label of data \( x_j \) by \( L(x_j) \in \{1, 2, \ldots, c\} \), the modified objective function of SSFCM is written as follows:

\[
\min J_m = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m|x_j - v_i|^2 + \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m|x_j - v_i|^2
\]

where \( u_{ij} \) and \( u_{ij} \) define the membership degrees of the unlabeled data point \( x_j \) and the labeled one \( x_j \) to class \( i \), respectively, which satisfy \( 0 \leq u_{ij} \leq 1 \), \( \sum_{j=1}^{n} u_{ij} = 1 \) and \( 0 \leq u_{ij} \leq 1 \), \( \sum_{i=1}^{c} u_{ij} = 1 \). An additional condition must be satisfied for the labeled data

\[
 u_{ij} \geq u_{ij}, \quad \forall k \in \{1, 2, \ldots, c\}/\{i\}, \quad j = 1, 2, \ldots, n\text{ if } L(x_j) = i
\]

Using the Lagrange multiplier method, we can obtain the iteration equations for optimization as follows:

For data prototype \( v_i \)

\[
v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j + \sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m + \sum_{j=1}^{n} u_{ij}^m}
\]

For unlabeled data point \( x_j \)

\[
u_{ij} = \frac{1}{\sum_{j=1}^{n} \left( \frac{d_{ij}}{d_{ij}} \right)}
\]
3. Our algorithm

In this section, we will give a framework for semi-supervised classification, where a semi-supervised clustering process is integrated into self-training. First, semi-supervised clustering using both labeled and unlabeled data is employed to learn the underlying data space structure and a classifier is trained using labeled data. Semi-supervised clustering produces the membership degrees of each unlabeled sample to different classes. The unlabeled sample that has higher certainty of belonging to one class (i.e., has one high value of membership degree) is then classified by the classifier trained using labeled data. The most confidently classified unlabeled data, together with their predicted labels, are added to the labeled set. The semi-supervised clustering algorithm re-clusters remaining unlabeled data and the classifier is re-trained. This procedure is repeated until all unlabeled data are labeled. In this way our framework combines clustering and classification in unison. Fig. 2 illustrates the procedure of our framework.

It is worth pointing out the difference between our algorithm and Help-training. Help-training employs labeled data to approach the data space and help train the discriminative classifier using a generative model. In this process, it only uses local inner-class information and will easily lead to local optima. Nevertheless, our algorithm applies clustering analysis to cluster labeled and unlabeled data and reveals the underlying information including global inner-class information. This will ensure the correctness of the training process and get the global optimum. Fig. 3 illustrates the intuition of our algorithm. It is the testing result of the toy dataset shown in Fig. 1. As can be seen, our algorithm outperforms Help-training. Data points between the decision boundary found by our algorithm and that by Help-training will be misclassified by Help-training.

![Flow chart of our framework](Image)

Fig. 2. Flow chart of our framework (L and U represent labeled and unlabeled data, respectively).

In our implementation, we picked the popular SSFCM and SVMs, respectively, for semi-supervised clustering and classification, but our framework is not limited to these algorithms. As mentioned previously, SSFCM is robust to outliers (e.g., incorrectly labeled data), because it uses fuzzy membership degrees instead of hard labeling. On the other hand, SVMs [28] are a powerful machine learning technique based on the principle of structural risk minimization. They can solve linearly nonseparable problems using kernel tricks and have shown excellent generalization performance. Detailed surveys of SVM can be found in [29].

A specific implementation of our algorithm, where SSFCM is used to help to train SVM, is described in Algorithm 1. We denote the confidence thresholds of the SSFCM and the SVM by $\epsilon_1$ and $\epsilon_2$, respectively. If the selected set $T_2$ by SVM is empty, the value of $\epsilon_1$ will drop 0.05 at each step. $N$ is the minimal number of unlabeled data.

**Algorithm 1.** SSFCM + SVM.

**Input:** a labeled dataset $L^{(0)}$, an unlabeled dataset $U^{(0)}$

**Output:** an SVM classifier

**Method:**

1. Initialize the dataset $L = L^{(0)}$ and $U = U^{(0)}$, threshold value $\epsilon_1$, $\epsilon_2$, $N$
2. Repeat until $|U| \leq N$
   - Estimate the membership degree using SSFCM for unlabeled data
   - Select a dataset $T_1$ where each sample $x_i$ has high certainty of belonging to one class, i.e., $\exists i, s.t. u_{ij} \geq \epsilon_1$, for $x_j$
   - Train the SVM with $L$
   - Compute the output $f(x)$ of the SVM for the selected dataset $T_1$
   - Select a dataset $T_2$ where the output of each sample $x$ by the SVM has high values, i.e., $f(x) \geq \epsilon_2$
   - Update the current labeled set $L \leftarrow L \cup T_2$
   - Update the current unlabeled set $U \leftarrow U \setminus T_2$
   - Reduce the value of $\epsilon_1$ if $T_2 = \emptyset$
3. Label the remaining unlabeled data with the trained SVM, if $U \neq \emptyset$
4. Retrain the SVM

4. Experiments

In our experiments we have used LIBSVM [30]. We have tested our algorithm on two artificial datasets and two real datasets.
from the UCI Dataset Repository [31]: IRIS and Waveform databases. The details are discussed below.

4.1. datasets

To evaluate the performance of our algorithm, we first investigated two artificial datasets: Gauss50 and Gauss50x. They are generated in the same way as in [19], each of which includes two classes in a 50-dimensional data space. Each class of Gauss50 is generated with equal probability from a Gaussian distribution with a unit covariance matrix. Class 1 is a Gaussian distribution with mean \((0.23, 0.23, \ldots, 0.23)\) and class 2 is a Gaussian distribution with mean \((-0.23, -0.23, \ldots, -0.23)\). Each class of Gauss50x is generated with equal probability from a Gaussian mixture distribution. Conditional distributions of the two classes are

\[
\begin{align*}
\Pr(x|y=1) &= 0.49N(\mu_1, I) + 0.51N(\mu_2, I) \\
\Pr(x|y=-1) &= 0.49N(-\mu_1, I) + 0.51N(-\mu_2, I)
\end{align*}
\]

(8)

here \(\mu_1 = (0.25, 0.25, \ldots, 0.25)\), \(\mu_2 = (-0.25, -0.25, \ldots, -0.25)\), \(N(\mu, I)\) is a Gaussian distribution with mean \(\mu\) and unit covariance matrix.

Besides the artificial datasets, we also tested two UCI datasets, IRIS and waveform, both of which are multi-class problems. Each dataset is divided into two subsets: 60% for training and 40% for testing. Table 1 shows the properties of these datasets.

For a fair comparison, the parameter of the RBF kernel for the SVM is set to \(1/\sigma^2 = 1/D\), where \(D\) is the dimension of the datasets. In the experiments, the multi-class classifier is trained using the one-against-all strategy.

4.2. Comparing our algorithm with Help-training and Self-training

To evaluate the performance of our algorithm, we run a series of experiments against Self-training and Help-training. We randomly divide the training set into two subsets: 10% as labeled data and 90% as unlabeled data. We repeat this process 10 times to test our algorithm and the other two methods. The results are shown in Tables 2–5.

Table 1
Description of experimental datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># attributes(D)</th>
<th># class</th>
<th># training size</th>
<th># testing size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss50</td>
<td>50</td>
<td>2</td>
<td>550</td>
<td>1000</td>
</tr>
<tr>
<td>Gauss50x</td>
<td>50</td>
<td>2</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>IRIS</td>
<td>9</td>
<td>3</td>
<td>90</td>
<td>50</td>
</tr>
<tr>
<td>Waveform</td>
<td>21</td>
<td>3</td>
<td>3000</td>
<td>2000</td>
</tr>
</tbody>
</table>

Table 2
Testing error rate on Gauss50.

<table>
<thead>
<tr>
<th>Selection</th>
<th>Supervised SVM with labeled data</th>
<th>Semi-supervised SVM with Self-training</th>
<th>Semi-supervised SVM with Help-training</th>
<th>Our algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>10.9</td>
<td>6.3</td>
<td>6.3</td>
<td>5.8</td>
</tr>
<tr>
<td>#2</td>
<td>15.2</td>
<td>7.0</td>
<td>7.0</td>
<td>6.8</td>
</tr>
<tr>
<td>#3</td>
<td>10.1</td>
<td>6.4</td>
<td>5.9</td>
<td>6.4</td>
</tr>
<tr>
<td>#4</td>
<td>8.7</td>
<td>6.1</td>
<td>6.4</td>
<td>6.0</td>
</tr>
<tr>
<td>#5</td>
<td>11.4</td>
<td>6.3</td>
<td>6.7</td>
<td>6.4</td>
</tr>
<tr>
<td>#6</td>
<td>10.0</td>
<td>5.7</td>
<td>6.2</td>
<td>5.7</td>
</tr>
<tr>
<td>#7</td>
<td>11.5</td>
<td>6.9</td>
<td>6.0</td>
<td>5.5</td>
</tr>
<tr>
<td>#8</td>
<td>10.7</td>
<td>7.7</td>
<td>6.3</td>
<td>7.7</td>
</tr>
<tr>
<td>#9</td>
<td>9.1</td>
<td>5.7</td>
<td>6.9</td>
<td>5.6</td>
</tr>
<tr>
<td>#10</td>
<td>11.9</td>
<td>5.6</td>
<td>6.0</td>
<td>5.5</td>
</tr>
<tr>
<td>Mean error</td>
<td>10.95</td>
<td>6.41</td>
<td>6.37</td>
<td>6.14</td>
</tr>
<tr>
<td>Std. deviation</td>
<td>1.81</td>
<td>0.68</td>
<td>0.38</td>
<td>0.67</td>
</tr>
</tbody>
</table>

Table 3
Testing error rate on Gauss50x.

<table>
<thead>
<tr>
<th>Selection</th>
<th>Supervised SVM with labeled data</th>
<th>Semi-supervised SVM with Self-training</th>
<th>Semi-supervised SVM with Help-training</th>
<th>Our algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>14.2</td>
<td>14.2</td>
<td>14.0</td>
<td>12.9</td>
</tr>
<tr>
<td>#2</td>
<td>13.9</td>
<td>13.4</td>
<td>13.5</td>
<td>13.2</td>
</tr>
<tr>
<td>#3</td>
<td>17.2</td>
<td>15.3</td>
<td>15.7</td>
<td>14.1</td>
</tr>
<tr>
<td>#4</td>
<td>15.8</td>
<td>17.6</td>
<td>14.0</td>
<td>14.3</td>
</tr>
<tr>
<td>#5</td>
<td>15.2</td>
<td>14.3</td>
<td>12.9</td>
<td>13.4</td>
</tr>
<tr>
<td>#6</td>
<td>18.6</td>
<td>13.0</td>
<td>16.0</td>
<td>12.2</td>
</tr>
<tr>
<td>#7</td>
<td>17.3</td>
<td>18.9</td>
<td>13.9</td>
<td>14.3</td>
</tr>
<tr>
<td>#8</td>
<td>15.2</td>
<td>14.7</td>
<td>12.9</td>
<td>12.7</td>
</tr>
<tr>
<td>#9</td>
<td>13.9</td>
<td>14.1</td>
<td>13.6</td>
<td>12.9</td>
</tr>
<tr>
<td>#10</td>
<td>15.6</td>
<td>14.4</td>
<td>13.4</td>
<td>13.2</td>
</tr>
<tr>
<td>Mean error</td>
<td>15.69</td>
<td>14.99</td>
<td>13.99</td>
<td>13.32</td>
</tr>
<tr>
<td>Std. deviation</td>
<td>1.58</td>
<td>1.86</td>
<td>1.06</td>
<td>0.71</td>
</tr>
</tbody>
</table>

Table 4
Testing error rate on IRIS.

<table>
<thead>
<tr>
<th>Selection</th>
<th>Supervised SVM with labeled data</th>
<th>Semi-supervised SVM with Self-training</th>
<th>Semi-supervised SVM with Help-training</th>
<th>Our algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>5.6</td>
<td>3.33</td>
<td>10</td>
<td>5.6</td>
</tr>
<tr>
<td>#2</td>
<td>15.6</td>
<td>26.7</td>
<td>7.8</td>
<td>5.6</td>
</tr>
<tr>
<td>#3</td>
<td>15.6</td>
<td>33.3</td>
<td>7.8</td>
<td>5.6</td>
</tr>
<tr>
<td>#4</td>
<td>52.2</td>
<td>14.4</td>
<td>38.9</td>
<td>14.4</td>
</tr>
<tr>
<td>#5</td>
<td>6.7</td>
<td>33.3</td>
<td>12.2</td>
<td>5.6</td>
</tr>
<tr>
<td>#6</td>
<td>27.8</td>
<td>6.7</td>
<td>6.7</td>
<td>5.6</td>
</tr>
<tr>
<td>#7</td>
<td>13.3</td>
<td>13.3</td>
<td>6.7</td>
<td>5.6</td>
</tr>
<tr>
<td>#8</td>
<td>15.6</td>
<td>13.3</td>
<td>5.6</td>
<td>13.3</td>
</tr>
<tr>
<td>#9</td>
<td>11.1</td>
<td>4.4</td>
<td>10</td>
<td>4.4</td>
</tr>
<tr>
<td>#10</td>
<td>14.4</td>
<td>33.3</td>
<td>7.8</td>
<td>5.6</td>
</tr>
<tr>
<td>Mean error</td>
<td>17.7</td>
<td>21.2</td>
<td>11.4</td>
<td>7.2</td>
</tr>
<tr>
<td>Std. deviation</td>
<td>13.5</td>
<td>11.9</td>
<td>9.9</td>
<td>3.5</td>
</tr>
</tbody>
</table>

Table 5
Testing error rate on Waveform.

<table>
<thead>
<tr>
<th>Selection</th>
<th>Supervised SVM with labeled data</th>
<th>Semi-supervised SVM with Self-training</th>
<th>Semi-supervised SVM with Help-training</th>
<th>Our algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>16.3</td>
<td>20.0</td>
<td>20.2</td>
<td>15.7</td>
</tr>
<tr>
<td>#2</td>
<td>15.85</td>
<td>15.5</td>
<td>24.05</td>
<td>14.75</td>
</tr>
<tr>
<td>#3</td>
<td>15.8</td>
<td>18.05</td>
<td>20.8</td>
<td>15.95</td>
</tr>
<tr>
<td>#4</td>
<td>27.25</td>
<td>20.3</td>
<td>20.8</td>
<td>17.15</td>
</tr>
<tr>
<td>#5</td>
<td>16.4</td>
<td>17.85</td>
<td>21.0</td>
<td>15.95</td>
</tr>
<tr>
<td>#6</td>
<td>14.9</td>
<td>17.3</td>
<td>18.4</td>
<td>15.6</td>
</tr>
<tr>
<td>#7</td>
<td>17.3</td>
<td>18.5</td>
<td>24.5</td>
<td>16.7</td>
</tr>
<tr>
<td>#8</td>
<td>16.3</td>
<td>17.65</td>
<td>22.8</td>
<td>14.75</td>
</tr>
<tr>
<td>#9</td>
<td>15.75</td>
<td>17.75</td>
<td>24.4</td>
<td>15.35</td>
</tr>
<tr>
<td>#10</td>
<td>15.9</td>
<td>18.6</td>
<td>20.2</td>
<td>15.6</td>
</tr>
<tr>
<td>Mean error</td>
<td>17.18</td>
<td>18.15</td>
<td>21.72</td>
<td>15.75</td>
</tr>
<tr>
<td>Std. deviation</td>
<td>3.59</td>
<td>1.36</td>
<td>2.09</td>
<td>0.76</td>
</tr>
</tbody>
</table>

From these tables we have two observations. First, as can be seen, all three semi-supervised classification methods give fairly good results. It can be concluded that unlabeled data can help to train a better classifier and improve the generalization ability of the classifier. Second, our algorithm obtains the best overall
results compared to Self-training and Help-training. Especially, our algorithm has an obvious improvement on the IRIS and Waveform datasets. And the standard deviation of testing errors of our algorithm is the lowest (as also shown in Fig. 4). This shows that our algorithm is more robust and depends less on the initial training data. We believe the reason may be that our algorithm uses unlabeled data to reveal the data space structure to compensate for the lack of prior information, even if the initial model assumption is not consistent with the real data space structure.

4.3. Error rate of labeling newly added data

We investigate the details of the training behavior with respect to the different methods in this section. We consider the error rate of labeling newly added data in the training process as a performance indicator of different methods on various datasets. Again we randomly selected 10% of the training set as labeled data and the remaining as unlabeled data each time and repeated the process 10 times. Figs. 5–8 show the performance of our algorithm and the other two semi-supervised classification methods. As shown in these figures, our algorithm outperforms both Self-training and Help-training.

4.4. The impact of the ratio of labeled data

In this section we will discuss the impact of the ratio of labeled data on the performance of the different methods. Experiments are designed to compare the performance of our algorithm to the other three methods while the ratio of labeled data in the training set increases from 10% to 100%. The results are shown in Figs. 9–12.

Not surprisingly, the test errors of all four methods decreases overall as the ratio of labeled data increases. However, the error rates obtained by our algorithm are generally lower than the other methods, and the performance of our algorithm is relatively less dependent on the ratio of labeled data. Sometimes we can even approach the Bayes error. For example, the Bayes error for Gauss50 dataset is 5% [19] and the result given by our algorithm is also 5% when the ratio of labeled data is 70%. Interestingly, we find that sometimes (e.g., Fig. 12) the performance of our
algorithm and the other two semi-supervised algorithms are worse than that of the supervised SVM that is only trained by labeled data. It may be because in those cases the labeled data can represent the whole data space fairly well, while the unlabeled data actually increase the probability of overfitting, and this shows, unfortunately, that unlabeled data are not always useful for improving the performance of a classifier.

4.5. The impact of parameter $e_1$

In this section, we analyze the behavior of our algorithm under the change of parameters. The parameter $e_1$ influences the number of selected data points using SSFCM, while parameter $e_2$ controls how many of these points affect the update of the classifier. The difference between our algorithm and the other two semi-supervised methods is that we select some data points with parameter $e_1$ using SSFCM prior to train the classifier. Here we only discuss the influence of parameter $e_1$ on each dataset. For each dataset, we choose 10% as labeled data in the training set and the rest as unlabeled and repeat for 10 times. The results are

---

Fig. 8. Error rate of labeling newly added data using different methods with respect to each selection on Waveform dataset.

Fig. 9. Test error of the different methods with respect to the ratio of labeled data on Gauss50 dataset.

Fig. 10. Test error of the different methods with respect to the ratio of labeled data on Gauss50x dataset.

Fig. 11. Test error of the different methods with respect to the ratio of labeled data on IRIS dataset.

Fig. 12. Test error of the different methods with respect to the ratio of labeled data on Waveform dataset.
shown in Figs. 13–16. From these figures, we have the following observations. First, when parameter $e_1$ is less than or equal to $1/c$, test errors do not change and our algorithm fails to help to train a better classifier. This is because our algorithm will select all unlabeled data in each iteration, not the really informative part. In this case, the algorithm is essentially Self-training. Second, when parameter $e_1$ is greater than $1/c$ and less than 1, our algorithm has an auxiliary function and achieves a good performance. And we also find that test errors change very slightly as parameter $e_1$ varies. So our algorithm is quite robust with respect to parameter $e_1$. Third, when parameter $e_1$ is equal to 1, test errors slightly increase compared to the previous case (i.e., $1/c < e_1 < 1$), because unlabeled data are directly classified by the classifier trained only by labeled data and our algorithm is essentially the supervised SVM.

4.6. Comparisons of our algorithm to other semi-supervised SVMs

Finally, we compare our algorithm to some of the other semi-supervised SVM methods [10,32]: S$^3$VM-light [33] and S$^3$VM-CCCP [34]. The results are presented in Table 6. Our algorithm achieves performance comparable to that of S$^3$VM-light and S$^3$VM-CCCP in general. We note that our algorithm has slightly higher error rates on the IRIS dataset, which is most probably because this dataset is very small and the data space structure is quite clearly captured by the small amount of labeled data, while the selection process of our algorithm is not really useful in helping train a better classifier.

---

**Table 6**

Comparison between our algorithm and other semi-supervised SVM methods. In this table we report the error rate on each dataset.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>S$^3$VM-light (%)</th>
<th>S$^3$VM-CCCP (%)</th>
<th>Our algorithm (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss50</td>
<td>7.72 ± 1.86</td>
<td>7.74 ± 2.06</td>
<td>6.14 ± 0.67</td>
</tr>
<tr>
<td>Gauss50x</td>
<td>13.64 ± 0.8</td>
<td>13.47 ± 0.97</td>
<td>13.32 ± 0.71</td>
</tr>
<tr>
<td>IRIS</td>
<td>4.9 ± 0.97</td>
<td>6.01 ± 1.33</td>
<td>7.2 ± 3.5</td>
</tr>
<tr>
<td>Waveform</td>
<td>15.75 ± 0.83</td>
<td>16.02 ± 1.06</td>
<td>15.79 ± 0.76</td>
</tr>
</tbody>
</table>
5. Conclusion

In this paper we introduced a framework where semi-supervised clustering is integrated into Self-training as a helping strategy. This work is motivated by the observation that unlabeled data contain some inner data space structure information and the underlying information can be revealed by clustering and help to train a better classifier. In particular, we considered SSFCM and SVMs as the choices of clustering and classification, respectively. We carried out a series of experiments to analyze our algorithm and some other semi-supervised learning algorithms. The results show that our algorithm performed better than Self-training and Help-training in various datasets. In comparison to S$^2$VM-light and S$^2$VM-CCCP, our algorithm also achieved comparable, if not better, performance.

In the future, we will investigate other clustering algorithms, e.g., manifold learning, to improve the performance of semi-supervised learning. From Section 4.4, it has been found that the performance of semi-supervised classification methods is not always better than that of supervised methods [35]. We will explore the safer situations to use unlabeled data and study safer semi-supervised classification approaches.

Acknowledgments

We would like to express our sincere thanks to Mathias M. Adankon and Mohamed Cheriet for offering the source codes of Help-training. The work is supported by the National Natural Science Foundation of China under Grants 61072143, 61105014, and 61170093.

References

Rui Huang, corresponding author, received the BS degree in computer science from Peking University in 1999. After earning the ME degree in intelligent systems from Chinese Academy of Sciences in 2002, he joined Rutgers University, where he received the PhD degree in computer science in 2008 and worked as a postdoctoral research associate. He is currently a faculty member in Huazhong University of Science and Technology. His research interests are deformable models, graphical models, image and video based object segmentation and recognition, shape analysis, etc.

Xiaojun Tong received his MS degree from Department of Mathematics, Northwest University in 1993 and PhD degree from the Huazhong University of Science and Technology in 2002. He has published more than 26 papers in journals and conference proceedings. His current research interests include pattern recognition, image processing and machine learning.

Zhiping Dan received the BS degree from Mathematics Department, Wuhan University, in 1997 and the MS degree from Mathematics Department, Huazhong University of Science and Technology, China, in 2005. He is presently a PhD Candidate with the Institute for Pattern Recognition and Artificial Intelligence, Huazhong University of Science and Technology. He is also an associate professor at the College of Computer and Information Technology, China Three Gorges University. His research interests are in the areas of pattern recognition, computer vision, and machine learning.