Appl Intell (2018) 48:2373–2392
https://doi.org/10.1007/s10489-017-1076-z

Global and local learning from positive and unlabeled examples

Ting Ke[1] · Ling Jing[2] · Hui Lv[1] · Lidong Zhang[1] · Yaping Hu[1]

Published online: 11 November 2017

Abstract In common binary classification scenarios, learning algorithms assume the presence of both positive and negative examples. Unfortunately, in many practical areas, only limited labeled positive examples and large amounts of unlabeled examples are available, but there are no negative examples. In such cases, the algorithm that only exploits positive and unlabeled examples is needed. Such learning is termed as positive and unlabeled (PU) learning. In this paper, a novel classifier called global and local learning classifier (GLLC) for PU learning is proposed. The advantages of GLLC are as follows: (1) both intrinsic geometric structure and accurate positive information of PU data are exploited from global learning. (2) The smoothness and manifold of data are reflected sufficiently from local learning. (3) The algorithm of GLLC has faster training speed because the linear equations are solved. (4) The experiments on both synthetic and real datasets verify the above opinions and show that the classification result of GLLC is much better than those popular methods, such as LUHC, Pulce, BSVM, NB and so on.

Keywords Positive and unlabeled learning · Least squares support vector machine · Global · Local · Regularization

✉ Ting Ke
keting@tust.edu.cn; kk.ting@163.com

[1] Department of Mathematics, College of Science, Tianjin University of Science & Technology, 300457, Tianjin, People's Republic of China

[2] College of Science, China Agricultural University, Beijing 100083, People's Republic of China

1 Introduction

Data mining is a very important and widely used technology in many diverse domains. Classification is an important data mining technology. In common binary classification tasks, the presence of both positive and negative examples in training data is needed to build an efficient classifier. Nevertheless, the acquisition of class labels is both costly and difficult in some real-world fields. In the worst case, it is impossible to extract class labels. In such cases, the algorithm for classification that only exploits a small portion of positive and large amount of unlabeled examples is needed. Such learning is termed as positive and unlabeled (PU) learning. Figure 1 shows the data distribution for PU learning in two-dimensional space. From the Fig. 1, we observe that each example has two features, $[x]_1$ and $[x]_2$. "Red +" represents positive examples and "blue circle" represents unlabeled examples, where "blue $\oplus$" and "blue $\ominus$" describe positive and negative examples in unlabeled ones respectively. The objective of PU learning is to predict the label of these unlabeled examples (blue circle) and any new example point. In this paper, our main work is classification for PU learning.

PU learning is a hot topic in the literatures of data mining. For example in practice, users may mark their favorite Web pages (positive examples), but they are usually unwilling to mark boring pages (negative examples). Recently, more and more applications for PU learning appear, such as bioinformatics classification [1], time series classification [2], data stream classification [3, 4], opinion mining [5, 6] and so on.

For PU learning, several approaches have been proposed. A theoretical study of probably approximately correct (PAC) learning from positive and unlabeled examples was first done in [7]. The study concentrated on the

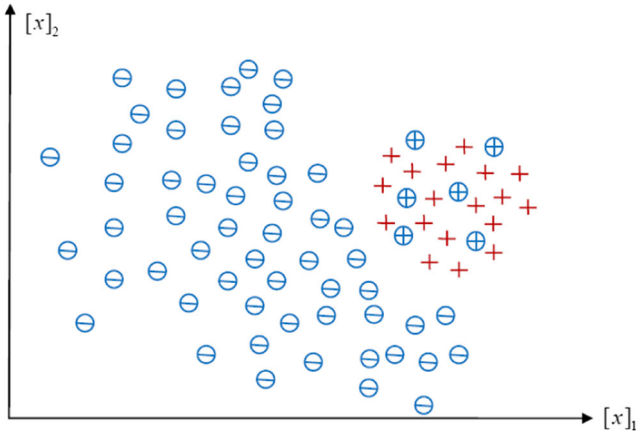

Fig. 1 Two-dimensional data distribution for PU learning

computational complexity of learning and showed that function classes learnable under the statistical queries model. Recently, learning from positive examples was also studied theoretically in [8] within a Bayesian framework where the distribution of functions and examples were assumed known. In [9], it was shown that if the sample size was large enough, minimizing the number of unlabeled examples classified as positive while constraining the positive examples to be correctly classified would give a good classifier. With these theories, various approaches have been suggested in the literatures to solve PU learning. The main difference for these methods was how to use unlabeled examples, and there were four types of methods.

The first type is the two-step strategy, which selects possible negative or positive examples from unlabeled examples, and then builds classifiers using positive examples and negative examples. The popular techniques for extracting possible negative or positive examples included spy, Rocchio and 1-DNF [10]. After extracting possible negative or positive examples, standard machine learning methods such as Naïve Bayes and SVM [11] are used to train classifiers. At present, the two-step strategy is a widely used method, such as S-EM [12], ROC-SVM [13], 1-DNF, PNLH [14], LCLC [15], Pulce [16, 17] etc. Pulce took advantage of the intrinsic relationships between attribute values and exceeded the independence assumption made by Naïve Bayes. In fact, Pulce leveraged the statistical properties of the data to learn a distance metric during the classification task.

The second type of method is related to one-class classification, which estimates the distribution of the positive class only from positive examples, such as one-class support vector machine [18, 19]. In fact, it has been proved that these methods are effective only for the case where the number of positive examples is large enough to capture the characteristics of the positive class, and their performance would be rather poor when this number is very small [20].

The third type of method is the probability estimation approach [21, 22]. In [21], it took each unlabeled example as both positive and negative example with weights pre-computed by an additional classifier for protein record identification. Luigi Cerulo et al made use of the approach in [23] to reconstruct gene regulatory networks without negative examples.

The fourth type of methods is a one-step method. It includes BSVM [9], WL [24, 25] and LUHC [26] which is most related with our work. BSVM was built by giving appropriate weights to the positive examples and unlabeled examples which were regarded as negative examples with noise, respectively. Experimental results indicated that the performance was better than most of two-step strategies. WL used Logistic Regression technique after weighting the negative class. LUHL proposed a Laplacian unit-hyperplane classifier which adding a manifold regularizer to make the predicted labels and the initial labels sufficiently close on the labeled points.

For PU learning, we mainly propose a novel classifier based on global and local learning (GLLC). From the global perceptive, our classifier constructs a biased least square support vector machine (BLSSVM) with all positive and unlabeled examples (can be taken as negative examples with noise) rather than only support vectors like BSVM. The obvious advantage of BLSSVM is that a little noise in a large amount of negative examples will have little effect on the building of the final classifier, whereas the same noise in support vectors will produce serious deviation on the construction of the final classifier, such as BSVM. In addition, BLSSVM has a very small fluctuation and excellent performance over a wide ratio of positive examples in unlabeled examples. This is because the number of negative examples is far more than positive examples in unlabeled examples and the distribution of negative class changes little over a wide ratio of positive examples in unlabeled examples. This leads to the final classifier being more stable and accurate whatever the ratio of positive examples in unlabeled examples is. From the local perspective, a smooth regularization term indicates the two data points should belong to the same class if they are close in the intrinsic geometry. This smooth term not only reflects geometrical properties of the training examples but also makes up the insufficient training of the global learning. In summary, combining global learning and local learning will produce a novel classifier which is called GLLC for short. To sum up, GLLC makes best use of the advantages and bypasses the disadvantages. Finally the numerical experiments on both synthetic and several real world datasets are presented to show the effectiveness of GLLC. The preliminary results show that our GLLC is less sensitive to the proportion of labeled examples, and superior to BSVM, LUHC and EM on both the ability of classification and the computational efficiency.

The rest of this paper is organized as follows. Some previous works are introduced in Section 2. The proposed GLLC approach is presented in Section 3. Then in Section 4, experiments on both synthetic and real datasets are reported. Finally, we conclude this paper in Section 5.

2 Related works

Before we go into the detail of related works, let us give the definition of PU learning firstly. Throughout this paper, we suppose that the training dataset is represented as $T = \{(x_i, y_i), i = 1, \cdots, p, x_j, j = p+1, \cdots, p+u\}$ where p and u are the number of positive and unlabeled examples respectively, $x_i \in \mathbb{R}^n$ is a training example input and $y_i = 1$ is the positive class label of x_i, $(i = 1, \cdots, p)$, the label of x_j, $j = p+1, \cdots, p+u$ is unknown, but it equals $+1$ or -1. The objective of PU learning is to find a real function $f(x)$ in $\mathbb{R}^n$ such that the output value of y for any x can be predicted by the sign function of $f(x)$ as

$$sign(f(x)). \tag{1}$$

2.1 BSVM

One of the most popular methods for PU learning was proposed based on SVM technique in [9] which was called BSVM. BSVM took unlabeled examples as negative examples with noise, namely $y_i = -1$, $(i = p+1, \cdots, p+u)$, and then constructed an SVM classifier by giving a larger penalty parameter to weight the positive examples errors and a smaller penalty parameter to weight the negative examples errors. Then the optimization problem of BSVM can be formulated as

$$\begin{aligned} &\min_{(w,b,\xi)} \frac{1}{2} \| w \|^2 + C_p \sum_{i=1}^{p} \xi_i + C_n \sum_{i=p+1}^{p+u} \xi_i \\ &s.t. y_i (< w, \varphi(x_i) > + b) \geq 1 - \xi_i i = 1, 2, ..., p+u \\ &\xi_i \geq 0 i = 1, 2, ..., p+u \end{aligned} \tag{2}$$

where $\xi = (\xi_1, \xi_2, ..., \xi_{p+u})^T$ is penalizing variable vector. C_p and C_n represent the penalty parameters of misclassification for positive and unlabeled examples respectively. Similar to classical SVM, function $\varphi(\cdot)$ maps the input space into a higher dimensional space when the dataset is nonlinearly separable. And then searching for a hyperplane in the feature spaces

$$f(x) = < w, \varphi(x) > + b = 0 \tag{3}$$

to predict the class label of any example x with (1).

Although BSVM [9] is a state-of-the-art learning algorithm for PU learning [27], only its support vectors play a role in classification, and performance is not satisfactory when the number of the labeled positive examples is small.

2.2 Least squares SVM

The least squares support vector machine (LSSVM) is a promising classification technique proposed firstly by Suykens et al. [28]. LSSVM seeks an optimal separating hyper-plane $f(x) = < w, \varphi(x) > + b = 0, w \in \mathbb{R}^n, b \in \mathbb{R}$

That maximizes the margin between two classes based on given training examples $\{x_i, y_i\}, i = 1, \cdots, l$, where x_i is a vector in the input space $\mathbb{R}^n$ and y_i denotes the class label taking a value of $+1$ or -1. $\varphi(\cdot)$ is a nonlinear function which is used to map the input space into a higher dimensional space. Therefore, the maximum-margin classifier is obtained by solving

$$\begin{aligned} &\min_{(w,b,\xi)} \tfrac{1}{2} \| w \|^2 + \tfrac{C}{2} \sum_{i=1}^{l} \xi_i^2 \\ &s.t. y_i (< w, \varphi(x_i) > + b) = 1 - \xi_i i = 1, 2, ..., l \end{aligned} \tag{4}$$

The geometric interpretation of the above problem (4) with $x \in \mathbb{R}^2$ for linearly separable case is shown in Fig. 2., where minimizing $\frac{1}{2} \| w \|^2$ realizes the maximal margin between the straight lines $< w, \varphi(x) > + b = 1$ *and* $< w, \varphi(x) > + b = -1$, minimizing $\sum_{i=1}^{l} \xi_i^2$ makes the two straight lines proximal to all inputs of positive ("+") and negative ("○") examples respectively.

It's worth mentioning that the superiority of LSSVM is simpler and faster than SVM. This is because LSSVM needs to solve a quadratic programming with only equality constraints, or equivalently a linear system of equations, but SVM needs to solve a quadratic programming with inequality constraints.

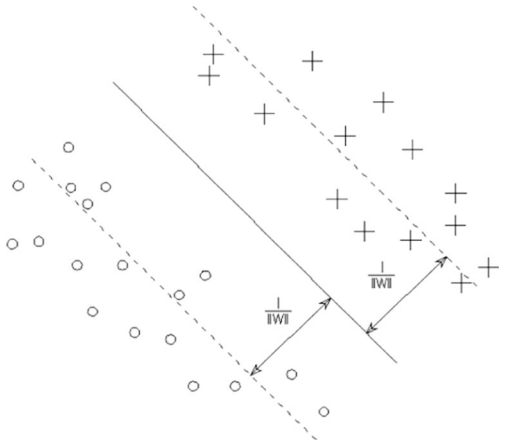

Fig. 2 An interpretation of two-dimensional least squares SVM

3 Global and local learning

3.1 Global learning

Consider the PU learning described in the beginning of Section 2. To overcome the shortcoming of BSVM, we first construct a biased least squares SVM (BLSSVM) classifier by giving two different penalty parameters to weight the misclassification errors of positive and unlabeled examples respectively, where unlabeled examples can be regarded as negative examples with noise, i.e., $y_i = -1, i = p+1, \cdots, p+u$. Thus, we seek the decision function

$$f(x) =< w, x > +b = 0, w \in \mathbb{R}^n, b \in \mathbb{R}$$

so that the classification hyper-plane separates both positive examples and unlabeled examples with the maximum margin, resulting in the primal optimization problem

$$\begin{aligned} &\min_{(w,b,\xi)} \tfrac{1}{2} \| w \|^2 + \tfrac{C_p}{2} \sum_{i=1}^{p} \xi_i^2 + \tfrac{C_n}{2} \sum_{i=p+1}^{p+u} \xi_i^2 \\ &s.t. y_i(< w, x_i > +b) = 1 - \xi_i i = 1, 2, ..., p+u \end{aligned} \tag{5}$$

Both penalizing term and equality constraints reflect that all positive examples approximate to straight line $< w, x > +b = 1$ and unlabeled examples approximate to $< w, x > +b = -1$ respectively. However, the degree of approximation for positive and unlabeled examples is different, which is controlled by the parameters C_p and C_n. Obviously, C_p is larger than C_n because we pay more attention to classify positive examples for PU learning. More concisely, Let

$$\begin{aligned} P &= \left(x_1, x_2, \cdots, x_p\right)^T, \\ U &= \left(x_{p+1}, x_{p+2}, \cdots, x_{p+u}\right)^T, \\ X &= \left(P^T, U^T\right)^T, \\ Y &= diag(y_1, y_2, \cdots, y_{p+u}), \\ C &= diag(\underbrace{C_p, \cdots, C_p}_{p}, \underbrace{C_n, \cdots, C_n}_{u}), \\ \xi &= (\xi_1, \xi_2, \cdots, \xi_{p+u})^T \end{aligned} \tag{6}$$

Thus, the optimization model (5) can rewrite as matrix format

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2} w^T w + \frac{1}{2} \xi^T C \xi \\ &s.t. Y\,(Xw + eb) + \xi = e \end{aligned} \tag{7}$$

where e is an appropriate vector of ones.

Clearly, all training data takes part in classification. We can call the BLSSVM classifier which is obtained by (7) as global learning classifier. Compared with traditional methods, such as BSVM, BLSSVM has some obvious merits. First, BLSSVM can reflect the class labels of all examples more sufficiently and accurately than BSVM. This is because that only support vectors determine the final classifier for BSVM, which can be deemed as local learning. Whereas the whole examples are used to construct the final classifier for BLSSVM, which can be regarded as global learning. Obviously, negative support vectors may contain some positive examples and produce some effects on the construction of BSVM. Contrarily, if all examples take part in the building of classifier, compared with most correct negative examples, those false negative examples are not important for the final classifier construction. Second, BLSSVM is a more stable classifier than BSVM. Namely, the performance of BLSSVM is less sensitive than that of BSVM over a wide ratio of positive examples in unlabeled examples. This is because that the number of negative examples is far more than positive examples in unlabeled examples and the distribution of negative class changes little over a wide ratio of positive examples in unlabeled examples. All in all, Global learning for classification is more accurate and effective for PU learning.

3.2 Local learning

In the semi-supervised learning scenario, a common hypothesis [29–32] is that two data points are more likely to belong to the same class if they are close in the intrinsic geometry. In the other words, the difference between function $f(x_i)$ and $f(x_j)$ should be smaller if the distance between x_i and x_j is shorter. In this paper, we take this common hypothesis into PU learning, minimize the following regularization term

$$\| f \|_M^2 = \frac{1}{(p+u)^2} \sum_{i,j=1}^{p+u} w_{ij} (f(x_i) - f(x_j))^2 \tag{8}$$

where w_{ij} is similarity weight between x_i and x_j. It can be calculated by Gaussian functions:

$$w_{ij} = \begin{cases} \exp(-\left\| x_i - x_j \right\|^2 \Big/ 2\sigma) & if x_i, x_j \; are \; neighbor \\ 0 & otherwise \end{cases} \tag{9}$$

where σ is a bandwidth hyper-parameter. It controls the decay rate.

Obviously, if the x_i is near to x_j, implying that they have a high similarity, the difference between $f(x_i)$ and $f(x_j)$ should be severely punished since w_{ij} is large, resulting in that $\left| f(x_i) - f(x_j) \right|$ is small. Note $L = D - W$, W is the graph weight matrix where (i,j)-th entry is w_{ij} and D is a diagonal degree matrix with $D_{ii} = \sum_{j=1}^{l+u} w_{ij}$. The regularization item in (8) can be rewritten as matrix format

$$\| f \|_M^2 = f^T L f \tag{10}$$

where $f = \left(f(x_1), f(x_2), \cdots f(x_{p+u})\right)^T$. Then, the final classification is performed by (1).

It is easily observed that, regularization term in (10) is local learning, which just considers the class label of neighborhood data points. Such local learning reflects intrinsic geometric information for classification.

3.3 Global and local learning

Now let us consider PU learning scenario, combining global learning in (7) and local learning in (10), we obtain the optimization problem

$$\begin{aligned} &\min_{w,b,\xi} \frac{\lambda}{2} w^T w + \frac{1}{2}\xi^T C\xi + f^T Lf \\ &s.t. Y\left(Xw + \mathrm{e}b\right) + \xi = e \end{aligned} \tag{11}$$

Note that λ is positive parameter. Let

$$f = \left(f(x_1), f(x_2), \cdots f(x_{p+u})\right)^T = Xw + eb \tag{12}$$

Then, the above problem (11) leads to our primal problem

$$\begin{aligned} &\min_{w,b,\xi} \tfrac{\lambda}{2} w^T w + \tfrac{1}{2}\xi^T C\xi + \tfrac{1}{2}(Xw + eb)^T L(Xw + eb) \\ &s.t. Y\left(Xw + \mathrm{e}b\right) + \xi = e \end{aligned} \tag{13}$$

Clearly, the minimizing problem (13) is a quadratic programming with only equality constraints. Introducing the Lagrange multiplier $\alpha = (\alpha_1, \cdots, \alpha_{p+u})^T$ for optimization problem (13), we obtain

$$\begin{aligned} L\left(w, b, \xi, \alpha\right) = &\frac{\lambda}{2} w^T w + \frac{1}{2}\xi^{\mathrm{T}} C\xi \\ &+ \frac{1}{2}(Xw + eb)^T L(Xw + eb) \\ &- \alpha^T Y\left(Xw + eb\right) - \alpha^T \xi + \alpha^T e \end{aligned} \tag{14}$$

Based on the Karush-Kuhn-Tucker (KKT) necessary and sufficient optimality conditions, taking the partial derivatives of (14) with respect to w, b, q, α and equating them to zero, we obtain the following optimal conditions

$$\frac{\partial L}{\partial w} = \lambda w + X^T L(Xw + eb) - X^T Y\alpha = 0 \tag{15}$$

$$\frac{\partial L}{\partial b} = \mathrm{e}^T L(Xw + eb) - \mathrm{e}^T Y\alpha = 0 \tag{16}$$

$$\frac{\partial L}{\partial \xi} = c\xi - \alpha = 0 \tag{17}$$

$$\frac{\partial L}{\partial \alpha} = Y\left(Xw + \mathrm{e}b\right) + \xi - e = 0 \tag{18}$$

Substituting (17) into (18), we can solve the resulting equations for w, α and b

$$\begin{cases} (\lambda I + X^T LX)w + X^T Leb - X^T Y\alpha = 0 \\ \mathrm{e}^T LXw + \mathrm{e}^T Leb - \mathrm{e}^T Y\alpha = 0 \\ YXw + Y\mathrm{e}b + C^{-1}\alpha = e \end{cases} \tag{19–21}$$

where I is an identity matrix. The matrix format is

$$\begin{pmatrix} \lambda I + X^T LX & X^T Le & -X^T Y \\ \mathrm{e}^T LX & \mathrm{e}^T Le & \mathrm{e}^T Y \\ YX & Y\mathrm{e} & C^{-1} \end{pmatrix} \begin{pmatrix} w \\ b \\ \alpha \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ e \end{pmatrix} \tag{22}$$

It is worth mentioning that the dimension of variables is too high to solve in short time. Therefore, we can solve the value of w^* from the formulation (19) when the matrix $(\lambda I + X^T LX)$ is invertible. Namely,

$$w^* = (\lambda I + X^T LX)^{-1}(X^T Y\alpha - X^T Leb) \tag{23}$$

Let

$$A = (\lambda I + X^T LX)^{-1} \tag{24}$$

Substituting w^* into equations (20–21), we can obtain the solution of $\alpha*$ and b^* from the following linear equations

$$\begin{pmatrix} \mathrm{e}^T LXAX^T Y - \mathrm{e}^T Y & \mathrm{e}^T Le - \mathrm{e}^T LXAX^T Le \\ YXAX^T Y + C^{-1} & Y\mathrm{e} - YXAX^T Le \end{pmatrix} \begin{pmatrix} \alpha \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ e \end{pmatrix} \tag{25}$$

Once $\alpha*$ and b^* are found, a new example x can be classified as positive class or negative class by the following formulation

$$\begin{aligned} y &= sign(f(x)) \\ &= sign(\sum_{j=1}^{p+u} \alpha_j y_j < x, x_j > + b) \end{aligned} \tag{26}$$

Now, the key problem is how to guarantee the invertibility of $(\lambda I + X^T LX)$ and what are the merits of $(\lambda I + X^T LX)$? The following theorems can answer these questions.

Theorem 1 *$(\lambda I + X^T LX)$ is symmetric and positive definite.*

Proof L is a symmetric and semi-positive definite matrix, then the following equations are satisfied

$$\forall z \in \mathrm{R}^n, z^T X^T LXz = (Xz)^T L(Xz) \geq 0 \tag{27}$$

$$(X^T LX)^T = X^T L^T X = X^T LX \tag{28}$$

Obviously, $X^T LX$ is a symmetric and semi-positive definite matrix. Noting that $\lambda > 0$, we can derive λI is symmetric and positive definite. Therefore $(\lambda I + X^T LX)$ is symmetric and positive definite. □

Theorem 2 *$A = (\lambda I + X^T LX)^{-1}$ is also a positive definite matrix and there exists the matrix B, such that*

$$A = BB^T \tag{29}$$

Proof Since $(\lambda I + X^T L X)$ is a positive definite matrix, its invertible matrix is also a positive definite matrix. It can be decomposed into

$$A = T\Lambda^{\frac{1}{2}}\Lambda^{\frac{1}{2}}T^T = (T\Lambda^{\frac{1}{2}})(T\Lambda^{\frac{1}{2}})^T = BB^T \tag{30}$$

where Λ is a diagnose matrix, the diagnosed entry is eigenvalue, T is composed by eigenvectors. □

Through the previous theorems, substituting A into BB^T, formula (20) can be rewritten as

$$\begin{pmatrix} \mathrm{e}^T L(XB)(XB)^T Y - \mathrm{e}^T Y & \mathrm{e}^T Le - \mathrm{e}^T L(XB)(XB)^T Le \\ Y(XB)(XB)^T Y + C^{-1} & Y\mathrm{e} - Y(XB)(XB)^T Le \end{pmatrix} \begin{pmatrix} \alpha \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ e \end{pmatrix} \tag{31}$$

It is natural to find that coefficient matrix in (26) which is partitioned into four blocks always contains inner product operation, just like $(XB)(XB)^T$. As we all known, inner product is a kind of kernel or some distance measure, such as linear kernels, RBF kernels, Euclidean distance, Mahalanobis distance and so on. In addition, we find the object of inner product is XB, and it can be regarded as a linear transformation for training data X. In fact, this kind of linear transformation plays an important role in feature selection and dimension decrease. In conclusion, $(XB)(XB)^T$ is a novel metric measure after linear mapping of X. These new insights prompt us to extend the above linear separable case to nonlinear separable case. For nonlinearly separable case, we also introduce a nonlinear function $\Phi(XB)$ mapping the input feature space to the higher dimensional space. Naturally, inner product $\Phi(XB)\Phi(XB)^T$ is deemed as a kernel function (matrix). Note $K = \Phi(XB)\Phi(XB)^T$. Therefore, for nonlinear separable case, substituting $(XB)(XB)^T$ for K will receive the following linear equations

$$\begin{pmatrix} \mathrm{e}^T LKY - \mathrm{e}^T Y & \mathrm{e}^T Le - \mathrm{e}^T LKLe \\ YKY + C^{-1} & Y\mathrm{e} - YKLe \end{pmatrix} \begin{pmatrix} \alpha \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ e \end{pmatrix} \tag{32}$$

Based on (27), a nonlinear classifier appears and classifies any new example x as positive class or negative class by the following decision function

$$\begin{aligned} y &= sign(f(x)) \\ &= sign(\sum_{j=1}^{p+u} \alpha_j^* y_j K(Bx, Bx_j) + b^*) \end{aligned} \tag{33}$$

Right now, we establish the complete learning framework. Noting that our classifier is constructed based on global and local learning, so it is called GLLC for short. From a global learning perspective, GLLC not only has a strong robustness but also has a very small fluctuation over a wide ratio of positive examples in unlabeled examples. From a local learning perspective, a smooth regularization term will be minimized to make the two neighborhood data points are more likely belong to the same class if they are close in the intrinsic geometry. This smooth term not only reflects geometrical properties of the training examples but also alleviates the insufficient training of the global learning. Third, the time complexity of GLLC is lower than that of BSVM, where GLLC only needs to solve linear equations and BSVM is a quadratic programming. The implementation of the algorithm for GLLC is shown in Table 1.

4 Experiments

To evaluate the performance of GLLC, we perform experiments on three sets of artificial datasets, six UCI benchmark datasets [33] and one USPS handwritten image dataset in this paper. More concretely, we compare GLLC with the following state-of-the-art algorithms:

1. BSVM is a classical support vector machine. It uses positive examples and unlabeled examples which can be negative examples with noise to build an SVM classifier.
2. Pulce takes into account data dependencies and learns a distance model from attribute relationships to train a k-NN-like classifier.
3. LUHC determines a decision unit-hyperplane by solving a quadratic programming problem.
4. NB is another PU learning algorithm, which takes 'selected completely at random' assumption and estimates the prior probability of positive, then learn classifier model.
5. S-EM uses spy technique to identify reliable negative examples from the unlabeled examples, and uses the Expectation Maximization (EM) algorithm to build a NB classifier.

Table 1 The solving algorithm for GLLC

- **Input:** A group of parameters λ, C_p, C_n, σ, a threshold ε=10^{-3}, Training dataset X, class labels of training data Y; Test dataset Q;

- Solve the system of equations in (27) :

If $\left|\det\begin{pmatrix} \mathrm{e}^T LKY - \mathrm{e}^T Y & \mathrm{e}^T Le - \mathrm{e}^T LKLe \\ YKY + C^{-1} & Y\mathrm{e} - YKLe \end{pmatrix}\right| > \varepsilon$

Variable vector $\begin{pmatrix} \alpha^* \\ b^* \end{pmatrix} = \begin{pmatrix} \mathrm{e}^T LKY - \mathrm{e}^T Y & \mathrm{e}^T Le - \mathrm{e}^T LKLe \\ YKY + C^{-1} & Y\mathrm{e} - YKLe \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ e \end{pmatrix}$;

Predict the class labels of test dataset using (28);

Else

The system of equations in (27) has no solutions;

End

- **Output:** decision function and the class labels of test data.

All the classifiers are implemented in MATLAB 2014a environment on a PC with Intel Core(TM) core i3 CPU (2.4 GHz) with 2 GB RAMS. In both classification datasets, we use LIBSVM [34] to build a classifier for BSVM, LPU package [35] to implement system for NB and S-EM.

4.1 Performance metric

We use the popular F value on the positive class as the evaluation measure. F value takes into account of both recall and precision

$$F = \frac{2pr}{p+r} \tag{34}$$

Where

$$p = \frac{TP}{TP+FP}$$

And

$$r = \frac{TP}{TP+FN} \quad .$$

TP, *TN*, *FP* and *FN* are the number of true positive, true negative, false positive and false negative, respectively. A high precision ensures that the identified positives are predominantly true positives, and a high recall ensures that most of the positives are identified. F value captures the average effect of both precision and recall, and is therefore suitable for our purpose. In addition, the accuracy of classification is also showed in this paper to evaluate the performance of GLLC.

Table 2 Parameter selection algorithm for GLLC

Set ranges for parameters
$\lambda \in \{\lambda_{\min}, \cdots, \lambda_{\max}\}, C_n \in \{C_{\min}, \cdots, C_{\max}\},$
$C_p = 2C_n, \sigma \in \{\sigma_{\min}, \cdots, \sigma_{\max}\}, N_{fold}$;
Partition the training dataset into N_{fold} partitions: $X_i = (P_i, U_i)$,
$i = 1, \cdots, N_{fold}$;
Perform N_{fold} fold cross validation method to optimize parameters:
For each group of parameters $\lambda, C_p, C_n, \sigma$,
For $i = 1 : N_{fold}$
Substitute $X_i = (P_i, U_i)$ into the system of (27) ;
Solve the system of (27), and compute the F value $\hat{F}_i$ using (30) ;
End
Compute the average F value $\overline{F} = \left(\sum_{i=1}^{N_{fold}} \hat{F}_i\right) / N_{fold}$;
End
Find the optimal parameters vector $\lambda^*, C_p^*, C_n^*, \sigma^*$ that maximizes the average F value using the predictions.

4.2 The selection of the parameters

For our GLLC, hyper-parameters are set by cross validation from some grids. Unfortunately, F value cannot be computed on the validation set during the training process because there is no negative example. An approximate computing method [9] is used to evaluate the performance by

$$\hat{F} = \frac{r_p^2}{P(f(x)=1)} \tag{35}$$

where x is the random variable representing the input vector, $P(f(x) = 1)$ is the probability of an input example x classified as positive example, r_p is the recall for positive set P in the validation set. Therefore, parameter selection algorithm for our GLLC is shown in Table 2. Concretely, hyper-parameters are set by five-fold cross validation from some grids introduced in the following.

- GLLC. RBF kernel parameter σ, C_n and λ are chosen from $\{2^{-6}, 2^{-5}, \cdots, 2^6\}$ and C_p is searched from $2C_n$,
- LUHC RBF kernel parameter σ and λ are also searched from $\{2^{-6}, 2^{-5}, \cdots, 2^6\}$ and ν is selected from $\{0.1, 0.2, \cdots, 0.9\}$
- BSVM. RBF kernel parameter σ, C_n are also chosen from the set $\{2^{-6}, 2^{-5}, \cdots, 2^6\}$ and C_p is searched from $2C_n$
- Pulce. Parameter k is varied over the set of values {1,3,7}.

The final decision function is obtained by the optimal parameters by selecting the best average F value in (30).

4.3 Experiment results

4.3.1 Artificial datasets

Consider three types of two-dimensional synthetic 'two half moons' datasets with a few randomly labeled positive examples. These three types of datasets contain 200, 400 and 600 examples respectively, where red star represents positive examples and black dot represent negative examples. In order to demonstrate the classification performance for PU learning, 10%, 20% and 30% proposition of positive examples are labeled randomly and radius data points are taken as unlabeled examples for these three types of datasets, which are marked blue triangle in Figs. 3, 4, 5, 6, 7, 8, 9, 10 and 11. In order to reflect the advantages of the manifold regularization in our GLLC algorithm, we compare the GLLC with LUHC and BSVM only in this experiment. The RBF kernel is used by all these three methods. In Figs. 3–11a, red stars represent positive unlabeled examples; black solid points represent negative unlabeled examples; All these examples are unlabeled examples. Blue triangles are positive examples. Figures 3–11b, c and d show the classification results

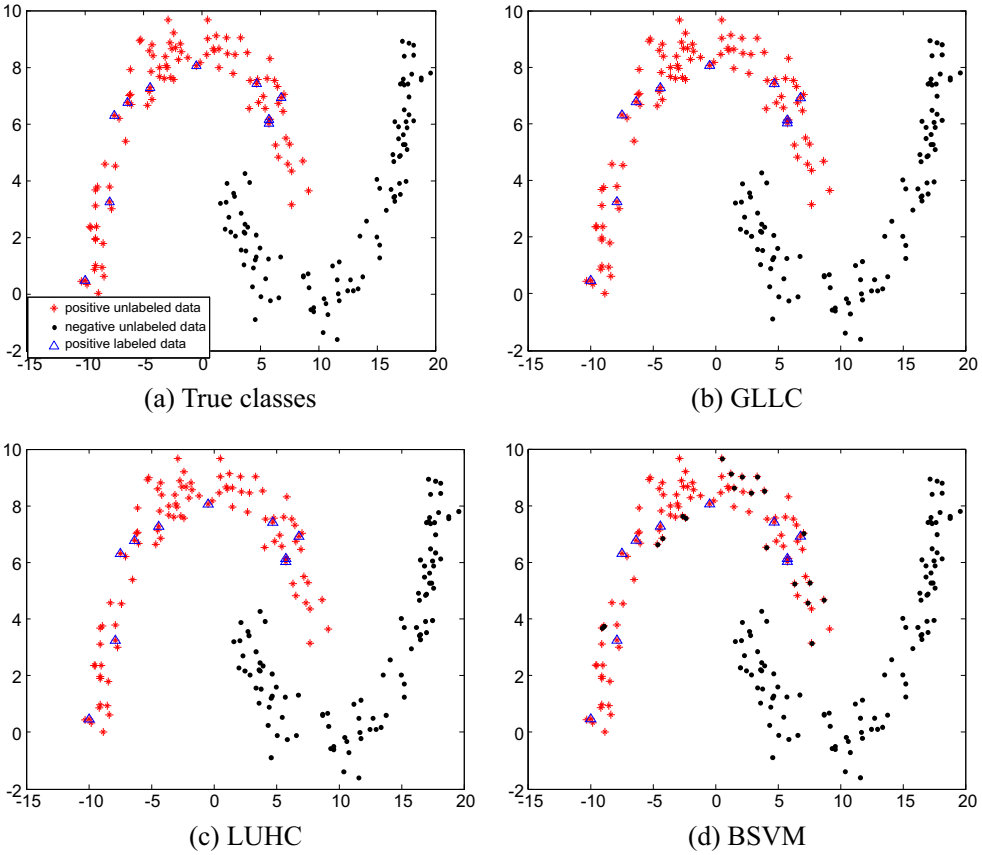

Fig. 3 Comparison classification results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 200 and the proportions of positive examples is 10%

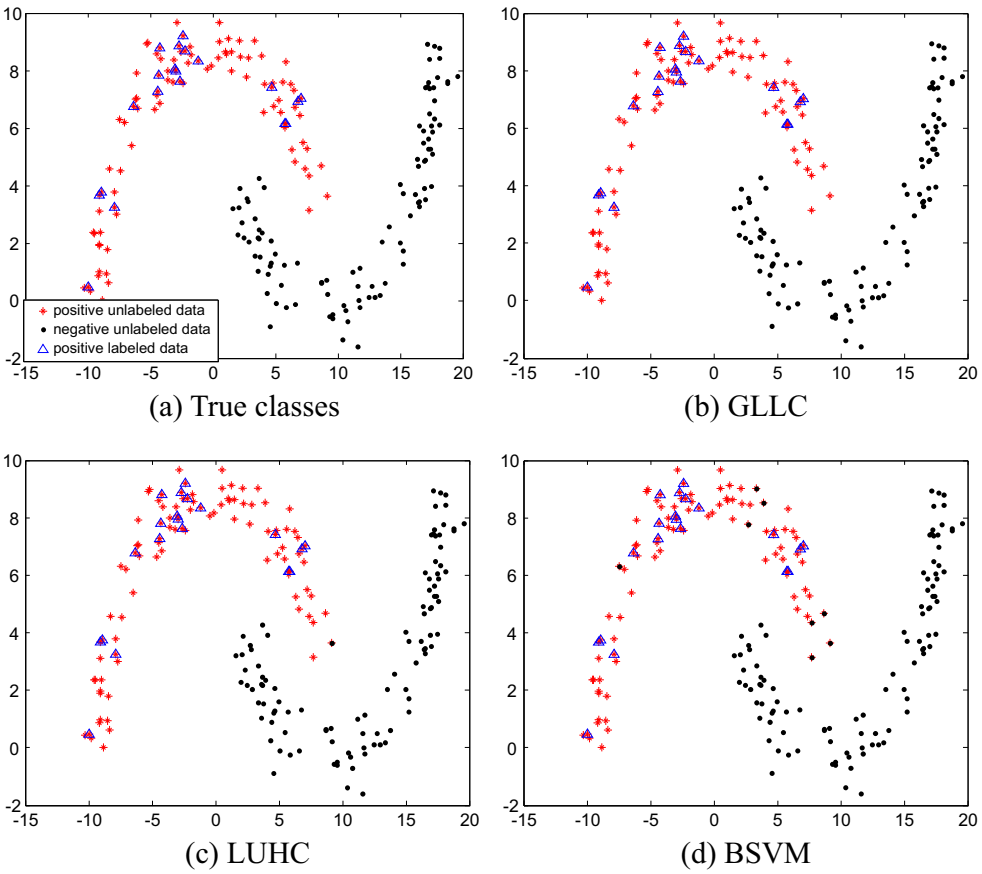

Fig. 4 Comparison results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 200 and the proportions of positive examples is 20%

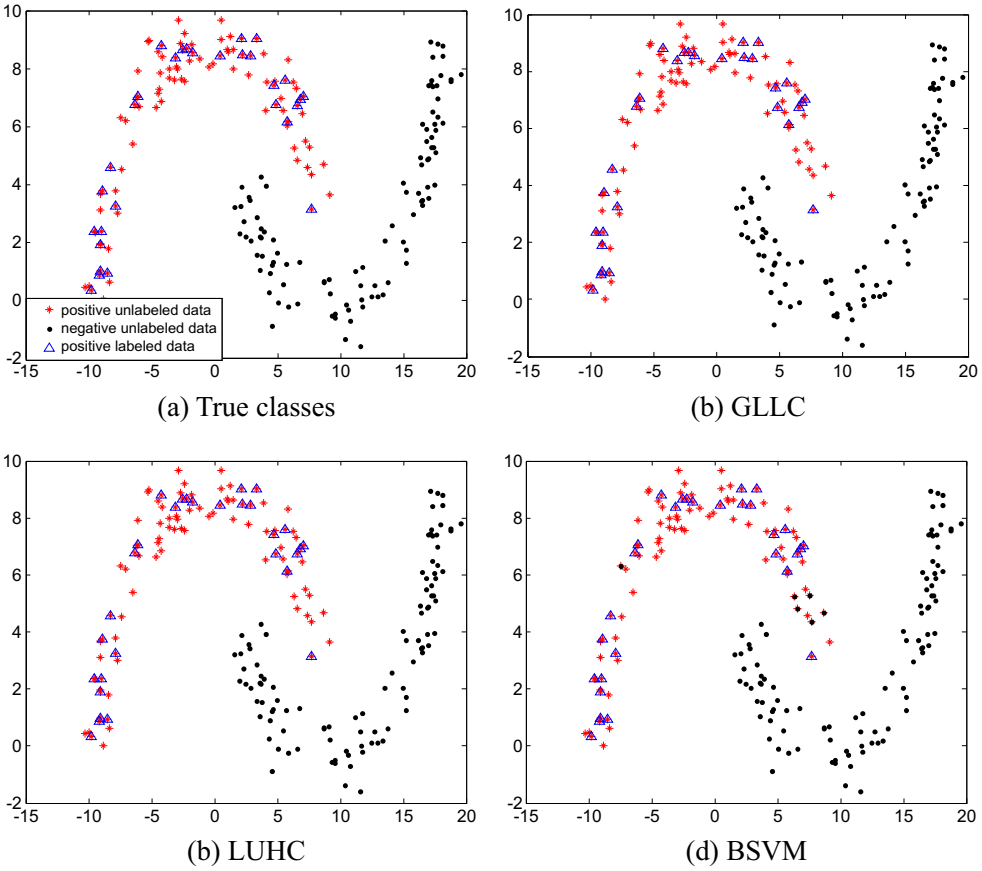

Fig. 5 Comparison results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 200 and the proportions of positive examples is 30%

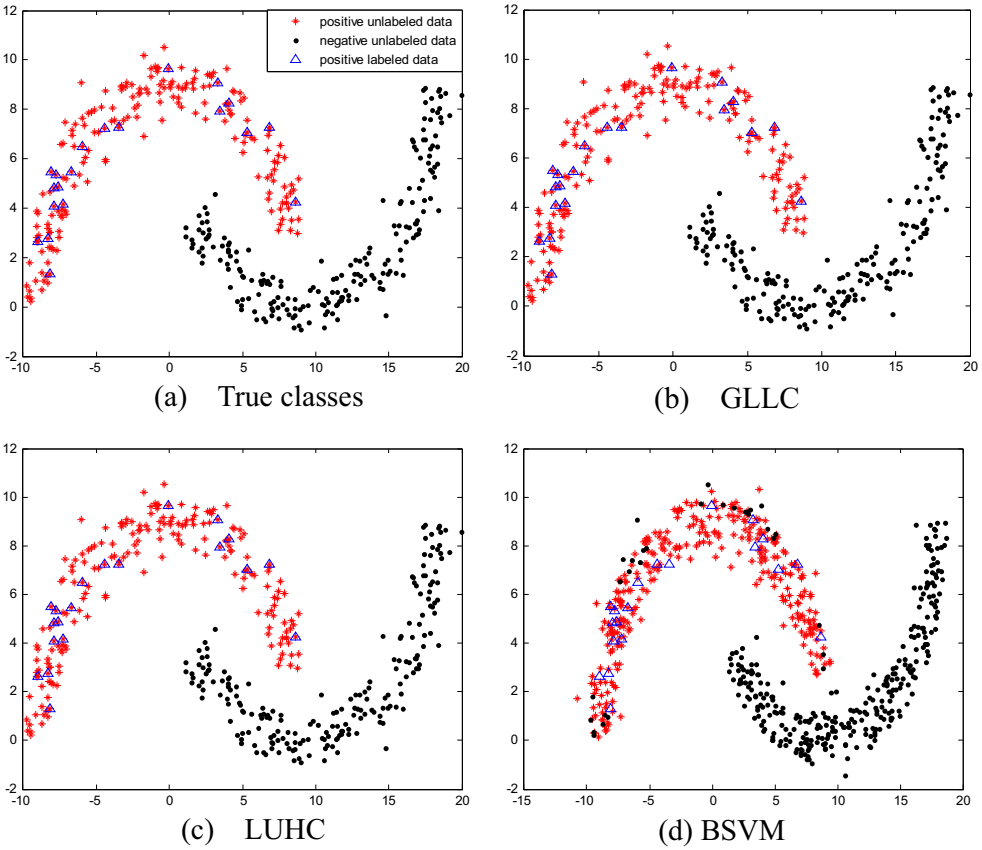

Fig. 6 Comparison results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 400 and the proportions of positive examples is 10%

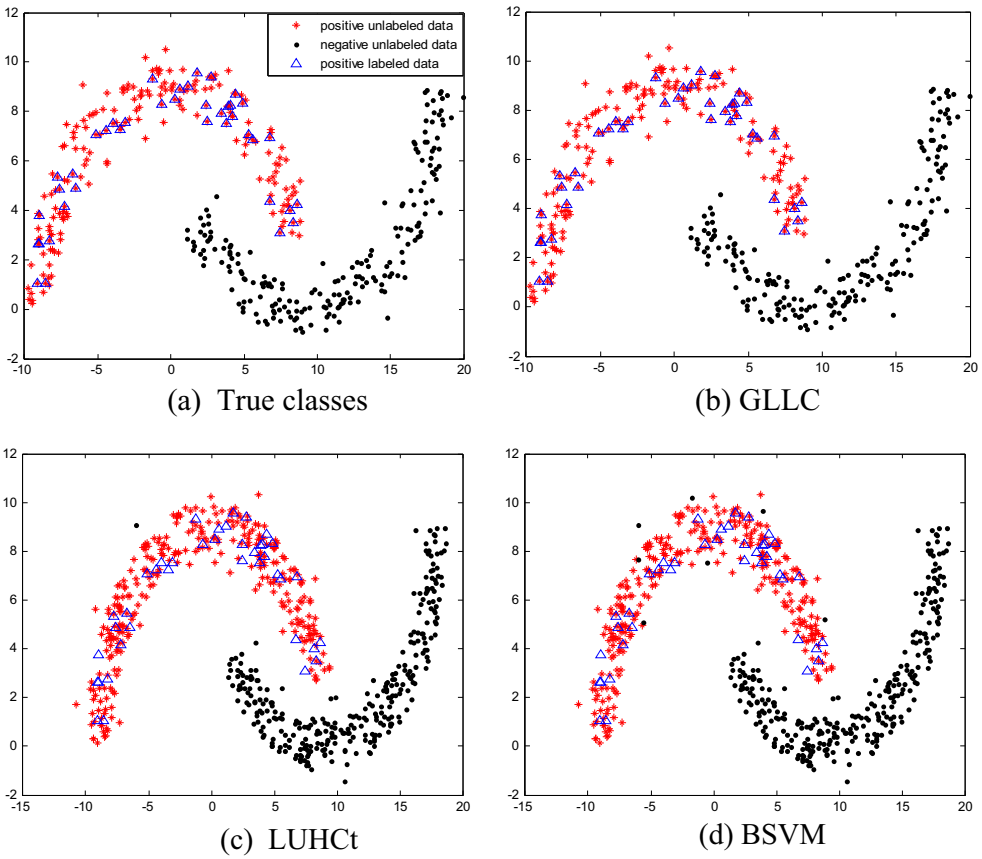

Fig. 7 Comparison results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 400 and the proportions of positive examples is 20%

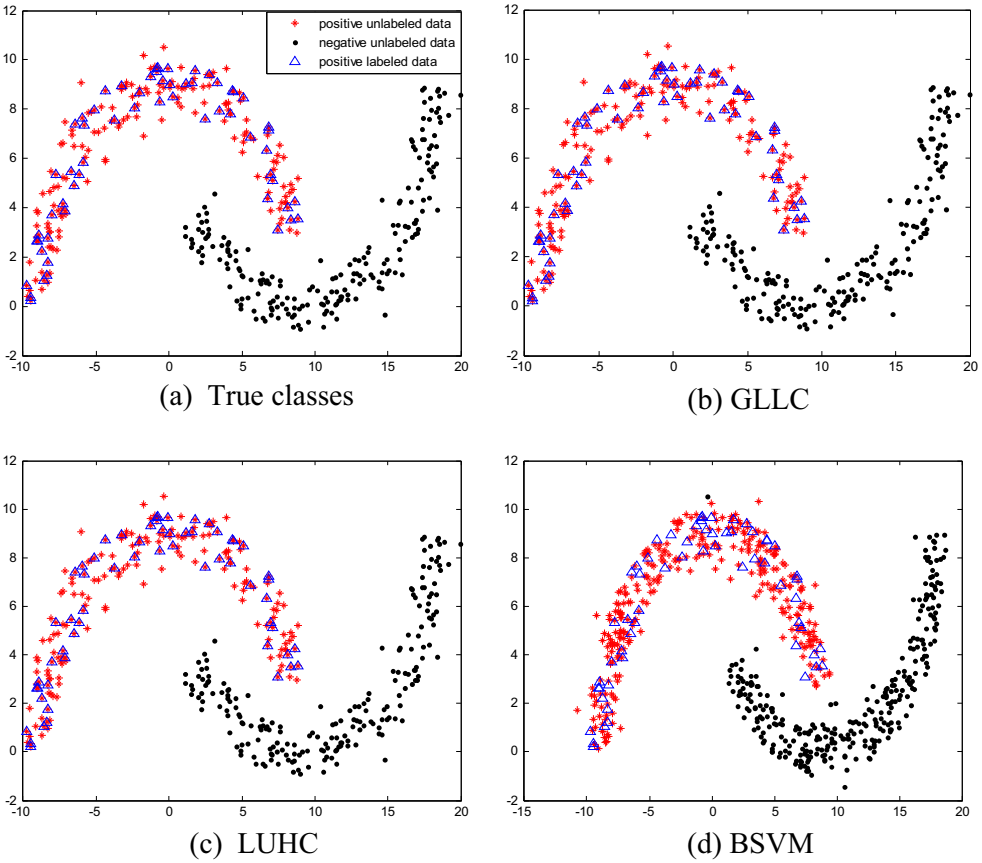

Fig. 8 Comparison results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 400 and the proportions of positive examples is 30%

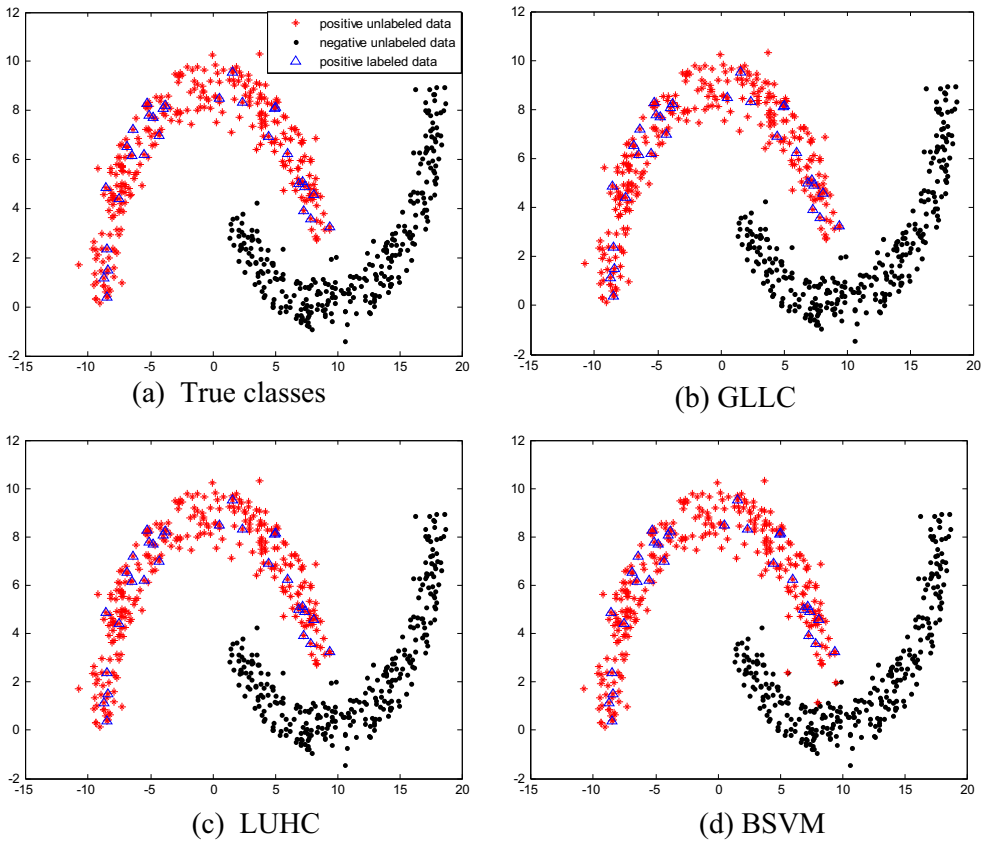

Fig. 9 Comparison results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 600 and the proportions of positive examples is 10%

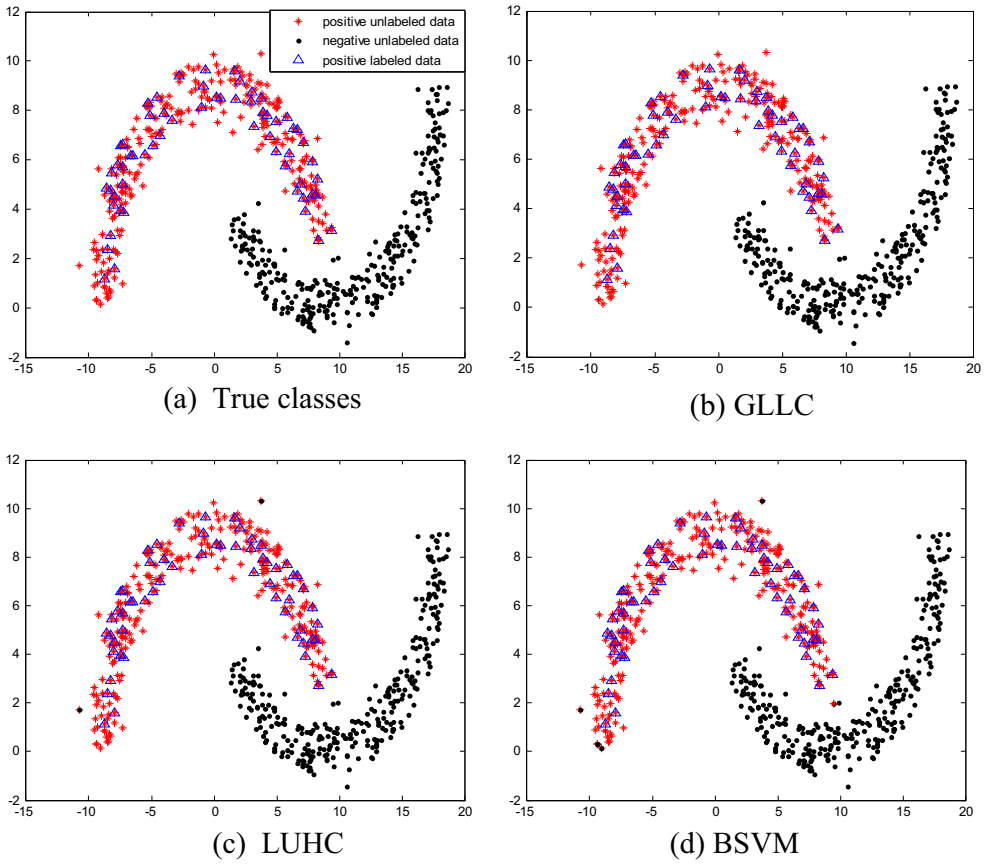

Fig. 10 Comparison results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 600 and the proportions of positive examples is 20%

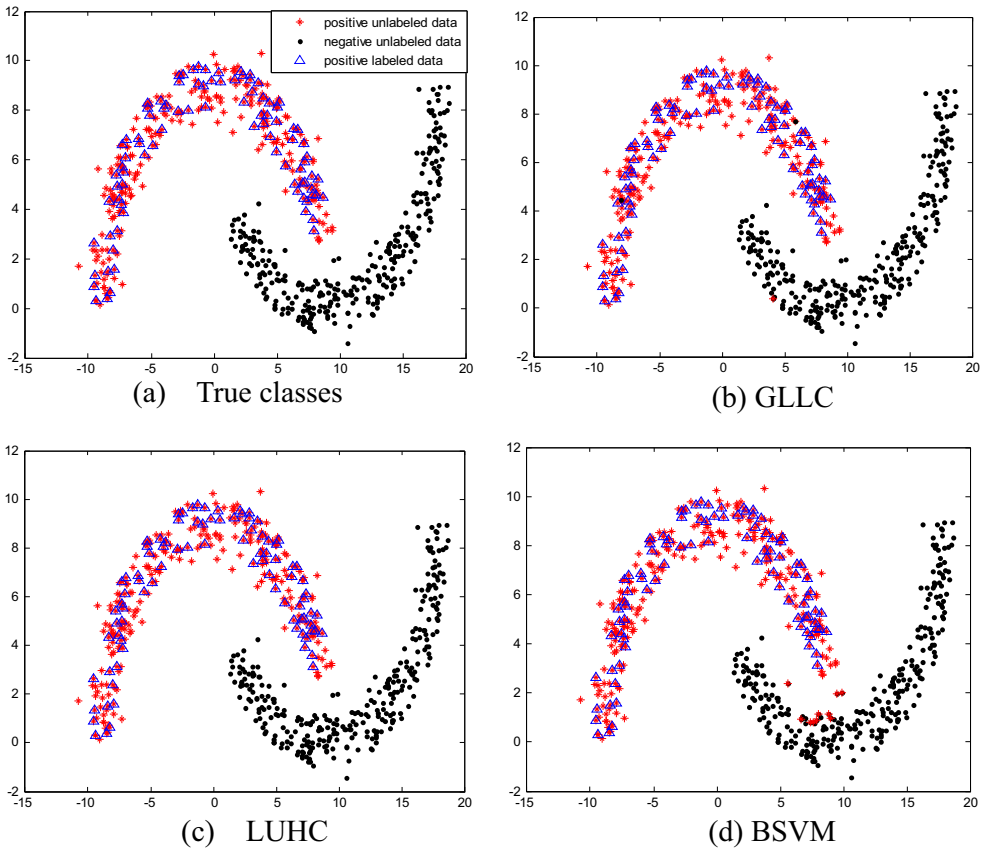

Fig. 11 Comparison results of our GLLC, LUHC and BSVM on artificial half-moons datasets when the number of labeled examples is 400 and the proportions of positive examples is 30%

of GLLC, LUHC and BSVM. More specifically, we can draw the following conclusions:

No matter how many samples and what ratio of labeled positive samples, our GLLC always identifies all positive examples, i.e., the accuracy and F value of GLLC is 100%. This result verifies effectiveness of our local learning again.

LUHC also has good classification effective, especially when the number of labeled examples is larger However, its classification power isn't as good as GLLC, such as Fig. 4c.

With the decrease of the labeled positive examples, more and more unlabeled positive examples are classified negative examples inaccurately for BSVM.

All in all, we summarize an important conclusion, that is Laplacian regularization for local learning plays a key role in classification for PU learning.

4.3.2 UCI datasets

To better illustrate the effective of our GLLC in this paper, we also provide the numerical results of GLLC and other classical methods on six UCI datasets. More concretely, our experiments are set up in the following way: firstly, each dataset is divided into two subsets: 50% for training and 50% for testing; then, we randomly select 10%, 20% and 30% of the training set as positive set, and use the remainder as unlabeled data; Finally, we transform them into PU tasks. Each experiment runs 10 times independently, and records the average F value and classification accuracy.

Table 3 shows the details of the UCI datasets. Instance: Total number of examples, Feature: Number of features, Test: Number of test examples, Train (10%): Number of

Table 3 Details of the six UCI datasets

Datasets	#Instance	#Feature	#Train(10%)	#Train(20%)	#Train(30%)	#Test
German	1000	24	70/430	140/360	210/290	500
Australian	690	14	31/304	62/283	93/252	345
Hepatitis	155	19	4/74	7/71	10/68	77
Hearts	270	14	15/120	30/105	45/90	135
CMC	1473	9	114/623	228/509	342/395	736
Ionosphere	351	34	13/163	26/150	38/138	175

Table 4 Average F value of GLLC-lin, GLLC-rbf, LUHC-lin, LUHC-rbf, Pulce, BSVM-lin, BSVM-rbf, NB, and S-EM on UCI datasets at the test set with 10% of labeled positive examples

Datasets	GLLC-lin	GLLC-rbf	LUHC-lin	LUHC-rbf	Pulce	BSVM-lin	BSVM-rbf	NB	S-EM
German	0.833	**0.826**	0.539	0.548	0.508	0.578	0.536	0.480	0.522
Australian	0.849	**0.850**	0.598	0.671	0.671	0.487	0.562	0.510	0.621
Hepatitis	0.556	0.571	0.516	0.481	**0.742**	0.256	0.571	0.355	0.425
Hearts	0.751	**0.812**	0.494	0.537	0.605	0.481	0.516	0.475	0.546
CMC	0.872	**0.874**	0.438	0.482	0.498	0.396	0.366	0.399	0.453
Ionosphere	0.727	**0.898**	0.565	0.518	0.741	0.462	0.527	0.569	0.386

Table 5 Average F value of GLLC-lin, GLLC-rbf, LUHC-lin, LUHC-rbf, Pulce, BSVM-lin, BSVM-rbf, NB, and S-EM on UCI datasets at the test set with 20% of labeled positive examples

Datasets	GLLC-lin	GLLC-rbf	LUHC-lin	LUHC-rbf	Pulce	BSVM-lin	BSVM-rbf	NB	S-EM
German	0.828	**0.827**	0.547	0.584	0.544	0.520	0.561	0.474	0.550
Australian	0.847	**0.861**	0.614	0.656	0.691	0.514	0.583	0.498	0.631
Hepatitis	0.579	0.604	0.561	0.548	**0.819**	0.428	0.514	0.449	0.532
Hearts	0.879	**0.865**	0.547	0.592	0.653	0.529	0.563	0.499	0.588
CMC	0.873	**0.873**	0.468	0.514	0.536	0.426	0.479	0.384	0.527
Ionosphere	0.781	**0.933**	0.584	0.533	0.754	0.486	0.549	0.544	0.501

Table 6 Average F value of GLLC-lin, GLLC-rbf, LUHC-lin, LUHC-rbf, Pulce, BSVM-lin, BSVM-rbf, NB, and S-EM on UCI datasets at the test set with 30% of labeled positive examples

Datasets	GLLC-lin	GLLC-rbf	LUHC-lin	LUHC-rbf	Pulce	BSVM-lin	BSVM-rbf	NB	S-EM
German	0.850	**0.835**	0.537	0.577	0.555	0.538	0.519	0.486	0.562
Australian	0.865	**0.889**	0.618	0.663	0.701	0.567	0.607	0.547	0.644
Hepatitis	0.651	0.632	0.587	0.620	**0.833**	0.500	0.538	0.489	0.608
Hearts	0.850	**0.840**	0.544	0.607	0.679	0.529	0.582	0.471	0.595
CMC	0.873	**0.877**	0.475	0.531	0.542	0.450	0.592	0.402	0.514
Ionosphere	0.786	**0.945**	0.605	0.531	0.776	0.557	0.523	0.539	0.584

Table 7 Average accuracy of GLLC-lin, GLLC-rbf, LUHC-lin, LUHC-rbf, Pulce, BSVM-lin, BSVM-rbf, NB, and S-EM on UCI datasets at the test set with 10% of labeled positive examples

Datasets	GLLC-lin	GLLC-rbf	LUHC-lin	LUHC-rbf	Pulce	BSVM-lin	BSVM-rbf	NB	S-EM
German	**0.72**	0.704	0.652	0.6832	0.627	0.6623	0.6373	0.5879	0.6676
Australian	0.8517	**0.8576**	0.723	0.7611	0.723	0.641	0.6933	0.6682	0.7341
Hepatitis	0.6923	0.7308	0.6874	0.7176	**0.804**	0.5813	0.7009	0.5481	0.6256
Hearts	0.6815	**0.7704**	0.6744	0.7301	0.65	0.6588	0.7141	0.6264	0.6629
CMC	0.7734	**0.7772**	0.6465	0.6324	0.622	0.5223	0.4673	0.5388	0.5785
Ionosphere	0.8295	**0.9261**	0.7215	0.6735	0.795	0.7015	0.7401	0.6821	0.6484

Table 8 Average accuracy of GLLC-lin, GLLC-rbf, LUHC-lin, LUHC-rbf, Pulce, BSVM-lin, BSVM-rbf, NB, and S-EM on UCI datasets at the test set with 20% of labeled positive examples

Datasets	GLLC-lin	GLLC-rbf	LUHC-lin	LUHC-rbf	Pulce	BSVM-lin	BSVM-rbf	NB	S-EM
German	0.710	**0.712**	0.668	0.698	0.686	0.646	0.672	0.568	0.674
Australian	**0.866**	0.853	0.737	0.758	0.764	0.674	0.714	0.635	0.764
Hepatitis	0.727	0.792	0.719	0.748	**0.837**	0.686	0.723	0.603	0.696
Hearts	**0.844**	0.837	0.698	0.764	0.702	0.703	0.726	0.646	0.716
CMC	**0.775**	0.775	0.618	0.658	0.674	0.584	0.628	0.524	0.617
Ionosphere	**0.954**	0.847	0.728	0.704	0.804	0.719	0.736	0.716	0.692

training examples, where 10% positive examples are selected as positive set and the rest as the unlabeled set, Train (20%): Number of training examples, where 20% positive examples are selected as positive set and the rest as the unlabeled set, Train (30%): Number of training examples, where 10% positive examples are selected as positive set and the rest as the unlabeled set.

The values in Tables 4, 5 and 6 are the mean classification F value and Tables 7, 8 and 9 show the average accuracy when 10%, 20% and 30% of the training set are selected as the labeled positive examples, respectively. The best F value and accuracy are depicted by bold figures. From these tables, we can also observe the superiority of GLLC. Compared with LUHC, BSVM for linear and Gauss kernel situations, Pulce, NB and S-EM, GLLC has the best performance, and the F value exceeds to 0.81 on all datasets except Hepatitis. The gaps between GLLC and other competitors are more and more apparent when the ratio of the labeled positive is smaller and smaller. These results verify our theory analysis again in Section 2. Namely, GLLC is insensitive to the counts of labeled positive examples. However, why is Hepatitis dataset an exception? Based on the analysis of its features, we find the dependencies among attributes are strong and the dataset is correlated. Pulce just takes into account these neighborhood function attribute relationships to train a knn-like classifier. Nevertheless, SVMs-like classifiers are only focusing on providing maximum margin hyper-plane between two classes without considering dependencies among attributes. That is why classification average F value and accuracy of Pulce are the higher than that of other SVMs classifiers.

In order to contrast more obvious, we also provide the change trend of average F value and average classification accuracy for different ratio of labeled positive examples in Figs. 12 and 13. In Figs. 12 and 13, x-axis represents the percentage of randomly labeled positive points; y-axis is the average classification F value and accuracy. It is evident from experimental results in Figs. 12 and 13 that the performance of GLLC is always excellent and effective on five datasets. F value and accuracy reaches to 0.8–0.95, whereas other competitors are just between 0.5 and 0.7. Corresponding to tables above, Pulce has the best result on Hepatitis dataset. This behavior clearly affects that the dataset are dense and correlated. Pulce exploits the dependencies among attributes. In addition, the performance of our GLLC is more stable than other methods with the change for the percentage of labeled positive examples changes.

4.3.3 Handwritten image

In the following, we conduct the performance evaluation on big data, USPS. The USPS [36] database consists of grayscale handwritten digit images from 0 to 9, as shown in Fig. 14. Each digit contains 1100 images, and the size of each image is 16 *16 pixels with 256 gray levels. Here we select four pair wise digits of varying difficulty for classification.

Figure 15a–d gives the average F value of digit 1 versus 7, 2 versus 5, 0 versus 8 and 4 versus 6 with 5%,

Table 9 Average accuracy of GLLC-lin, GLLC-rbf, LUHC-lin, LUHC-rbf, Pulce, BSVM-lin, BSVM-rbf, NB, and S-EM on UCI datasets at the test set with 30% of labeled positive examples

Datasets	GLLC-lin	GLLC-rbf	LUHC-lin	LUHC-rbf	Pulce	BSVM-lin	BSVM-rbf	NB	S-EM
German	**0.815**	0.728	0.678	0.692	0.692	0.654	0.679	0.588	0.703
Australian	0.872	**0.893**	0.745	0.763	0.773	0.709	0.741	0.675	0.751
Hepatitis	0.808	0.818	0.721	0.753	**0.858**	0.695	0.714	0.643	0.743
Hearts	**0.815**	0.807	0.723	0.752	0.724	0.711	0.743	0.697	0.759
CMC	0.775	**0.784**	0.630	0.664	0.677	0.602	0.642	0.561	0.636
Ionosphere	0.858	**0.960**	0.785	0.725	0.829	0.732	0.751	0.695	0.742

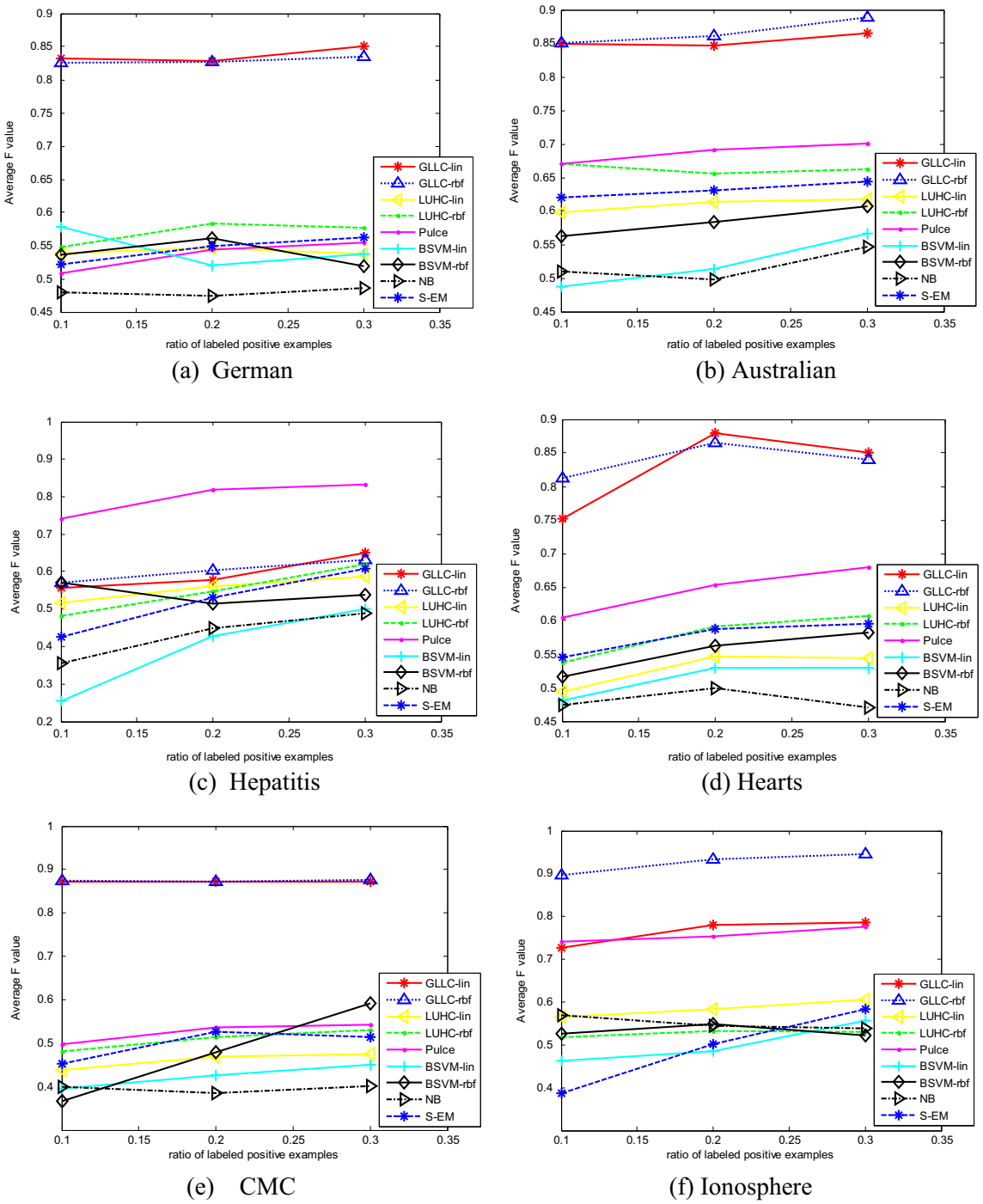

Fig. 12 Comparison the mean F value of our GLLC-lin, GLLC-rbf, LUHC-lin, LUHC-rbf, Pulce, BSVM-lin, BSVM-rbf, NB and S-EM on six UCI datasets

10%, 20%, 30%, 40% and 50% ratio of the training sets are considered as labeled positive examples, respectively. In this subsection, we choose to compare GLLC with the approaches LUHC, pulce and BSVM. We use linear kernel for our GLLC, LUHC and BSVM. From Fig. 15 (a-d), it can be noticed that competitors exhibit an increment trend in performance when the number of labeled positive examples grows from 5% to 50%. Different from these approaches, our GLLC shows excellent results, F value is always near to 100% whatever the ratio of labeled positive examples is. The difference of competitors and GLLC is obvious.

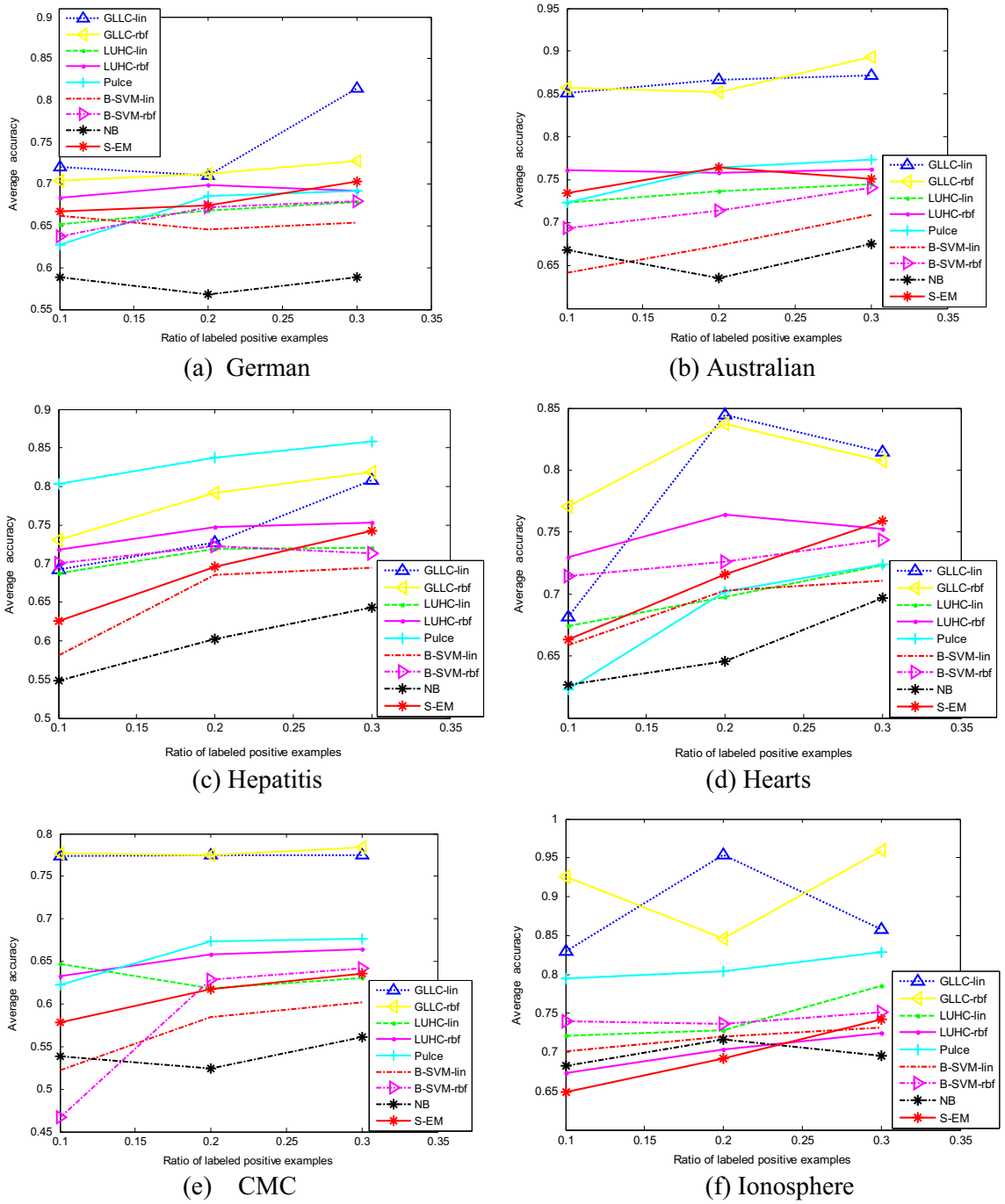

Fig. 13 Comparison the mean accuracy of our GLLC-lin, GLLC-rbf, LUHC-lin, LUHC-rbf, Pulce, BSVM-lin, BSVM-rbf, NB and S-EM on six UCI datasets

Fig. 14 Representation of Handwritten image on USPS

4.4 Experimental analysis

4.4.1 The analysis of parameters

To further analyze our GLLC, in this subsection, we report the influence of our experiment results on the four parameters λ, σ, C_p and C_n The common approach to select the parameters is to find their optimal values from a range of

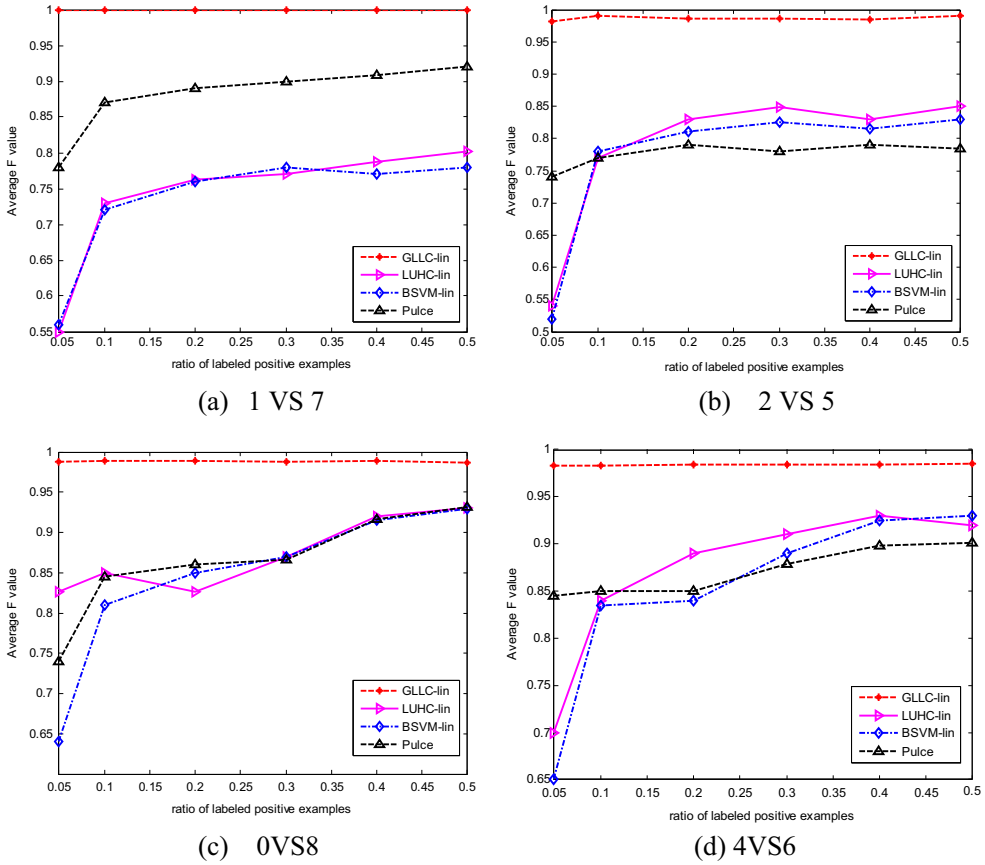

Fig. 15 F value of GLLC-lin, LUHC-lin, Pulce, BSVM-lin on handwritten image datasets, where x-axis is the ratio of positive labeled examples

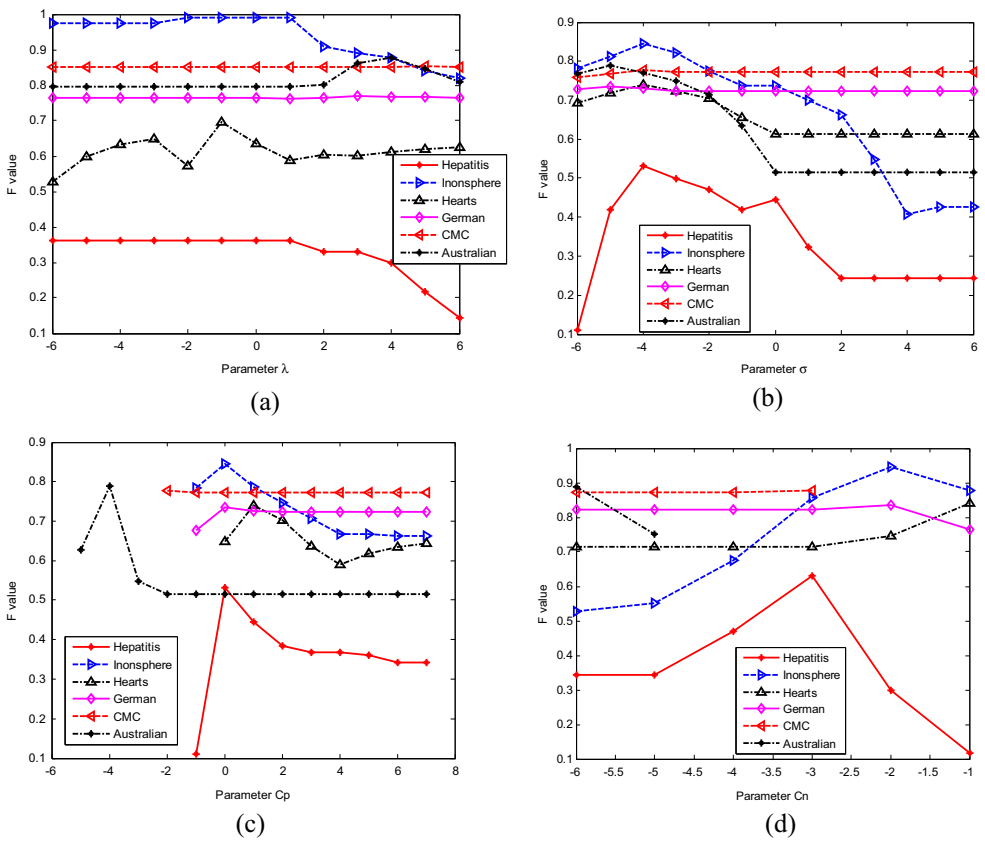

Fig. 16 Influence of parameter λ, σ, C_p and C_n to F value when the ratio of labeled positive examples is 0.3

values. Below, we take six UCI datasets as an example to analyze how the parameters affect the performance of GLLC. Figure 16a–d shows the average F value of GLLC with these four parameters when the ratio of labeled positive examples is 0.3. From Fig. 16a–d, we can observe that F value does change as the parameter changes. This observation means the parameters effect on the classification performance of our GLLC. As for a future comment, if only the parameters reach suitable values, average F value would be higher. In other words, the influence of parameters is obvious and we have to select best values to promise the optimal measure. Besides, Fig. 16c–d shows the change trend of C_p and C_n, subject to $C_p > C_n$.

4.4.2 The analysis of time complexity

In this subsection, we also analyze the time complexity of GLLC. Take the prediction of handwritten image as an example. Figure 17 lists the mean training central processing unit (CPU) time of GLLC with some other competitors, where the time units are seconds. From the Fig. 17, we can see GLLC always consume the least amount of mean training time in the same hardware environment no matter what ratio of labeled positive examples is. Moreover, with the ratio of the labeled positive examples increasing, the gaps of computational time between GLLC and other competitors are with a growing trend. It is reasonable that GLLC just solve a group of linear equations which will produce little effect on the computational time, whereas LUHC, BSVM have to solve QPP and Pulce computes an amount of distances by considering dependencies of attributions which will have a dramatic effect, especially when the ratio of labeled positive increases. To sum up, compared with LUHC, BSVM and Pulce, our GLLC not only has better classification ability but also spends remarkably less computational time

5 Conclusions

In this paper we have put forward a novel classifier combined with global and local regularization, named GLLC for short. In theory, we have proved that our GLLC is not only stable and robust to the changes of the counts of labeled positive examples but also has very low computational time. Experiments on artificial datasets, six UCI datasets and handwritten image classification have shown that GLLC is more effective than LUHC, BSVM and other popular methods for PU learning. In brief, GLLC has more strong discriminative power for positive and unlabeled learning.

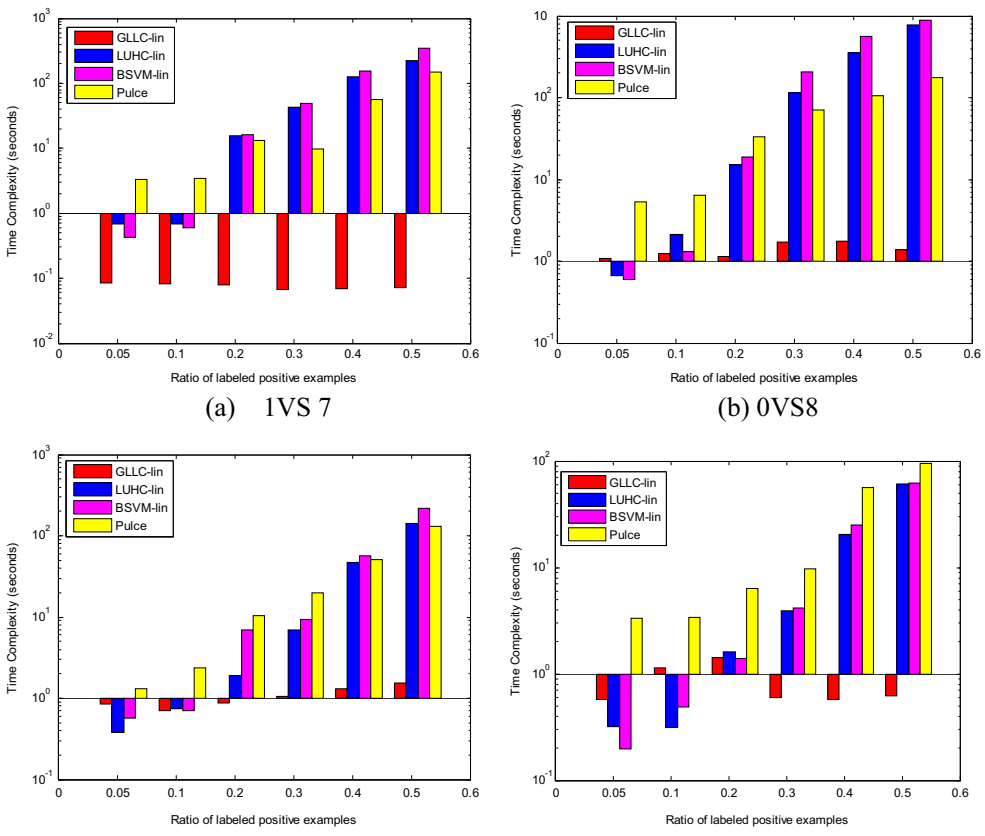

Fig. 17 Training consuming time of GLLC-lin, LUHC-lin, Pulce, BSVM-lin on handwritten image datasets, where x-axis is the ratio of positive labeled examples

Acknowledgments This work is supported by the youth innovative Foundation of Tianjin University of Science & Technology (2016LG30, 2016LG29).

References

1. Kılıc C, Tan M (2010) Positive unlabeled learning for deriving protein interaction networks. Network Modeling and Analysis in Health Informatics and Bioinformatics 1(3):87–102
2. Nguyen MN, Li XL, Ng SK (2011) Positive unlabeled learning for time series classification. In: Proceedings of the twenty-second international joint conference on artificial intelligence, pp 1421–1426
3. Li XL, Yu PS, Liu B, Ng SK (2009) Positive unlabeled learning for data stream classification. In: Proceedings of the ninth SIAM international conference on data mining (SDM'09), pp 257–268
4. Pan S, Zhang Y, Li X (2012) Dynamic classifier ensemble for positive unlabeled text stream classification. Knowl Inf Syst 33(2):267–287
5. Wang S, Chen ZY, Liu B (2016) Mining aspect-specific opinion using a holistic lifelong topic model. In: Proceedings of the international World Wide Web conference
6. Chen ZY, Ma NZ, Liu B (2015) Lifelong learning for sentiment classification. In: Proceedings of the 53st annual meeting of the association for computational linguistics, pp 26–31
7. Denis F (1998) PAC Learning from positive statistical queries. Lect Notes Comput Sci 1501:112–126
8. Muggleton S (1997) Learning from the positive data. machine learning, inductive logic programming. Lect Notes Comput Sci 1314:358–376
9. Liu B, Dai Y, Li XL, Lee WS, Yu PS (2003) Building text classifiers using positive and unlabeled examples. In: Proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, Florida, United States. IEEE. pp 179–188
10. Yu H, Han J, Chang KCC (2004) PEBL: Web Page classification without negative examples. IEEE Trans Knowl Data Eng 16(1):70–81
11. Vapnik VN (1995) The nature of statistical learning theory. Springer, Berlin
12. Christoffe M, Plessis D, Sugiyama M (2014) Semi-supervised learning of class balance under class-prior change by distribution matching. Neural Netw 50:110–119
13. Li XL, Liu B (2003) Learning to classify text using positive and unlabeled data. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence, Acapulco, vol 18. Springer, Mexico, pp 587–594
14. Fung GPC, Yu JX, Lu H, Yu PS (2006) Text classification without negative examples revisit. IEEE Trans Knowl Data Eng 18(1):6–20
15. Nguyen MN, Li XL, Ng SK (2011) Positive unlabeled learning for time series classification. In: Proceedings of international joint conference on artificial intelligence, IJCAI, pp 1421–1426
16. Ienco D, Pensa RG (2016) Positive and unlabeled learning in categorical data. Neurocomputing 196:113–124
17. Liu B, Lee WS, Yu PS et al (2002) Partially supervised classification of text documents. In: Proceedings of the 19th international conference on machine learning, pp 387–394
18. Schkopf B, John CP, John S, Alex J, Robert C (2001) Estimating the Support of a High-dimensional Distribution. Neural Comput 13(7):1443–1471
19. Zhu F, Ye N, Yu W, Xu S, Li GB (2014) Boundary detection and sample reduction for one-class support vector machines. Neurocomputing 123:166–173
20. Zhou K, Xue GR, Yang Q, Yu Y (2010) Learning with positive and unlabeled examples using topic-sensitive. PLSA, IEEE Trans Knowledge Data Eng 22(1):46–58
21. Zhang D, Lee WS (2005) A simple probabilistic approach to learning from positive and unlabeled examples. In: Proceedings of the 5th annual UK workshop on computational intelligence (UKCI), pp 83–87
22. Elkan C, Noto K (2008) Learning classifiers from only positive and unlabeled data. In: Proceedings of the 14th international conference on knowledge discovery and data mining, Las Vegas, vol 58(1). ACM, USA, pp 213–220
23. Luigi C, Charles E, Michele C (2010) Learning gene regulatory networks from only positive and unlabeled data. Bioinformatics 11(1):228–240
24. Lee WS, Liu B (2003) Learning with positive and unlabeled examples using weighted logistic regression. In: Proceedings of the 20th international conference on machine learning, Washington, vol 20. AAAI, United States, pp 448–455
25. Ke T, Yang B, Tan JY, Jing L (2012) Building high-performance classifiers on positive and unlabeled examples for text classification. Advances in Neural Networks ISNN, 2012. Lect Notes Comput Sci 7368:187–195
26. Shao YH, Chen WJ, Liu LM, Deng NY (2015) Laplacian unit-hyperplane learning from positive and unlabeled examples. Inf Sci 314:152–1687
27. Sellamanickam S, Garg P, Selvaraj SK (2011) A pairwise Ranking Based Approach to Learning with Positive and Unlabeled Examples. In Proceedings of the 20th ACM International Conference on Information and Knowledge Management, Glasgow, United Kingdom. ACM, New York, USA. 663–672
28. Suykens JAK (2000) Least squares support vector machines for classification and nonlinear modeling. Neural Network World 10(1–2):29–47
29. Chapelle O, Schokopf B, Zien A et al (2006) Semi-supervised learning. MIT press, Cambridge
30. Ke T, Tan JY, Yang B, Song LJ, Jing L (2014) A novel graph-based approach for transductive positive and unlabeled learning. J Comput Inf Syst 10(1):1–8
31. Zhang ZQ, Ke T, Deng NY, Tan JY (2014) Biased p-norm support vector machine for PU learning. Neurocomputing 136(136):256–261
32. Wang F (2010) A general learning framework using local and global regularization. Pattern Recogn 43:3120–3129
33. Blake CL, Merz CJ (1998) UCI Repository for Machine Learning Databases. <http://www.ics.uci.edu/mlearn/MLRepository.html>
34. Lin ZR (2016) LIBSVM. http://www.csie.ntu.edu.tw/~cjlin/libsvm
35. Liu B (2008) LPU package http://www.cs.uic.edu/~liub/LPU/LPU-download.html
36. USPS (1998) USPS Database. <http://www.cs.nyu.edu/roweis/data.html>

Ting Ke received Ph.D. degree in Mathematics from China Agricultural University in 2014. She is a lecturer in Tianjin University of Science & Technology, China. Her main research interesting includes semi-supervised learning, PU learning and optimization research and applications.

Ling Jing received Master's degree and Ph.D. in Mathematics from Chongqing University and Beijing University of Aeronautics & Astronautics, China, in 1994 and 1997, respectively. She is a professor and Ph. D. supervisor a China Agricultural University. Her main research interests lie in data mining and optimization research.

Hui Lv received Ph.D. degree in Mathematics from Tsinghua University in 2014. She is a lecturer in Tianjin University of Science & Technology, China. Her main research interests lie in numerical algebra and algorithm analysis.

Lidong Zhang is an Associate Professor in the Department of Mathematics at Tianjin University of Science & Technology, China. He obtained his Bachelor degree from the Hebei Normal University in 2003, and his Master Degree from Nankai University in 2006. In 2014 he obtained his Ph.D. degree in Management Science and Engineering from Tianjin University. His research interests include risk management, financial engineering and optimal control.

Yaping Hu received Ph.D. degree in Mathematics from East China University of Science and Technology in 2015. She is a lecturer in Tianjin University of Science & Technology, China. Her main research interesting includes Compressive Sensing and optimization research.