Semi-Supervised Classification via Local Spline Regression

Shiming Xiang, Feiping Nie, and Changshui Zhang, *Member, IEEE*

***Abstract*—This paper presents local spline regression for semi-supervised classification. The core idea in our approach is to introduce splines developed in Sobolev space to map the data points directly to be class labels. The spline is composed of polynomials and Green's functions. It is smooth, nonlinear and able to interpolate the scattered data points with high accuracy. Specifically, in each neighborhood, an optimal spline is estimated via regularized least squares regression. With this spline, each of the neighboring data points is mapped to be a class label. Then, the regularized loss is evaluated and further formulated in terms of class label vector. Finally, all the losses evaluated in local neighborhoods are accumulated together to measure the global consistency on the labeled and unlabeled data. To achieve the goal of semi-supervised classification, an objective function is constructed, by combining together the global loss of the local spline regressions and the squared errors of the class labels of the labeled data. In this way, a transductive classification algorithm is developed, in which a globally optimal classification can be finally obtained. In the semi-supervised learning setting, the proposed algorithm is analyzed and addressed into the Laplacian regularization framework. Comparative classification experiments on many public data sets and applications to interactive image segmentation and image matting illustrate the validity of our method.**

***Index Terms*—Semi-supervised classification, local spline regression, interactive image segmentation.**

I. Introduction

In recent years, semi-supervised classification has been received more and more attentions in machine learning and pattern recognition. The motivation behind semi-supervised classification is to employ a large number of unlabeled data to help build a better classifier from the labeled data. This yields desired work settings for many real-world applications where unlabeled data is abundant but labeling a large number of data points needs expensive human labor. A typical example may be webpage classification. It is easy to collect webpages, but labeling them into different topics may require experienced annotators to work for a long time. Thus how to make use of the unlabeled data to improve the performance of classifiers becomes an important problem. Such a motivation opens a hot research direction in machine learning.

A large family of semi-supervised learning algorithms have been proposed [13], [60]. The earliest algorithms are mainly developed from generative models, including Gaussian mixture model [38], mixture of experts [33], and the extensions [35], [18]. As mentioned in [60], an important problem in these algorithms is the model correctness. That is, with an incorrect model assumption, unlabeled data may even hurt the accuracy [15], [60]. Co-training is another important methodology [10]. The role of unlabeled data in co-training is to help reduce the size of version space. Later, many variants of co-training are developed [60], [27], [58], [16], [20], [14], [57], [2], [3]. Comparative experiments show that co-training is very effective if the sub-feature sets used to learn the classifiers are conditional independent from each other [34]. There are also many other algorithms constructed with different criteria, such as transductive support vector machine [25], Gaussian process [28], and those based on statistical physics theories [19], [44]. Real performances on many data sets show their ability to infer the class labels of unlabeled data points.

Many semi-supervised learning algorithms are developed on data graph [56], [59], [9], [4], [6], [41], [46], [52], [47], [45]. Typically, Zhu et al. [59] proposed an algorithm via harmonic energy minimization. Based on the Gaussian Random Field (GRF), harmonic functions are employed to propagate the label information to the unlabeled data. Zhou et al. [56] proposed an algorithm called Learning with Local and Global Consistency (LLGC), in which an iterative framework is constructed on data manifold via normalized Laplacian [39]. Later, Wang et al. [46] presented another label propagation algorithm via linear neighborhoods. A common point is that they can run iteratively. More formally, the task can be roughly addressed as the following optimization problem [24], [4], [46], [56], [59], [52]:

$$\min_{\mathbf{f}} \left(\mathbf{f}^T \mathbf{M} \mathbf{f} + \gamma (\mathbf{f} - \mathbf{y})^T \mathbf{D} (\mathbf{f} - \mathbf{y}) \right), \qquad (1)$$

where $\mathbf{f}$ is related to the class labels of all data, $\mathbf{y}$ records the class labels of the labeled data, $\mathbf{D}$ is a diagonal matrix whose diagonal elements are one for labeled data and zero for unlabeled data, and γ is a trade-off parameter.

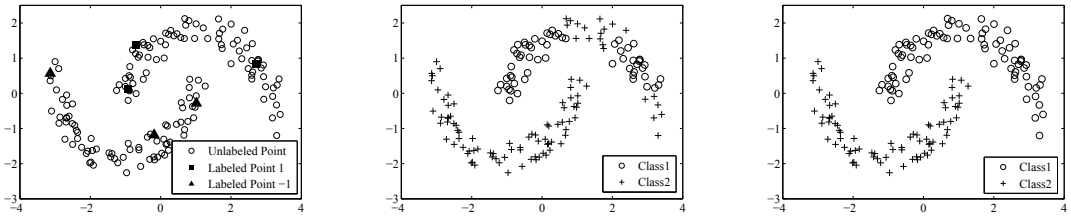
A key task in (1) is to construct matrix $\mathbf{M}$ from data graph. To this end, a basic assumption behind this is the *clustering assumption*, which states that the data points forming the same structure (manifold) are likely to have the same label. Thus, modeling the neighboring data points will play fundamental roles in exploring the data structures [6], [56], [7], [51], [50], [52], [23], [55], [37]. Accordingly, many local evaluations are introduced, typically including data affinity between neighbors [56], locally linear presentations [37], [46], [52], and locally nonlinear presentation with kernels [6], [51]. In spite

Manuscript received September 11, 2008; revised August 14, 2009.

S. Xiang is with National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China, e-mail: smxiang@gmail.com

Feiping Nie and Changshui Zhang are with State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology, Department of Automation, Tsinghua University, Beijing, 100084, China, e-mail: feipingnie@gmail.com, zcs@mail.tsinghua.edu.cn

Digital Object Indentifier 10.1109/TPAMI.2010.35 0162-8828/10/$26.00 © 2010 IEEE



(a) (b) (c)

Fig. 1. Classification of two half moons using the method in [56]. (a) The 160 data points with three labeled data points in each half moon; (b) The classification results with $\sigma = 0.013$; (c) The classification results with $\sigma = 0.021$. In experiments, the parameter α in Eq. (6) in [56] is fixed to be 0.99, and the size of neighborhood for each data point is set to be 7.

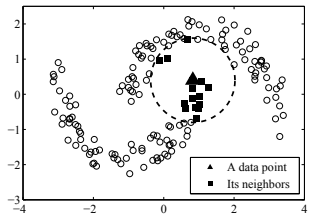


Fig. 2. A data point (solid triangle) and its 14 neighbors (solid squares). This data set is just the same as that used in Fig. 1. The coordinates of these 15 data points are listed in Table I.

Coordinates	True	Linear	Gaussian	Spline
(0.827, 0.399)	-1	-0.4508	-1.0000	-1.0000
(0.813, 0.167)	-1	-0.6229	-1.0000	-1.0000
(1.073, 0.373)	-1	-0.7316	-1.0000	-1.0000
(1.268, 0.204)	-1	-1.0726	-1.0000	-1.0000
(1.013, -0.070)	-1	-1.0240	-1.0000	-1.0000
(0.819, -0.129)	-1	-0.8667	-1.0001	-1.0000
(0.587, -0.265)	-1	-0.7310	-1.0001	-1.0000
(1.016, -0.283)	-1	-1.1978	-1.0001	-1.0000
(0.907, -0.373)	-1	-1.1556	-1.0001	-1.0000
(0.978, -0.397)	-1	-1.2494	-1.0001	-1.0000
(0.627, -0.411)	-1	-0.8899	-1.0001	-1.0000
(0.101, 1.004)	1	0.8012	1.0000	1.0000
(-0.121, 0.958)	1	0.9985	1.0000	1.0000
(0.938, -0.697)	-1	-1.4482	-1.0000	-1.0000
(0.674, 1.558)	1	0.6408	1.0000	1.0000
Average error	—	0.075	1.7×10^{-9}	3.7×10^{-28}

TABLE I
THE FIRST AND SECOND COLUMN LIST THE 15 NEIGHBORING DATA POINTS IN FIG. 2 AND THEIR TRUE LABELS. FROM THE THIRD TO THE FIFTH COLUMN ARE THE MAPPED RESULTS WITH (4), (3), AND (7) (SEE SECTION II-A), RESPECTIVELY. THE LAST ROW REPORTS THE AVERAGES OF THE SQUARED ERRORS.

of the great successes in semi-supervised learning, there still exist some issues to be further addressed:

(1) A commonly-used affinity measure between two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ is the Gaussian function [56], [59]:

$$w_{ij} = \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/(2\sigma^2)). \quad (2)$$

In (2), we face to supply a well-tuned σ to obtain satisfactory results. A problem is that a small change of σ may generate different classification results. Fig. 1 shows an example, where LLGC [56] is implemented. Only 139/160 correct classification rate is obtained with $\sigma = 0.013$, while 100% correct recognition rate is achieved with $\sigma = 0.021$. Such a problem may also occur in regression frameworks constructed with Gaussian functions [6]:

$$f_i = \sum_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j), \quad (3)$$

where f_i is the class label of $\mathbf{x}_i$ and $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/(2\sigma^2))$.

(2) Although locally linear representations have no such parameters to be well tuned to data, they may lack the ability to accurately capture the local geometry. In addition, if the size of the neighborhoods is supplied inappropriately, linear representations may introduce model errors. Fig. 2 shows a data point and its 14 neighbors, whose coordinates and true class labels are listed in the first and second column in Table I. The linear transformation can be used to map them to be class labels:

$$f = \mathbf{W}^T\mathbf{x} + b. \quad (4)$$

In (4), the optimal $\mathbf{W} \in \mathbb{R}^2$ and $b \in \mathbb{R}$ can be estimated under the least squares framework, based on the 15 data points and their class labels. The optimal transformation can be used in turn to map them. The results are listed in the third column in Table I. As can be seen, most of them are far away from their true values. Similarly, we can also model them with Eq. (3). The fourth column in Table I reports the results obtained via Eq. (3) with $\sigma = 0.65$. The average of the squared errors indicates that the nonlinear model is more accurate than the linear model.

In this paper, we hope to construct the model with powerful nonlinear description as well as easiness in usage. To this end, we use splines developed in Sobolev space [1] to replace the commonly used linear and kernel functions, and develop a semi-supervised classification algorithm via spline regression on local neighborhoods. This spline is a combination of polynomials and Green's functions, which is popularly used to interpolate scattered data in geometrical design [11]. Recently, we also used this spline in nonlinear dimensionality reduction [53]. Practice also shows that it can effectively handle high dimensional data [53], [43]. This spline is smooth, nonlinear and able to interpolate the scattered data points with high accuracy. This is desired for obtaining accurate local evaluations. For the above 15 data points in Table I, for

example, with this spline, the average of the squared errors is only about 3.7×10^{-28} (see the last column in Table I).

After the regularized objective function is minimized, the regularized loss is evaluated and then formulated in terms of class label vector. Matrix $\mathbf{M}$ is finally constructed by summing the local losses estimated from all of the neighborhoods. We prove that $\mathbf{M}$ is a Laplacian matrix. Thus, our method can be addressed into the Laplacian regularization framework [5], [56]. This illustrates its connections to the existing methods. Meanwhile, the main algorithm can be easily implemented. Comparative classification experiments on many public data sets and applications to interactive natural image segmentation and image matting illustrate the validity of our proposed method.

The remainder of this paper is organized as follows. Section II constructs the spline and evaluates the regularized loss. The algorithm is given in Section III. Section IV reports the experimental results. Applications in interactive natural image segmentation and image matting are given in Section V, followed the conclusions in Section VI.

II. Class Label Regressions on Neighborhoods

Our problem can be formulated as follows. Given n data points $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_l, \mathbf{x}_{l+1}, \cdots, \mathbf{x}_n\} \in \mathbb{R}^m$ and the class labels of the first l data points in $\{f_i\}_{i=1}^{l}$. The task is to infer the class labels $\{f_i\}_{i=l+1}^{n}$ of the remaining $n-l$ unlabeled data points in $\{\mathbf{x}_i\}_{i=l+1}^{n}$.

To construct matrix $\mathbf{M}$ in (1), splines developed in Sobolev space [1] will be used to map the neighboring data points and the regularized loss will be evaluated. Matrix $\mathbf{M}$ will be finally constructed by summing together the local losses estimated from all of the neighborhoods.

Sobolev space is composed of integrable functions with weak derivatives. In contrast to Hilbert space, it is capable of supplying functions with more general forms. The motivations that we use the spline developed in Sobolev space are summarized as follows. (1) This spline is smooth, nonlinear and able to interpolate the scattered data points with arbitrarily controllable accuracy. (2) With this spline, the regularized loss can be formulated in terms of class label vector. Although the class labels are currently unknown, this formulation gives us a way to develop an objective function, from which the class labels are finally solved.

A. Spline Regression

For the moment, we consider the two-class case. In this case, the class label f_i of $\mathbf{x}_i$ will belong to the two-value label set $\mathcal{L} = \{+1, -1\}$.

For each data point $\mathbf{x}_i \in \mathcal{X}$, its k nearest neighbors in $\mathcal{X}$ can be determined via Euclidean distance metric. Denote these k neighbors including $\mathbf{x}_i$ itself by $\mathcal{N}_i = \{\mathbf{x}_{i_j}\}_{j=1}^{k}$, where subscript i_j stands for an index, namely, $i_j \in \{1, 2, \cdots, n\}$ and $i_1 = i$. Now the task is to find a function $g_i : \mathbb{R}^m \to \mathbb{R}$ such that it can directly map each source data point $\mathbf{x}_{i_j} \in \mathbb{R}^m$ to be a class label

$$f_{i_j} = g_i(\mathbf{x}_{i_j}), \quad j = 1, 2, \cdots, k. \tag{5}$$

For the above regression task, we can consider the following general objective function with data fitting and function smoothness:

$$J(g_i) = \sum_{j=1}^{k} (f_{i_j} - g_i(\mathbf{x}_{i_j}))^2 + \lambda S(g_i), \tag{6}$$

where $S(g_i)$ is a penalty functional, and λ is a trade-off parameter.

With different hypothesis spaces, minimizing $J(g_i)$ in (6) will generate g_i with different forms. Actually, if the linear space is considered, we can get linear transformations as formulated in (4). If the general reproducing kernel Hilbert space is considered, we can obtain nonlinear function mappings. One common form of such mappings is a combination of Gaussian functions as formulated in (3). Here we consider to find the continuous function in Sobolev space [1]. Accordingly, the penalty functional $S(g_i)$ is then defined as a semi-norm [17], [42]. With this form, research work [17] shows that the minimizer g_i in (6) will be given by

$$g_i(\mathbf{x}) = \sum_{j=1}^{d} \beta_{i,j}\, p_j(\mathbf{x}) + \sum_{j=1}^{k} \alpha_{i,j}\, \phi_{i,j}(\mathbf{x}), \tag{7}$$

where $\phi_{i,j}(\mathbf{x})$ is a Green's function [17], namely, $\phi_{i,j}(\mathbf{x}) = (||\mathbf{x} - \mathbf{x}_{i_j}||)^{2s-m} \log(||\mathbf{x} - \mathbf{x}_{i_j}||)$, if m is even; $\phi_{i,j}(\mathbf{x}) = (||\mathbf{x} - \mathbf{x}_{i_j}||)^{2s-m}$, otherwise. Here $||\cdot||$ denotes the Euclidean distance, and s is the order of the partial derivatives in the semi-norm [42].

In (7), $d = C^{s}_{m+s-1}$ and $\{p_j(\mathbf{x})\}_{j=1}^{d}$ are a set of primitive polynomials which can span the polynomial space with degree less than s. In the case of $s = 2$ (linear polynomial space) and $m = 3$, for example, denoting $\mathbf{x} = [x_1, x_2, x_3]^T \in \mathbb{R}^3$, then we have four primitive polynomials: a constant term $p_1(\mathbf{x}) = 1$, and three monomials $p_2(\mathbf{x}) = x_1$, $p_3(\mathbf{x}) = x_2$ and $p_4(\mathbf{x}) = x_3$. In this case, the spline is a C_1-smoothness function ($2s - m = 1$).

Substituting the k data points in $\mathcal{N}_i$ into (7) will yield k equations. However, there are totally $k + d$ coefficients to be solved. To this end, we introduce d new equations, according to the definition of conditionally positive semi-definiteness of functions [54], [49]. For k data points in $\mathcal{N}_i$, we obtain

$$\begin{pmatrix} \mathbf{e}^T \\ \mathbf{P}_i \end{pmatrix} \alpha_i = \mathbf{0}, \tag{8}$$

where $\mathbf{e}$ is a k-dimensional vector with all 1s, namely, $\mathbf{e} = [1, 1, \cdots, 1]^T \in \mathbb{R}^k$ (corresponding to the constant primitive polynomial "1" in $\{p_j(\mathbf{x})\}_{j=1}^{d}$), $\mathbf{P}_i \in \mathbb{R}^{(d-1)\times k}$ collects the values of the $d-1$ non-constant primitive polynomials in $\{p_j(\mathbf{x})\}_{j=1}^{d}$ with the k data points in $\mathcal{N}_i$ as inputs, and α_i collects the coefficients in (7), namely, $\alpha_i = [\alpha_{i,1}, \alpha_{i,2}, \cdots, \alpha_{i,k}]^T \in \mathbb{R}^k$.

Now considering the regularization parameter λ, and combining together the k equations derived from (7), we have

$$\begin{pmatrix} \mathbf{K}_i + \lambda \mathbf{I} & \mathbf{e} & \mathbf{P}_i^T \\ \mathbf{e}^T & 0 & \mathbf{0} \\ \mathbf{P}_i & 0 & \mathbf{0} \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_{i,0} \\ \beta_i \end{pmatrix} = \begin{pmatrix} \mathbf{F}_i \\ 0 \\ \mathbf{0} \end{pmatrix}, \tag{9}$$

where $\mathbf{K}_i$ is a $k \times k$ symmetrical matrix with the r-th row and c-th column element $K_{r,c} = \phi_{i,c}(\mathbf{x}_{i_r})$, $\mathbf{F}_i = [f_{i_1}, f_{i_2}, \cdots, f_{i_k}]^T \in \mathbb{R}^k$ collects the class labels of the

k data points in $\mathcal{N}_i$, $\mathbf{I}$ is a $k \times k$ identity matrix, $\alpha_i = [\alpha_{i,1}, \alpha_{i,2}, \cdots, \alpha_{i,k}]^T \in \mathbb{R}^k$, $\beta_{i,0} \in \mathbb{R}$, and $\beta_i = [\beta_{i,1}, \beta_{i,2}, \cdots, \beta_{i,d-1}]^T \in \mathbb{R}^{d-1}$.

Note that the regularization parameter λ is added to balance between the smoothness of the spline and the accuracy near the k scattered data points. That is, a larger λ will result in that the spline is more smooth (flat), but the interpolation errors will be increased. On the contrary, a smaller λ will decrease the smoothness of the spline, but the interpolation accuracy near the scattered data points will be improved. In computation, a small λ can be employed to achieve the goal of balance.

In Table I, the last column values are obtained via (7) with linear polynomials. When solving Eq. (9), λ is set to be 0.001. The average of the squared errors listed in the last row in Table I indicates that spline can yield more accurate results.

Finally, we point out that, if we do not consider the second part of the Green's functions in (7), $g_i(\mathbf{x})$ will turn out to be the commonly used polynomial function in statistical analysis. Accordingly, the regression task can be addressed in a penalty-free polynomial space. In general, polynomial regressions can generate satisfactory results, especially when we are given sufficient observation data. However, a main drawback of polynomial regression is that the primitive polynomials are non-local. In other words, the regression value of a data point is depended not only on itself but also on those data points far from it. As a result, although the global error can be reduced to some degree, the regression values at some data points may be far from the true values. In Table I, for example, the average of the squared errors under the linear polynomial is only about 0.075, while at the first data point, the bias is greater than 0.5. In contrast, although the spline in (7) includes the polynomial function, it shows the property of locality. Actually, with this spline, the data points are first globally mapped with the polynomial function, and then they are locally warped with the radial basis functions (namely, the Green's functions) defined at the scattered data points. As a result, better local fitting near the scattered data points can be achieved.

B. The Regularized Loss

Note that currently we can not solve Eq. (9) since the class label vector $\mathbf{F}_i = [f_{i_1}, f_{i_2}, \cdots, f_{i_k}]^T \in \mathbb{R}^k$ are actually unknown. Alternatively, we analyze the regularized loss. Previous research shows that for k scattered data points, the regularized loss minimization $J(g_i(\mathbf{x}))$ in Eq. (6) can be approximately evaluated as follows [42]:

$$J(g_i) \approx \sum_{j=1}^{k} (f_{i_j} - g_i(\mathbf{x}_{i_j}))^2 + \lambda \alpha_i^T \mathbf{K}_i \alpha_i. \qquad (10)$$

where the r-th row and c-th column element of $\mathbf{K}_i$ is calculated as $K_{r,c} = \phi_{i,c}(\mathbf{x}_{i_r})$.

As can be seen from Eq. (9), for a small λ, the conditions in (5) can be approximately satisfied. This means that the first term in (10) can be evaluated as zero. Then, we have $J(g_i) \approx \lambda \alpha_i^T \mathbf{K}_i \alpha_i$.

Theorem 1: For a small λ, the regularized loss minimization can be evaluated in terms of class label vector:

$$J(g_i) \approx \lambda \mathbf{F}_i^T \mathbf{M}_i \mathbf{F}_i, \qquad (11)$$

where $\mathbf{M}_i$ is the upper left $k \times k$ sub-matrix of the inverse matrix of the coefficient matrix in Eq. (9).

Proof: Based on Eq. (9), we have $(\mathbf{K}_i + \lambda \mathbf{I})\alpha_i + \mathbf{e}\beta_{i,0} + \mathbf{P}_i^T \beta_i = \mathbf{F}_i$, $\mathbf{e}^T \alpha_i = 0$ and $\mathbf{P}_i \alpha_i = \mathbf{0}$. Note that $\mathbf{M}_i$ is the upper left $k \times k$ sub-matrix of the inverse matrix of the coefficient matrix in Eq. (9). Then, solving Eq. (9) can generate $\alpha_i = \mathbf{M}_i \mathbf{F}_i$. Now it follows that $\alpha_i^T (\mathbf{K}_i + \lambda \mathbf{I})\alpha_i + \alpha_i^T \mathbf{e}\beta_{i,0} + \alpha_i^T \mathbf{P}_i^T \beta_i = \alpha_i^T \mathbf{F}_i \Rightarrow \alpha_i^T (\mathbf{K}_i + \lambda \mathbf{I})\alpha_i = \mathbf{F}_i^T \mathbf{M}_i \mathbf{F}_i$. The last equality holds since $\mathbf{M}_i$ is a symmetrical matrix. Thus for a small λ, we have $J(g_i) \approx \lambda \alpha_i^T \mathbf{K}_i \alpha_i \approx \lambda \alpha_i^T \mathbf{K}_i \alpha_i + \lambda^2 \alpha_i^T \alpha_i = \lambda \mathbf{F}_i^T \mathbf{M}_i \mathbf{F}_i$. ■

The formulation in (11) indicates that we can evaluate the regularized loss minimization in a compact way. More importantly, this provides us a way to develop the learning model from which the class labels are inferred. Furthermore, for $\mathbf{M}_i$, we have the following theorem:

Theorem 2: $\mathbf{M}_i$ is a Laplacian matrix.

Proof: We need to prove two points. One is that $\mathbf{M}_i$ is positive semi-definite and the other is that the sum of elements in each row of $\mathbf{M}_i$ equals to zero.

Following Theorem 1, it is easy to justify that for any vector $\mathbf{z} \in \mathbb{R}^k$, there exists a vector $\alpha \in \mathbb{R}^k$ such that $\alpha^T (\mathbf{K}_i + \lambda \mathbf{I})\alpha = \mathbf{z}^T \mathbf{M}_i \mathbf{z}$. In addition, α should also satisfy the conditions in (8). Note that the Green's function is a conditionally positive semi-definite function [49]. In addition, the conditions in (8) are actually the conditions which stipulate the solution to α should be located in the null space of $\mathbf{P}_i$. Based on the conditionally positive semi-definiteness, for such α, it holds that $\alpha^T \mathbf{K}_i \alpha \geq 0$. Note that λ is a positive number, thus $\mathbf{z}^T \mathbf{M}_i \mathbf{z} \geq 0$. This indicates that $\mathbf{M}_i$ is positive semi-definite.

For the second point, we only need to prove $\mathbf{M}_i \mathbf{e} = \mathbf{0}$. Here $\mathbf{e} = [1, 1, \cdots, 1]^T \in \mathbb{R}^k$. Actually, replacing $\mathbf{F}_i$ in Eq. (9) by $\mathbf{e}$, we can obtain a vector $\alpha \in \mathbb{R}^k$. Following the proof in Theorem 1, for this α, it also holds that $\mathbf{e}^T \mathbf{M}_i \mathbf{e} = \alpha^T \mathbf{e}$. In addition, from Eq. (9), we have $\alpha^T \mathbf{e} = \mathbf{e}^T \alpha = 0$. Thus $\mathbf{e}^T \mathbf{M}_i \mathbf{e} = 0$.

According to matrix theory [21], for semi-definite matrix $\mathbf{M}_i$, there exists a matrix $\mathbf{Q}$ such that $\mathbf{Q}^T \mathbf{Q} = \mathbf{M}_i$. Then $\mathbf{e}^T \mathbf{M}_i \mathbf{e} = \mathbf{0} \Rightarrow \mathbf{e}^T \mathbf{Q}^T \mathbf{Q} \mathbf{e} = \mathbf{0} \Rightarrow \mathbf{Q}\mathbf{e} = \mathbf{0} \Rightarrow \mathbf{Q}^T \mathbf{Q} \mathbf{e} = \mathbf{0} \Rightarrow \mathbf{M}_i \mathbf{e} = \mathbf{0}$. Thus we finish the proof. ■

The fact that $\mathbf{M}_i$ is a Laplacian matrix will give us a way to address our algorithm into the general Laplacian regularization framework.

Finally, we point out that, in view of data graph, the construction of $\mathbf{M}_i$ can actually allow negative edges in graph. The negative edges are generated since the spline regression is intrinsically a discriminative learning approach. That is, all the neighboring data points are employed together to learn a function through which their class labels are predicted. As a result, the constructed graph explores the relations between all of the neighboring data points, and thus shows local discriminating power. In contrast, LLGC [56] is developed on traditional graph Laplacian. That is, each pair of data points are independently assigned an edge weight. In this way, the constructed graph has always non-negative edges. However, independently assigning edge weights may be inadequate to natural images where strong structural dependencies exist

in local patches. With such weighted graphs, unsatisfactory segmentations may be generated. Such a drawback could be avoided to some degree via spline regressions (see Fig. 7 and Fig. 8).

III. Algorithm

In this section, we introduce the construction of matrix $\mathbf{M}$ and present the transductive classification algorithm.

A. Constructing Matrix $\mathbf{M}$

Section II introduces the spline and evaluates the regularized loss in each neighborhood $\mathcal{N}_i$, which is formulated in (11). Now it is natural to add together the losses estimated on all of the n neighborhoods $\{\mathcal{N}_i\}_{i=1}^n$. Minimizing this total loss will achieve global consistency on the labeled and unlabeled data. Now we have

$$E(\mathbf{f}) \approx \lambda \sum_{i=1}^{n} \mathbf{F}_i^T \mathbf{M}_i \mathbf{F}_i, \tag{12}$$

where $\mathbf{f} = [f_1, f_2, \cdots, f_n]^T \in \mathbb{R}^n$ is the vector of class labels.

Note that $\mathbf{F}_i = [f_{i_1}, f_{i_2}, \cdots, f_{i_k}]^T \in \mathbb{R}^k$ is just a sub-vector of the label vector $\mathbf{f}$. Thus there exists a row selection matrix $\mathbf{S}_i \in \mathbb{R}^{k \times n}$ such that $\mathbf{F}_i = \mathbf{S}_i \mathbf{f}$. The r-th row and c-th column element $s_i(r,c)$ of $\mathbf{S}_i$ can be defined as follows:

$$s_i(r,c) = \begin{cases} 1, & \text{if } c = i_r \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

In computation, we can ignore the coefficient λ in (12). Then, it follows

$$E(\mathbf{f}) \propto \sum_{i=1}^{n} \mathbf{F}_i^T \mathbf{M}_i \mathbf{F}_i = \mathbf{f}^T \mathbf{M} \mathbf{f}, \tag{14}$$

where $\mathbf{M} = \sum_{i=1}^{n} \mathbf{S}_i^T \mathbf{M}_i \mathbf{S}_i$.

Note that each $\mathbf{M}_i$ in (14) is a $k \times k$ matrix. Since some neighborhoods may intersect with each other, $\mathbf{M}$ will contain at most nk^2 non-zero elements. In the case of $k \ll n$, $\mathbf{M}$ is highly sparse. Thus it is possible for us to treat very large-scale data sets on PCs.

B. Solving Class Labels

After matrix $\mathbf{M}$ is constructed, we can minimize the objective function in (1) to obtain the class labels of the unlabeled data points. As mentioned previously, matrix $\mathbf{D} \in \mathbb{R}^{n \times n}$ in (1) is a diagonal matrix whose diagonal elements are one for labeled data points and zero for unlabeled data points, γ is positive parameter, and $\mathbf{y} \in \mathbb{R}^n$ is known. Denote $\mathbf{y} = [y_1, y_2, \cdots, y_n]^T$, then

$$y_i = \begin{cases} 1, & \text{if } \mathbf{x}_i \text{ is labeled as positive} \\ -1, & \text{if } \mathbf{x}_i \text{ is labeled as negative} \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

With the above constructions of $\mathbf{D}$ and $\mathbf{y}$, we see that the second term in (1) associates to the sum of the squared errors of the l labeled data points, namely, $(\mathbf{f}-\mathbf{y})^T \mathbf{D} (\mathbf{f}-\mathbf{y}) = \sum_{i=1}^{l} (f_i - y_i)^2$. Based on (14), the objective function in (1) can be rewritten as follows:

$$G(f) = \mathbf{f}^T \mathbf{M} \mathbf{f} + \gamma \sum_{i=1}^{l} (f_i - y_i)^2. \tag{16}$$

According to (14) the first term is just the sum of the regularized losses on all of the neighborhoods, which means that the predicted class labels should not change too much between neighboring data points. The second term is related to class label fitting, which means that the predicted class labels of the labeled data points should not change too much from the given class labels. The trade-off between these two terms is controlled by parameter γ. It can be easily justified that the labeled values will be exactly satisfied, when γ equals to positive infinity.

Now based on (1), we can get the globally optimal solution to $\mathbf{f}$ as follows:

$$\mathbf{f} = (\mathbf{M} + \gamma \mathbf{D})^{-1} \gamma \mathbf{D} \mathbf{y}, \tag{17}$$

After the class label vector $\mathbf{f}$ is solved from Eq. (17), each data point $\mathbf{x}_i$ can be classified as "+1", if $f_i \geq 0$; "-1", otherwise.

It is easy to extend the above computation to multi-class classification problems. Suppose we are given c classes. Different from two-class problems, here we use a class label matrix $\mathbf{F} = [\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_c] \in \mathbb{R}^{n \times c}$ to replace the label vector $\mathbf{f}$ in (1). Accordingly, the problem can be reformulated as follows:

$$\min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \operatorname{tr}\left(\mathbf{F}^T \mathbf{M} \mathbf{F} + \gamma (\mathbf{F} - \mathbf{Y})^T \mathbf{D} (\mathbf{F} - \mathbf{Y}) \right), \tag{18}$$

where tr is the trace operator of matrix, and $\mathbf{Y}$ is a $n \times c$ matrix, which records the information of the labeled data points. The element of $\mathbf{Y}$ is defined as follows [56]:

$$y_{ij} = \begin{cases} 1, & \text{if } \mathbf{x}_i \text{ is labeled as class } j \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

Note that the steps of constructing $\mathbf{M}$ in (18) are similar to those described for two-class problems. We explain this point as follows. In multi-class cases, in each neighborhood $\mathcal{N}_i$, the k-dimensional label vector $\mathbf{F}_i$ in (11) will be replaced by a $k \times c$ class label matrix. This means that, in each neighborhood, the spline in (7) will be employed c times, and totally c splines will be constructed to interpolate respectively the c column vectors. Accordingly, we can get c losses, each of which can be written as the same form as that in (11). Then we can sum these c losses together. With trace operator of matrix, and using the row selection matrix $\mathbf{S}_i$, this sum can be finally formulated as $\lambda \operatorname{tr}(\mathbf{F}^T \mathbf{S}_i^T \mathbf{M}_i \mathbf{F} \mathbf{S}_i)$. Here $\mathbf{M}_i$ equals to that in (11). Finally, similar to Eq. (14), we can get $E(\mathbf{F}) \propto \sum_{i=1}^{n} \operatorname{tr}(\mathbf{F}^T \mathbf{S}_i^T \mathbf{M}_i \mathbf{F} \mathbf{S}_i) = \operatorname{tr}(\mathbf{F}^T \mathbf{M} \mathbf{F})$. Thus we see $\mathbf{M}$ in (18) is identical to that in Eq. (14).

Now solving Problem (18), we can get the globally optimal solution to $\mathbf{F}$ as follows:

$$\mathbf{F} = (\mathbf{M} + \gamma \mathbf{D})^{-1} \gamma \mathbf{D} \mathbf{Y}. \tag{20}$$

After the class label matrix $\mathbf{F}$ is solved, $\mathbf{x}_i$ is finally classified into class $f_i = \arg\max_{1 \leq j \leq c} F_{ij}$.

Note that in Eq. (17) and Eq. (20) directly calculating the inverse matrix of $(\mathbf{M}+\gamma\mathbf{D})$ may be very expensive, especially when the number of the data points n is very large. However, $(\mathbf{M}+\gamma\mathbf{D})$ is a sparse matrix, especially in the case of $k \ll n$. Thus we can employ the existing sparse algorithms to solve the linear equations.

The algorithm steps are given in Table II. Note that, for high dimensional data, constructing a smooth spline may need a huge number of primitive polynomials. As a result, the size of the coefficient matrix in Eq. (9) will become very large. Solving the equations could be uneconomical or even impossible. In practice, an alternative way is to only keep the linear polynomial. Research work shows such a compromise can also generate satisfactory results in regression analysis [43]. In our transductive learning setting, only k neighboring data points are considered to construct matrix $\mathbf{M}_i$. In particular, k is usually very small. Thus, we can use principal component analysis [26] to project these data points into a subspace. In experiments in this paper, the dimensionality of the subspace is fixed to be three. The k projected points will be finally employed to calculate $\mathbf{M}_i$. This will speed up the computation as well as guarantee the smoothness of the splines.

Input: $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_l, \mathbf{x}_{l+1}, \cdots, \mathbf{x}_n\} \in \mathbb{R}^m$, $\{\mathbf{x}_i\}_{i=1}^{l}$ are labeled, $\{\mathbf{x}_i\}_{i=l+1}^{n}$ are unlabeled. The number of nearest neighbors k, the parameter γ in Problem (1), and the parameter λ in Eq. (9).

Output: The class labels of all the data points.

1) Construct matrix $\mathbf{M}$ from $\mathcal{X}$
   a) For each data point $\mathbf{x}_i \in \mathcal{X}$, determine its k nearest neighboring data points including $\mathbf{x}_i$ itself, $\mathcal{N}_i = \{\mathbf{x}_{i_j}\}_{j=1}^{k}$.
   b) Construct the coefficient matrix in Eq. (9), based on $\mathcal{N}_i$.
   c) Calculate $\mathbf{M}_i$ in Eq. (11), and accumulate it to matrix $\mathbf{M}$, namely, $\mathbf{M} \leftarrow \mathbf{M}_i$.
2) In the two-class case
   a) Construct vector $\mathbf{y}$ according to Eq. (15).
   b) Solve linear equations: $(\mathbf{M} + \gamma\mathbf{D})\mathbf{f} = \gamma\mathbf{D}\mathbf{y}$.
   c) Label $\mathbf{x}_i$ as "+1", if $f_i \geq 0$; "-1", otherwise.
3) In the multi-class case
   a) Construct matrix $\mathbf{Y}$ according to Eq. (19).
   b) Solve linear equations: $(\mathbf{M} + \gamma\mathbf{D})\mathbf{F} = \gamma\mathbf{D}\mathbf{Y}$.
   c) Label $\mathbf{x}_i$ as $l_i = \arg\max_{1 \leq j \leq c} F_{ij}$.

TABLE II
ALGORITHM DESCRIPTION.

C. Algorithm Analysis

To illustrate the property of our algorithm, we give the following theorem:

Theorem 3: $\mathbf{M}$ in (14) is a Laplacian matrix.

Proof: Note that $\mathbf{S}_i$ is a row selection matrix. Thus, in Eq. (14), to add $\mathbf{S}_i^T\mathbf{M}_i\mathbf{S}_i$ is just to add the elements of $\mathbf{M}_i$ to the corresponding elements of $\mathbf{M}$. Due to the fact that $\mathbf{M}_i$ is a Laplacian matrix, then $\mathbf{S}_i^T\mathbf{M}_i\mathbf{S}_i$ is also a Laplacian matrix. Note that the sum of n Laplacian matrices is also a Laplacian matrix. Thus we prove this theorem. ■

Theorem 3 indicates that our algorithm will share the same properties of Laplacian regularization framework [5]. Under this framework, we can explain the algorithm in view of data manifold. Since $\mathbf{M}$ is a Laplacian matrix, it can be viewed as a Laplacian-Beltrami operator on data manifold, which treats $\mathbf{f}$ (here discretized as a vector) as a function on this manifold such that the global geometry of data is exploited from neighborhoods. As a result, the term $\mathbf{f}^T\mathbf{M}\mathbf{f}$ measures the smoothness of the class labels. Thus the validity of the developed algorithm is theoretically guaranteed.

Data set	Size	Labeled	Class	Feature
USPS	1500	150	2	241(30)
Digital1	1500	150	2	241(30)
News20	1980	200	4	8014(30)
Breast	683	68	2	10
Diabetes	768	76	2	8
Iris	150	15	3	4
Ionosphere	351	36	2	34
Vehicle	846	84	4	18
Vote	435	44	2	16

TABLE III
A BRIEF DESCRIPTION OF THE DATA SETS.

IV. Experimental Results

A. Data Sets

Image Data Sets: USPS and Digital1. The data points in USPS are generated from the USPS benchmark[1]. The data points in Digital1 are transformed from the image of digital "1"[2]. No explicit structures are hidden in USPS data set. In contrast, the data points in Digital1 show strong manifold structure.

Text Data Set: News20. Here we conduct text classification experiments on the 20-newsgroups data set[3]. The articles including autos, motorcycles, baseball, and hockey are chosen from the version 20-news-18828. The articles were preprocessed with the same procedure as that used in [56].

UCI Data Sets: Breast, Diabetes, Iris, Ionosphere, Vehicle and Vote. These are taken from the UCI machine learning repository [4], which are commonly used to test semi-supervised classification algorithms.

Table III describes these data sets. For the first three high dimensional data sets, principal component analysis [26] is used to project them respectively into 30-dimensional subspace. Then all the semi-supervised algorithms for comparisons will be run on the same data.

B. Algorithms and Parameter Settings

We will compare our algorithm, Local Spline Regression (LSR), with several typical semi-supervised classification algorithms, including GRF [59], LLGC [56], Local Learning Regularization (LL-Reg) [52], Laplacian Regularized Least Squares (LapRLS) [6]. In experiments, the algorithm of Locally Linear Embedding Regularizer (LLE-Reg) [52] is also compared. To be clear, here we briefly explain LLE-Reg and LL-Reg. Given data point $\mathbf{x}_i$ and its neighborhood $\mathcal{N}_i = \{\mathbf{x}_{i_j}\}_{j=1}^{k}$ with $\mathbf{x}_i = \mathbf{x}_{i_1}$, LLE-Reg uses the class labels of $k-1$ data points in $\{\mathbf{x}_{i_j}\}_{j=2}^{k}$ to linearly reconstruct the class label of $\mathbf{x}_i$, while LL-Reg employs a linear transformation to map $\mathbf{x}_i$ directly to be a class label (for more details through Appendix A). Thus we see both LLE-Reg and LL-Reg use linear models to handle

[1] Available at http://www.kernel-machines.org/data
[2] Available at http://www.kyb.tuebingen.mpg.de/ssl-book/benchmarks.html
[3] Available at http://people.csail.mit.edu/jrennie/20Newsgroups/
[4] Available at http://www.ics.uci.edu/ mlearn/MLRepository.html

the neighboring data points. In contrast, LSR uses nonlinear function (spline) to map them.

Our algorithm has three parameters: the number of nearest neighbors k, the regularization parameter γ in Eq. (20) and the regularization parameter λ in Eq. (9). Formally, LLE-Reg and LL-Reg also have such three parameters. Differently, parameter λ in LLE-Reg and LL-Reg corresponds to that used to regularize the locally linear representation [37], [52].

GRF has two parameters: k and the parameter σ in Eq. (2). Besides, LLGC has a normalized regularization parameter $\alpha \in (0, 1)$ [56]. In experiments, it is fixed to be 0.99.

LapRLS has more parameters. In contrast to Problem (1), the objective function in LapRLS includes an additional kernel term related to all of the labeled data points. Besides k and the Gaussian kernel parameter σ, LapRLS has two global trade-off parameters γ_A and γ_I (see Eq. (4) in [6]). In experiments, we fix γ_A and γ_I both to be 0.001[5].

Totally 20 trials are run for each group of (k, γ, λ). In each trail, 10% of the data points are randomly selected, which are treated as labeled data points. The number of labeled dada points are listed in Table III. The final recognition rate is calculated as the average on these 20 trials.

Note that there is a common parameter σ in GRF, LLGC and LapRLS. Experiments show that directly setting it to be a fixed number is infeasible. This indicates that it should be changed from data set to data set. To determine a parameter σ from the data set, we first calculate the median value of the distances between all of the n data points, and denote it by d_m. Based on this value, we construct a candidate set $\{10^{-6}d_m, 10^{-4}d_m, 10^{-2}d_m, d_m, 10d_m, 10^2d_m\}$ from which a σ is selected. The cross validation approach is employed to achieve this goal. To this end, the labeled data points are divided into a few subsets. Specifically, for the labeled data points in USPS, Digital1, News20, Breast, Diabetes, and Vehicle data sets, they are divided into ten subsets, respectively. Since the number of the labeled data points in Iris, Ionosphere and Vote is very small, we only divide them into five subsets. When performing five-fold or ten-fold cross validation, we manually fix $k = 30$ in GRF, LLGC and LapRLS. The selected parameter σ will be finally used in experiments.

C. Experimental Results

Fig. 3 illustrates the experimental results with k =10, 15, $\cdots$, 50. From Fig. 3(a) to 3(i) are the results obtained from the nine data sets described in Section Table III. In experiments, γ is fixed to be 10000.0 and λ is fixed to be 0.0001. The results obtained with LL-Reg are not illustrated in Fig. 3. In our implementation, with the same parameter settings and data inputs, we obtain the same accuracy respectively with LL-Reg and LLE-Reg. This indicates that LL-Reg and LLE-Reg could be equivalent to each other [6]. As for this point, we give an explanation in Appendix A.

[5]Matlab code can be downloaded from http://manifold.cs.uchicago. edu /manifold_regularization /software.html.

[6]To avoid confusion, it should be emphasized here that the LLE-Reg used in [52] actually corresponds to the case of $\lambda = 0$ in Eq. (25). In our experiments, we take the same λ both in LLE-Reg and LL-Reg. Thus, the equivalence between LLE-Reg and LL-Reg holds only in the case of $\lambda > 0$.

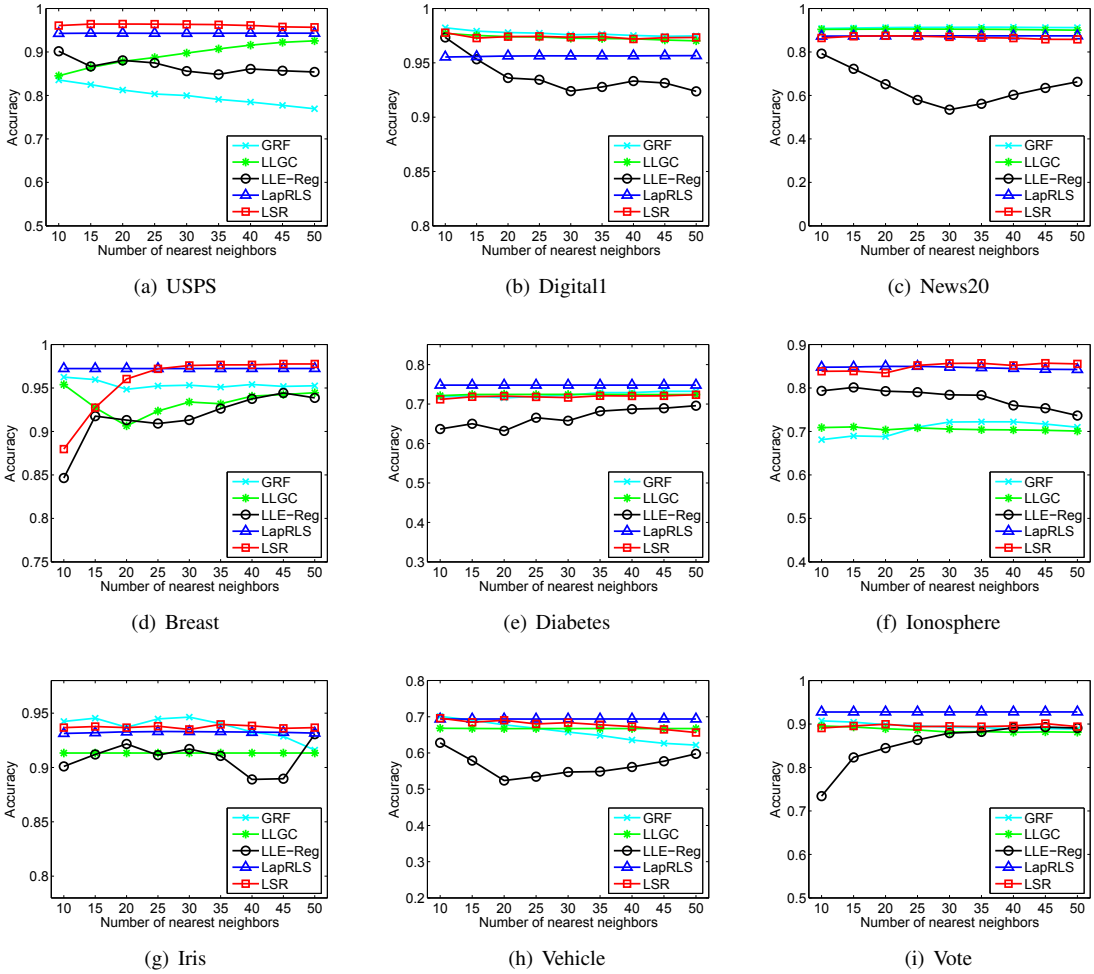
As can be seen from Fig. 3 in most experiments our algorithm achieves higher correct recognition rates, especially when compared with LLE-Reg. Both LLE-Reg and our algorithm are first derived the local evaluations from the same neighborhoods and all these local evaluations are then accumulated together to construct the global objective function. Differently, LSR employs splines to directly map neighboring data points to be their class labels, while LLE-Reg linearly reconstructs the class labels. The spline is a smooth nonlinear function in Sobolev space. Compared with linear functions, splines show more adaptability to the scattered data points. In contrast, LSR achieves higher accuracy in most experiments, compared with the classic semi-supervised algorithms of GRF and LLGC. In addition, the performance of our algorithm is also comparable to that of LapRLS, although its parameter σ has been tuned to data. Compared with the objective function in (1) where there are only two trade-off terms, the objective function in LapRLS is composed of more terms (see Eq. (4) in [6]). LapRLS shows satisfactory performance on real world data sets. In experiments, LapRLS achieves the highest accuracy on Diabetes and Vote data sets. Specifically, on Diabetes data set, the accuracy obtained with LapRLS is about 1.9% higher than that with LSR, while on Vote data set it is about 2.8% higher than that with LSR.

Fig. 4 illustrates the fact that the change of σ may significantly change the recognition accuracy. In experiments, we fix $k = 30$ and ran GRF, LLGC and LapRLS with $\sigma = 10^{-6}d_m, 10^{-4}d_m, 10^{-2}d_m, d_m, 10d_m$, and 10^2d_m, respectively. Note that when σ is very near to zero, LapRLS may generate very low accuracy, as witnessed in Digital1, Breast, Diabetes, Ionosphere, and Vote data sets in Fig. 4. Actually, in LapRLS, a small σ may lead the weight matrix of data to be zero. This will generate an ill-conditioned Laplacian matrix, whose elements may become infinite. In this case, LapRLS (code) outputs zero as the result.

In addition, the experimental results reported in Fig. 4 also show that the classification accuracy obtained with GRF, LLGC and LapRLS will decrease when σ is very large. Actually, in this case, the weights between neighbors will be close to one. Thus the affinities between neighbors are not well captured.

The parameter stability of γ and λ is also tested with experiments. Here only LLE-Reg is compared since formally it also has these two parameters. Fig. 5 reports the results of the nine data sets, with different γ, by fixing $k = 25$ and $\lambda = 0.0001$. The horizontal axis of each sub-figure in Fig. 5 stands for the exponent of γ. That is, the results are obtained with $\gamma = 10^2, 10^3, 10^4, 10^5$ and 10^6, respectively. As mentioned previously, the labeled values will be exactly satisfied when γ in (1) equals to positive infinity. Accordingly, we test it with large numbers. Fig. 5 indicates that in most experiments LSR achieves higher accuracy.

Fig. 6 reports the experimental results, with different λ. In experiments, we fix k to be 25 and γ to be 10000. The horizontal axis of each sub-figure in Fig. 6 stands for the exponent of λ. That is, the results are obtained with $\lambda = 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}$, and 10^{-2}. Note that the role of λ is very different from that of γ. As mentioned in

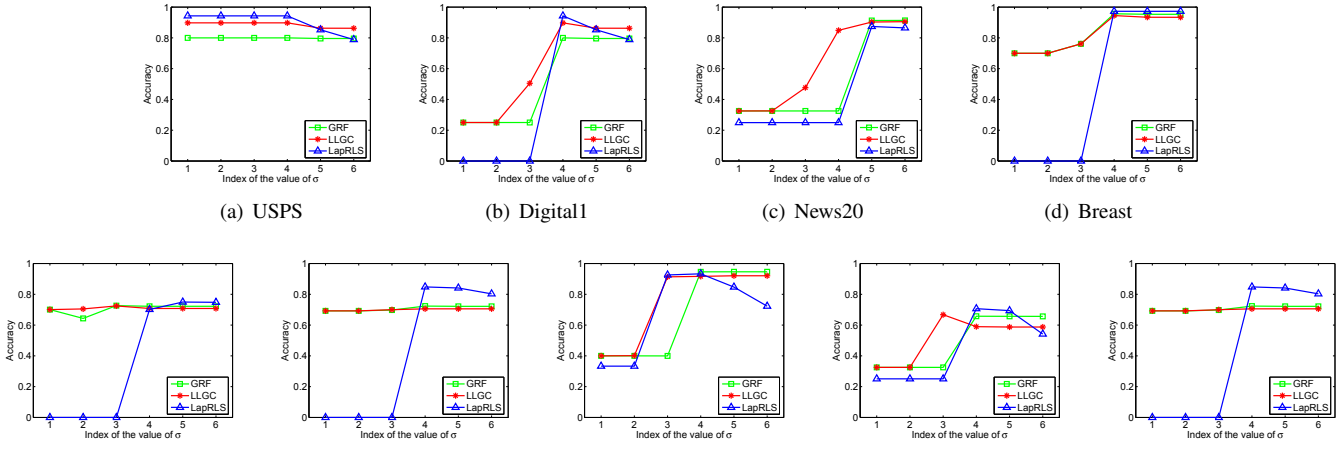


(a) USPS (b) Digital1 (c) News20

(d) Breast (e) Diabetes (f) Ionosphere

(g) Iris (h) Vehicle (i) Vote

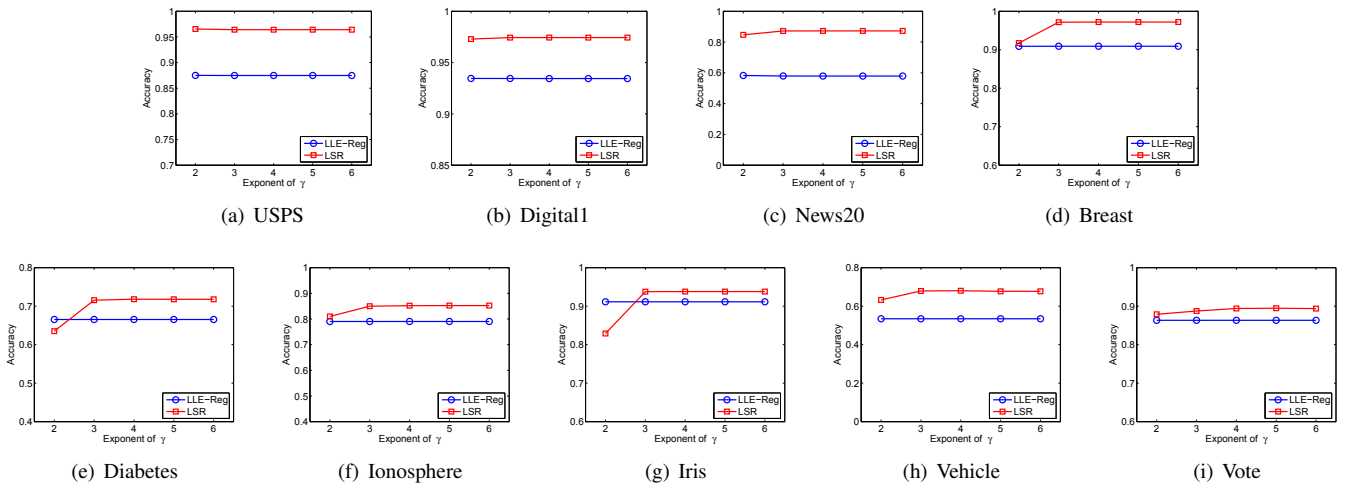
Fig. 3. Classification results with different parameter k.



(a) USPS (b) Digital1 (c) News20 (d) Breast

(e) Diabetes (f) Ionosphere (g) Iris (h) Vehicle (i) Vote

Fig. 4. Classification results with different parameter σ. The numbers in the horizontal axis are the indices of the values of σ. That is, the results are obtained with $\sigma = 10^{-6}d_m, 10^{-4}d_m, 10^{-2}d_m, d_m, 10d_m$, and $10^2 d_m$.



(a) USPS (b) Digital1 (c) News20 (d) Breast

(e) Diabetes (f) Ionosphere (g) Iris (h) Vehicle (i) Vote

Fig. 5. Classification results with different parameter γ. The numbers in the horizontal axis are the exponents of γ. That is, the results are obtained with $\gamma = 10^2, 10^3, 10^4, 10^5$ and 10^6.

previous sections, a smaller λ indicates that the conditions in Eq. (5) will be better satisfied. Thus we select to test λ with small numbers. Fig. 6 indicates that in most experiments LSR achieves higher accuracy.

Based on the above analyses on experimental results, here we summarize the performance of LSR as follows. First, the performance of LSR is comparable to those of GRF, LLGC and LapRLS, although the parameter σ is tuned to data. Actually, LSR outperforms GRF, LLGC and LapRLS in most data sets. Second, compared with locally linear methods, LSR employs smooth splines to capture the local information, which show more adaptability to the scattered neighboring data points. Finally, the experiment results illustrate that LSR is stable to the parameters k, γ and λ (see Figs. 3, 5, and 6). This will facilitate real world applications.

D. Computational Complexity

Now we analyze the computational complexity. RGF, LLGC, LLE-Reg, LL-Reg, LapRLS and LSR include the calculations of identifying the neighbors and solving the final linear equations. The main differences lie in the ways of modeling the data.

To identify k nearest neighbors for each data point in $\mathcal{X}$. The computational complexity is quadratic in the number of data points, namely, $O(n^2)$. Thus a large part of the computation time will be taken in this step.

Differently, given the k nearest neighbors and the data dimensionality m, the main computation in LSR is to construct matrix $\mathbf{M}_i$ in Eq. (11). The computational complexity will be up to about $O((k+d+1)^3)$, due to the calculation of the inverse matrix of the coefficient matrix in Eq. (9). In LLE-Reg, it will be about $O(nm^3 + mk^2)$, and for LL-Reg, it is about $O(nm^3 + mk^2 + 2k^2)$. Thus our algorithm is slower than the linear ones, especially when the size of the primitive polynomials d is very large. However, in implementation, since usually k is a small number, we can first use principal component analysis to locally reduce the dimensionality of the k neighboring data points. In contrast, GRF and LLGC will take less time to construct $\mathbf{M}$ since these two algorithms construct it directly via the similarity between pairs of neighboring data points. The computational complexity is about $O(nmk^2))$. Besides the same computation as performed by LLGC, LapRLS has an additional task, namely, constructing the kernel matrix related to all of the l labeled data points. Thus, the computation complexity in model training is about $O(nmk^2 + lmk^2)$. After the model is trained, LapRLS needs it to predict the $n-l$ unlabeled data points, and the complexity is about $O(lm(n-l) + l(n-l))$.

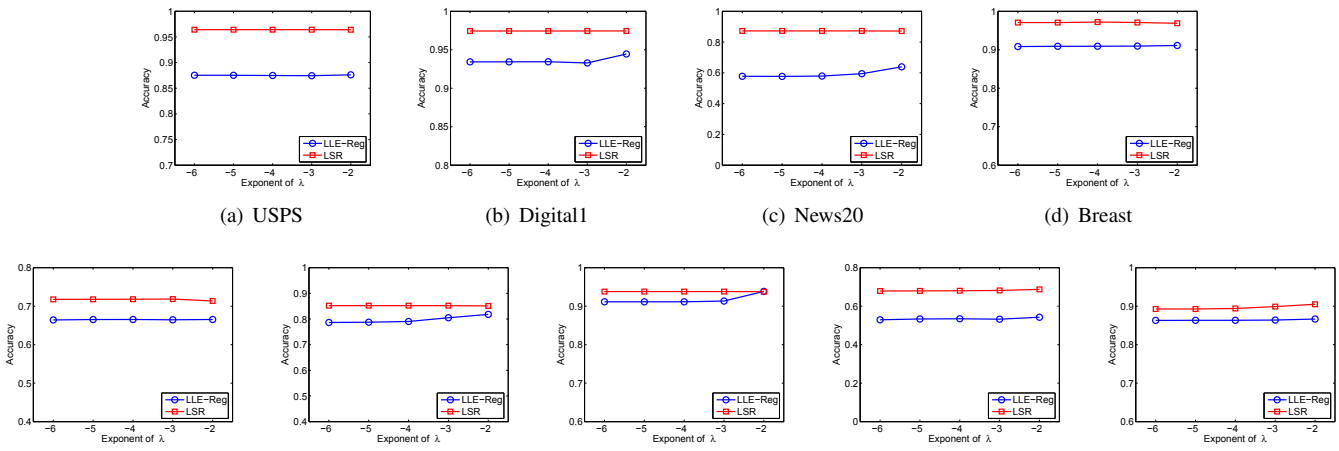
V. Applications to Interactive Image Segmentation and Image Matting

Segmenting foreground objects from natural images is a fundamental task in image understanding. But it is a challenging problem. The difficulties lie in the high complexity in visual pattern modeling and the intrinsic ambiguity in visual pattern grouping. To make the problems solvable, one method is to construct interactive frameworks, and allow the user to specify the foreground objects and the background. In view of machine learning, the user specifications about the image provide valuable supervised information to learn to group the visual patterns.

Some interactive segmentation frameworks are developed [12], [8], [36], [31], among which one popular user-computer interaction style is to draw zigzag lines on the image to label the foreground and background regions. Then the task is to infer the class labels of those unlabeled pixels. Such a task can be addressed in way of semi-supervised learning.

A. Image Segmentation

In this subsection, our task is to segment the foreground object from the background. In the semi-supervised classification setting, we assign class label "+1" to each of the user-specified foreground pixels and "-1" to each of the user-specified background pixels. Each pixel p is described as a 3-dimensional vector, i.e., $\mathbf{x}_p = [r, g, b]^T$, in which (r, g, b) is the normalized color of pixel p. For each pixel p, those pixels



(a) USPS (b) Digital1 (c) News20 (d) Breast

(e) Diabetes (f) Ionosphere (g) Iris (h) Vehicle (i) Vote

Fig. 6. Classification results with different parameter λ. The numbers in the horizontal axis are the exponents of λ. That is, the results are obtained with $\lambda = 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}$ and 10^{-2}.

in the 3×3 local window with p as the center are employed as its neighbors. In experiments, we fix γ to be 10000 and λ to be 0.0001. The parameter σ in GRF and LLGC is manually set to be 0.1. Here we do not employ the median value of the distances between pixels since it may equal to zero (due to the same color of many pixels in image). The parameter α in LLGC is also set to be 0.99. We will not conduct the experiments with LapRLS since in our implementation we face a problem of "out of memory" in Matlab work setting.

For comparisons, the segmentation results with Graph Cut (GC) are also reported. Here the algorithm in [12] is implemented[7]. In experiments, the data likelihoods of R_p("bkg") and R_p("obj") in [12] are calculated by Eq. (2) in [31], where K-means clustering algorithm is used to cluster respectively the user specified foreground and background pixels. When running the K-means clustering algorithm, the number of clusters is manually set to be 20.

Fig. 7 illustrates the experimental results of ten natural images in Berkeley image database [32]. In Fig. 7, the second column shows the user specified strokes about the foreground and background. From the third to the seventh column are the results obtained by GC, GRF, LLGC, LLE-Reg and our LSR. All these segmentations are obtained with the same strokes as labeled in the second column. For comparisons, the last column lists the ground truth obtained by hand. Fig. 8 illustrates the results of ten natural images in Grabcut database [36]. As can be seen from Fig. 7 and Fig. 8, satisfactory segmentations are obtained with our algorithm.

Table IV gives a quantitative comparison between the algorithms. The first column in Table IV lists the indices of the images in Fig. 7 and Fig. 8. For examples, the first row in Table IV corresponds to the first image in Fig. 7 and the 11-th row corresponds to the first image in Fig. 8. The number in Table IV stands for the classification rate, which is calculated as the ratio (percent) of the number of correctly classified pixels to that of the total pixels in the image. Here the correctly classified pixels are identified by taking the ground truth as a reference. As can be seen in Table IV, in most images our algorithm achieves higher accuracy.

[7]The source codes are downloaded from the author's homepage and run here for image segmentation

No.	GC	GRF	LLGC	LLE-Reg	LSR
1	98.7223	82.9950	80.1898	90.4293	**98.9544**
2	97.5657	97.3715	97.4664	98.4609	**99.1005**
3	99.5853	95.1008	94.5429	95.8499	**99.6130**
4	94.6890	94.8379	96.1478	96.4836	**98.8040**
5	98.3294	83.3440	87.9527	94.7313	**98.6361**
6	97.5781	95.4473	95.0599	95.3418	**99.0527**
7	98.4112	94.4305	91.7859	92.6855	**98.9953**
8	99.0552	98.8216	99.2903	99.5488	**99.6875**
9	97.9030	98.2331	98.1849	97.5496	**98.5587**
10	97.1354	98.9414	98.9362	99.0970	**99.5104**
11	**99.2591**	98.6797	96.6003	97.6719	99.2240
12	98.0794	99.2969	97.2604	98.8464	**99.6068**
13	98.4222	97.5896	98.4830	98.3911	**99.0148**
14	98.0374	97.8680	97.5204	97.3321	**98.4828**
15	99.1185	89.3363	90.8044	94.3763	**99.4504**
16	**99.7357**	93.0938	91.4596	94.3763	99.6797
17	99.5230	96.6444	97.9719	93.6133	**99.6622**
18	97.4030	97.3244	97.9037	94.9274	**99.5630**
19	99.3170	96.9319	98.8637	98.9985	**99.6741**
20	97.6171	91.8421	95.1742	98.9578	**99.8075**

TABLE IV
THE CLASSIFICATION RATE (%) OF THE 20 IMAGES IN FIG. 7 AND FIG. 8, ORDERED WITH INDICES IN THE FIRST COLUMN. THE FIRST TEN ROWS CORRESPOND TO THE IMAGES IN FIG. 7, AND THE LAST TEN ROWS CORRESPOND TO THE IMAGES IN FIG. 8.

B. Image Matting

The task in image segmentations is to assign each pixel to either the foreground or the background. Another task in image editing is the so-called image matting. In this work setting, an input image is assumed to be a composite of a foreground image and a background image. The task is to infer the foreground opacity from the image. In this way, we hope to generate soft edges between the background and the foreground objects.

Some image matting algorithms have been developed. Typical algorithms include Poisson matting [40], random

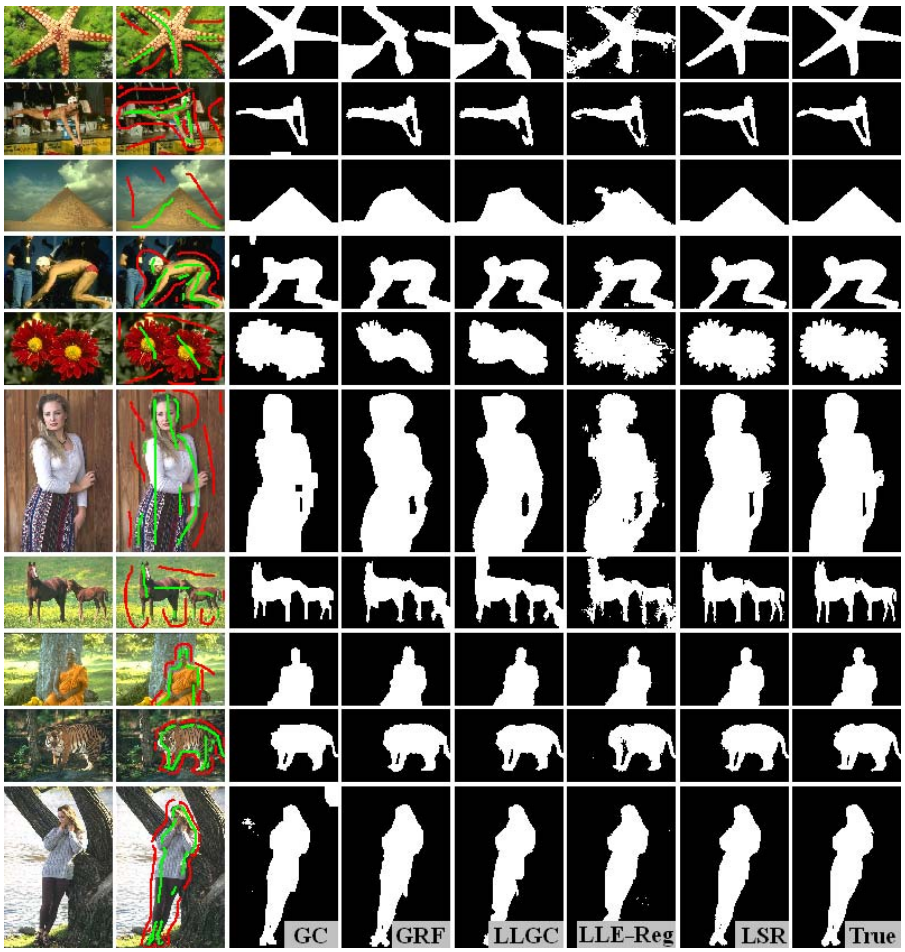


Fig. 7. Segmentation results of the images from Berkeley image database. The first and the second column are the source images and the user specified strokes about the foreground and background. From the third to the seventh column are the results obtained by GC, GRF, LLGC, LLE-Reg and LSR. The last column lists the ground truth for comparison.

walks [22], iterative matting via Belief Propagation (BP) [48], Matting Laplacian (ML) [29], and the unsupervised spectral matting [30]. Here we extend our algorithm for image matting. To this end, we no longer assign "-1" to each of the user specified background pixels, but assign "0" to each of them. We also assign "+1" to each of the user specified foreground pixels. In this way, we hope to obtain a soft label to each pixel which is limited in [0, 1]. Finally, after the linear equations in Eq. (17) are solved, $\mathbf{f}$ is directly treated as the foreground opacity. That is, the Step 2(c) of the algorithm in Table II will not be performed.

We compared our algorithm with BP, ML and LLE-Reg[8]. To compare them with Poisson matting and random walks, one can refer to [29]. In addition, we will not report the results with GRF and LLGC as none of them generates correct results in our implementation. The same parameters used in Fig. 7 and Fig. 8 are employed here in experiments. Figs. 9, 10 and 11 report the experimental results (due to different sizes of images, here we report the results in three figures). The images are taken from [48]. In each panel in Figs. 9, 10 and 11, the first is the source image and the second is the user specified strokes about the foreground and background. From the third to the last are the results obtained by BP, ML, LLE-Reg and our algorithm. As can be seen, satisfactory results are obtained with our algorithm. Note that BP also generates satisfactory results. In contrast, our algorithm explores more details of the foreground objects, as witnessed from the hairs of the girls in Fig. 9 and the feather of the peacock in Fig. 10. ML is a famous image matting approach. Experimental results show that the performance of our algorithm is comparable to that of ML. For examples, in the third girl image in Fig. 9 and in the first image in Fig. 10, some of the pixels in the rectangle regions are over highly assigned as the foreground. This is attenuated to some degree in our algorithm.

Finally, as a summary of this Section, we point out that GRF, LLGC, LLE-Reg and LSR are all developed to solve the problem of semi-supervised classification as formulated in (1). In contrast, they use different models to exploit the relations between neighboring data points. Specifically, both GRF and LLGC are developed via conventional graph construction, while LLE-Reg and LSR are developed via local

[8]We downloaded the software of BP and the codes of ML from the authors' homepages. When running the software of BP, the user specified strokes about the foreground and background are supplied via the human-computer interface with mouse, according to those used in ML, LLE-Reg, and LSR (see Figs. 9, 10, 11).

Fig. 8. Segmentation results of the images from Grabcut database. The first and the second column are the source images and the user specified strokes about the foreground and background. From the third to the seventh column are the results obtained by GC, GRF, LLGC, LLE-Reg and LSR. The last column lists the ground truth for comparison.

label inference. Furthermore, LLE-Reg is locally linear, and LSR is locally non-linear. Comparative experiments show that LSR achieves better segmentations in most images (see Figs. 7-11). This indicates that, in LSR, it is the proposed graph construction in itself that is superior to the conventional graph construction.

Although our algorithm can generate satisfactory results, it needs more time to finish the calculation. For example, for images with 320×214 pixels, our algorithm takes about 250 seconds on a PC with 2.4Ghz CPU and 1GB RAM, using Matlab 7.0, while GRF, LLGC and LLE-Reg need about 200, 62, 70 seconds, respectively. In contrast, ML takes about 163 seconds, while BP only needs about several seconds. One future work is to develop fast version of our algorithm for image segmentation and image matting.

VI. Conclusion

In this work we have developed the semi-supervised classification algorithm with local spline regression. Splines developed in Sobolev space are used to model each of the neighborhoods. Finally, we analyze the regularized loss on each neighborhood and formulate it in terms of class label vector of the neighboring data points. The losses on all of the neighborhoods are finally accumulated together. We addressed our algorithm into the Laplacian regularization framework, from which a globally optimal classification is obtained. Comparative classification experiments on many public data sets and applications to interactive natural image segmentation and image matting show the validity of our algorithm. In the future, we would like to research how to speed up the computation of our algorithm, especially when it is applied to image segmentation and image matting.

Fig. 9. Matting results (Group I). In each panel, the first and the second column are the source image and the user specified strokes about the foreground and background. From the third to the sixth column are the results obtained by BP, ML, LLE-Reg and LSR.

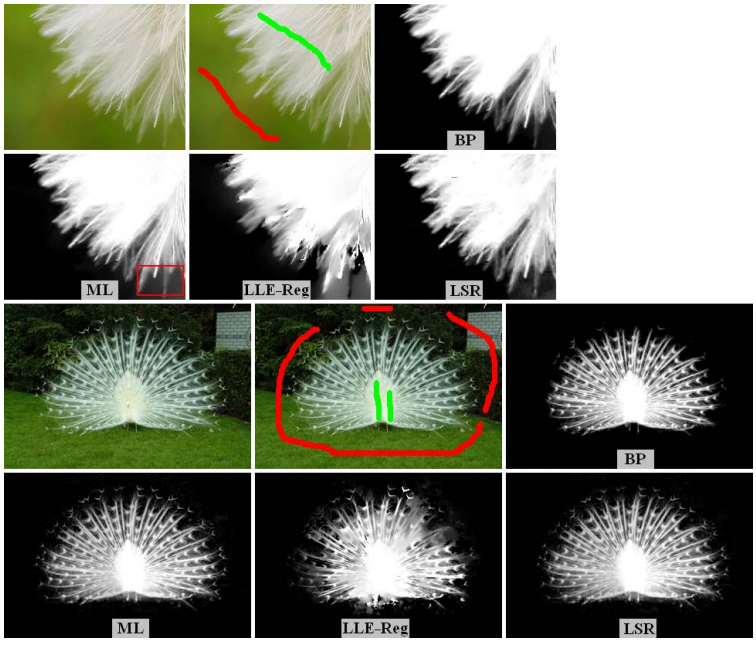


Fig. 10. Matting results (Group II). In each panel, the first and the second column are the source image and the user specified strokes about the foreground and background. From the third to the sixth column are the results obtained by BP, ML LLE-Reg and LSR.

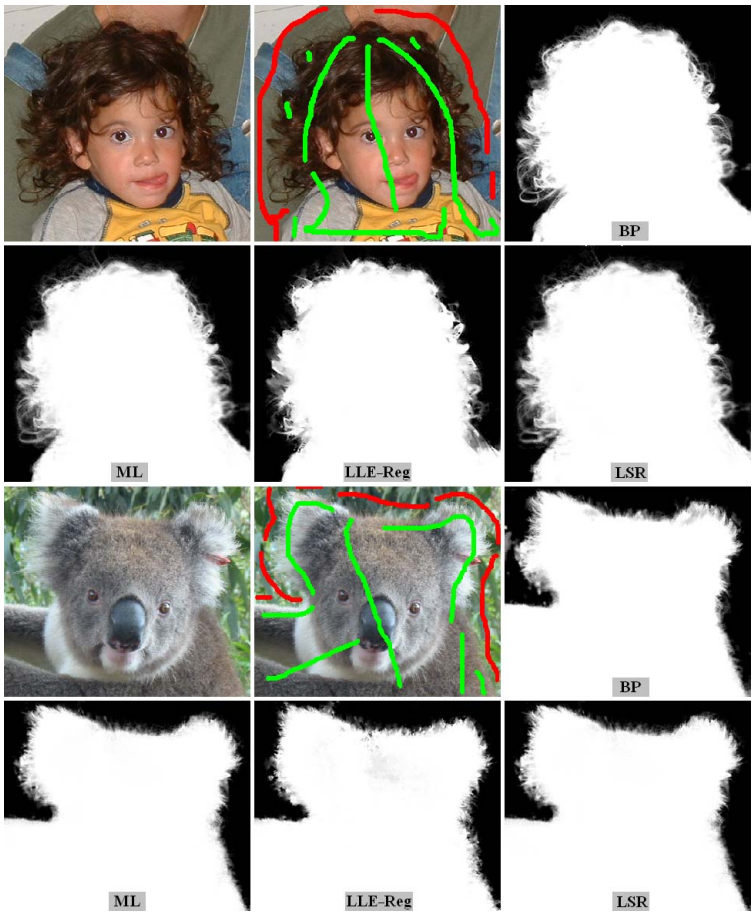


Fig. 11. Matting results (Group III). In each panel, the first and the second column are the source image and the user specified strokes about the foreground and background. From the third to the sixth column are the results obtained by BP, ML, LLE-Reg and LSR.

APPENDIX A
EQUIVALENCE BETWEEN LLE-REG AND LL-REG

LL-Reg and LLE-Reg are initially developed with different motivations and formulations. Specifically, LL-Reg is developed on locally linear transformations, while LLE-Reg is developed on locally linear reconstructions. Here we illustrate the fact that they are mathematically equivalent to each other.

Suppose we are given $\mathbf{x}_i$ and its k neighboring data points $\{\mathbf{x}_i\}_{i=1}^k$, including $\mathbf{x}_i$ itself, with $\mathbf{x}_i = \mathbf{x}_{i_1}$. In LLE-Reg, $\mathbf{x}_i$ is linearly reconstructed with its $k-1$ neighbors, namely,

$$\mathbf{x}_i \approx \sum\nolimits_{j=2}^{k} w_j \mathbf{x}_{i_j}, \tag{21}$$

where w_j, $j = 2, 3, \cdots, k$, are reconstruction weights, and $\sum_{j=2}^{k} w_j = 1$. After the optimal weights are evaluated, the class label f_i of $\mathbf{x}_i$ is then predicted as

$$f_i = \sum\nolimits_{j=2}^{k} w_j f_{i_j}. \tag{22}$$

In LL-Reg, a linear transformation is introduced to directly map the data point to be a class label:

$$f = \mathbf{u}^T(\mathbf{x} - \mathbf{x}_i) + b, \tag{23}$$

where $\mathbf{u} \in \mathbb{R}^m$ is a project vector and $b \in \mathbb{R}$ is a bias. Under the regularized least squares framework, the optimal $\mathbf{u}$ and b can be evaluated with $k-1$ data pairs $\{(\mathbf{x}_{i_j}, f_{i_j})\}_{j=2}^k$. Now it can be easily justified that if (23) is used to map $\mathbf{x}_i$, then

$$f_i = b. \tag{24}$$

To illustrate the equivalence between LLE-Reg and LL-Reg, we first give two Lemmas. For the optimal weights in (21), we have the following Lemma:

Lemma 1 Let $\mathbf{X} = [\mathbf{x}_{i_2} - \mathbf{x}_i, ..., \mathbf{x}_{i_k} - \mathbf{x}_i] \in \mathbb{R}^{m\times(k-1)}$, $\mathbf{w} = [w_2, w_3, ..., w_k]^T \in \mathbb{R}^{k-1}$, and $\mathbf{e} = [1, 1, ..., 1]^T \in \mathbb{R}^{k-1}$. Then [37]

$$\mathbf{w} = \frac{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}{\mathbf{e}^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}.$$

Proof: Based on Eq. (21), it can be easily justified that the sum of squared distances from $\mathbf{x}_i$ to its $k-1$ neighbors equals to $\mathbf{w}^T(\mathbf{X}^T\mathbf{X})\mathbf{w}$. Using the Lagrange multiplier method and considering the equality constraint $\sum_{j=2}^{k} w_j = 1$, we have the following Lagrange function:

$$L(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T(\mathbf{X}^T\mathbf{X})\mathbf{w} - \eta(\mathbf{w}^T\mathbf{e} - 1),$$

where η is the Lagrange multiplier.

Differentiating the objective function with respect to $\mathbf{w}$ and setting the derivative to be zero, we can obtain $\mathbf{w} = \eta(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}$. Then $\mathbf{w}^T\mathbf{e} = 1 \Rightarrow (\eta(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e})^T\mathbf{e} = 1 \Rightarrow \eta = \frac{1}{\mathbf{e}^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}$. Thus $\mathbf{w} = \frac{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}{\mathbf{e}^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}$. ■

In computation, $\mathbf{X}^T\mathbf{X}$ may be singular. To avoid singularity, a regularization term $\lambda\mathbf{I} \in \mathbb{R}^{(k-1)\times(k-1)}$ can be added to it. Then we have [37]

$$\mathbf{w} = \frac{(\lambda\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}{\mathbf{e}^T(\lambda\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}, \tag{25}$$

Now we consider LL-Reg [52]. We have the following Lemma:

Lemma 2 Let the bias b in (23) be the linear combination of the class labels, namely, $b = \alpha^T\mathbf{f}$, then [52]

$$\alpha = \frac{\mathbf{e}^T - \mathbf{e}^T\mathbf{X}^T\mathbf{X}(\lambda\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}}{(k-1) - \mathbf{e}^T\mathbf{X}^T\mathbf{X}(\lambda\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}. \tag{26}$$

Proof: Consider the follows objective function [52]:

$$G(\mathbf{u}, b) = \lambda\mathbf{u}^T\mathbf{u} + (\mathbf{X}^T\mathbf{u} + b\mathbf{e} - \mathbf{f})^T(\mathbf{X}^T\mathbf{u} + b\mathbf{e} - \mathbf{f}),$$

where $\mathbf{f} = [f_{i_2}, \cdots, f_{i_k}]^T \in \mathbb{R}^{k-1}$ collects the class labels of the $k-1$ neighbors. Taking $\partial G(\mathbf{w}, b)/\partial\mathbf{w} = 0$ and after performing simple algebra operations, we can obtain $\mathbf{u} = (\lambda\mathbf{I}+\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}(\mathbf{f}-b\mathbf{e})$. Similarly, let $\partial G(\mathbf{w}, b)/\partial b = 0$, we can get $b = \frac{1}{k-1}(\mathbf{e}^T\mathbf{f} - e^T\mathbf{X}^T\mathbf{u})$.

Substituting $\mathbf{u}$ to b, it follows

$$b = \frac{\mathbf{e}^T - \mathbf{e}^T\mathbf{X}^T(\lambda\mathbf{I} + \mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}}{(k-1) - \mathbf{e}^T\mathbf{X}^T(\lambda\mathbf{I} + \mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{e}}\mathbf{f}. \tag{27}$$

Note that $\mathbf{X}^T(\lambda\mathbf{I} + \mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X} = \mathbf{X}^T\mathbf{X}(\lambda\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}$, then (26) holds naturally. ■

Theorem 4 It holds that $\alpha = \mathbf{w}$.

Proof: Note that $\mathbf{e}^T\mathbf{e} = k-1$, then it follows

$$\begin{aligned}
\alpha^T &= \tfrac{\mathbf{e}^T - \mathbf{e}^T\mathbf{X}^T\mathbf{X}(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}}{(k-1)-\mathbf{e}^T\mathbf{X}^T\mathbf{X}(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}} \\
&= \tfrac{\mathbf{e}^T(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1} - \mathbf{e}^T\mathbf{X}^T\mathbf{X}(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}}{\mathbf{e}^T\mathbf{e}-\mathbf{e}^T\mathbf{X}^T\mathbf{X}(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}} \\
&= \tfrac{\mathbf{e}^T(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1} - \mathbf{e}^T\mathbf{X}^T\mathbf{X}(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}}{\mathbf{e}^T(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}-\mathbf{e}^T\mathbf{X}^T\mathbf{X}(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}} \\
&= \tfrac{\mathbf{e}^T(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X}-\mathbf{X}^T\mathbf{X})(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}}{\mathbf{e}^T(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X}-\mathbf{X}^T\mathbf{X})(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}} \\
&= \tfrac{\mathbf{e}^T(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}}{\mathbf{e}^T(\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X})^{-1}\mathbf{e}}.
\end{aligned}$$

Note that $\lambda\mathbf{I}+\mathbf{X}^T\mathbf{X}$ is symmetrical. Thus its inverse matrix is also symmetrical. Based on (26), $\alpha = \mathbf{w}$ holds naturally. ■

This theorem indicates that α in LL-Reg exactly equals to $\mathbf{w}$ in LLE-Reg. As a result, in each neighborhood we can get the same errors about the class labels, with LLE-Reg and LL-Reg. Then we can obtain the same global error if the errors evaluated in local neighborhoods are accumulated together. In terms of matrices, this global error can be formally formulated as $\mathbf{f}^T\mathbf{M}\mathbf{f}$. In this way, a matrix $\mathbf{M}$ is constructed, which is finally used to replace the matrix $\mathbf{M}$ in Problem (1). This explains why the same results with LLE-Reg and LL-Reg are obtained in the experiments.

Finally, it should be emphasized that, only in the case of "$\lambda > 0$" and given the same λ, we can say LLE-Reg and LL-Reg are equivalent to each other. To avoid confusion, here we point out that the LLE-Reg used in [52] actually corresponds to the case of "$\lambda = 0$". That is, no regularization term is employed when developing the original LLE-Reg in [52].

ACKNOWLEDGMENT

This work is supported by the Projects (Grant No. 60975037 and 60721003) of the National Natural Science Foundation of China.

REFERENCES

[1] R. A. Adams, *Sobolev spaces*. Boston, MA, USA: Academic Press, 1975.

[2] R. K. Ando and T. Zhang, "Two-view feature generation model for semisupervised learning," in *International Conference on Machine Learning*, Corvallis, Oregon, USA, 2007, pp. 25–32.

[3] M. F. Balcan, A. Blum, and K. Yang, "Co-training and expansion: Towards bridging theory and practice," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2004, pp. 89–96.

[4] M. Belkin, I. Matveeva, and P. Niyogi, "Regularization and semi-supervised learning on large graphs," in *International Conference on Learning Theory*, Banff, Canada, 2004, pp. 624–638.

[5] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems 14*. Cambridge, MA, USA: MIT Press, 2002, pp. 585–591.

[6] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *Journal of Machine Learning Research*, vol. 7, pp. 2399–2434, 2006.

[7] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation.*, vol. 15, no. 6, pp. 1373–1396, 2003.

[8] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr, "Interactive image segmentation using an adaptive gmmrf model," in *European Conference on Computer Vision*, Prague, Czech, 2004, pp. 428–441.

[9] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," in *Proceedings of International Conference of Machine Learning*, Williamstown, MA, USA, 2001, pp. 19–26.

[10] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *In Annual Conference on Computational Learning Theory*, Madison, Wisconsin, USA, 1998, pp. 92–100.

[11] F. Bookstein, "Principal warps: thin-plate splines and the decomposition of deformations," *IEEE Transactions on Pattern Analysis and Machine Learning*, vol. 11, no. 6, pp. 567–585, 1989.

[12] Y. Y. Boykov and M. P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images," in *International conference on Computer Vision*, Vancouver, Canada, 2001, pp. 105–112.

[13] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. MIT Press, 2006.

[14] N. V. Chawla and G. Karakoulas, "Learning from labeled and unlabeled data: An empirical study across techniques and domains," *Journal of Artificial Intelligence Research*, vol. 23, pp. 331–366, 2005.

[15] F. G. Cozman, I. Cohen, and M. C. Cirelo, "Semi-supervised learning of mixture models," in *International Conference on Machine Learning*, Washington, DC, USA, 2003, pp. 99–106.

[16] S. Dasgupta, M. L. Littman, and D. McAllester, "Pac generalization bounds for co-training," in *Advances in Neural Information Processing Systems 14*, Vancouver, Canada, 2001.

[17] J. Duchon, "Splines minimizing rotation-invariant semi-norms in sobolev spaces," in *Constructive Theory of Functions of Several Variables*, A. Dold and B. Eckmann, Eds. Springer-Verlag, 1977, pp. 85–100.

[18] A. Fujino, N. Ueda, and K. Saito, "A hybrid generative/discriminative approach semi-supervised classifier design," in *AAAI*, Pittsburgh, Pennsylvania, USA, 2005, pp. 764–769.

[19] G. Getz, N. Shental, and E. Domany, "Semi-supervised learning – a statistical physics approach," in *ICML Workshop on Learning with Partially Classified Training Data*, Bonn, Germany, 2005.

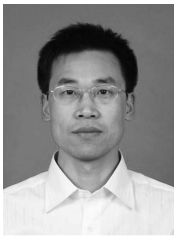
[20] S. Goldman and Y. Zhou, "Enhancing supervised learning with unlabeled data," in *International Conference on Machine Learning*, San Francisco, CA, USA, 2000, pp. 327–334.

[21] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: The Johns Hopkins University Press, 1996.

[22] L. Grady, T. Schiwietz, S. Aharon, and R. Westermann, "Random walks for interactive alpha-matting," in *Proceedings of Fifth IASTED International Conference on Visualization, Imaging and Image Processing*, Benidorm, Spain, 2005, pp. 423–429.

[23] X. He and P. Niyogi, "Locality preserving projections," in *Proceedings of the Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 2003.

[24] T. Joachims, "Transductive learning via spectral graph partitioning," in *Proceedings of International Conference of Machine Learning*, Washington, DC, USA, 2003, pp. 290–297.
[25] T. Joachims, "Transductive inference for text classification using support vector machines," in *International Conference on Machine Learning*, Bled, Slovenia, 1999, pp. 200–209.
[26] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed. New York, USA: Springer, 2002.
[27] R. Jones, "Learning to extract entities from labeled and unlabeled text," Carnegie Mellon University, Tech. Rep. CMU-LTI-05-191, 2005.
[28] N. D. Lawrence and M. I. Jordan, "Semi-supervised learning via gaussian processes," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2004, pp. 753–760.
[29] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 228–242, 2008.
[30] A. Levin, A. Rav-Acha, and D. Lischinski, "Spectral matting," in *International conference on Computer Vision and Pattern Recognition*, Minneapolis, Minnesota, USA, 2007, pp. 1–8.
[31] Y. Li, J. Sun, C. Tang, and H. Shum, "Lazy snapping," in *SIGGRAPH*, Los Angeles, USA, 2004, pp. 303–308.
[32] D. R. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *IEEE International Conference on Computer Vision*, Vancouver, Canada, 2001, pp. 416–425.
[33] D. J. Miller and H. S. Uyar, "A mixture of experts classifier with learning based on both labelled and unlabelled data," in *Advances in Neural Information Processing Systems 9*, Vancouver, Canada, 1996, pp. 571–577.
[34] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," in *International Conference on Information and Knowledge Management*, McLean, VA, USA, 2000, pp. 86–93.
[35] K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine Learning.*, vol. 39, no. 2-3, pp. 103–134, 2000.
[36] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut – interactive foreground extraction using iterated graph cuts," in *SIGGRAPH*, Los Angeles, USA, 2004, pp. 309–314.
[37] L. K. Saul and S. T. Roweis, "Thinking globally, fit locally: unsupervised learning of low dimensional manifolds," *Journal of Machine Learning Research*, vol. 4, pp. 119–155, 2003.
[38] B. Shahshahani and D. Landgrebe, "The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, no. 5, pp. 1087–1095, 1994.
[39] J. B. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
[40] J. Sun, J. Jia, C. K. Tang, and H.-Y. Shum, "Poisson matting," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 315–321, 2004.
[41] M. Szummer and T. Jaakkola, "Partially labeled classification with markov random walks," in *Advances in Neural Information Processing Systems 14*, Vancouver, Canada, 2001, pp. 945–952.
[42] G. Wahba, *Spline models for observational data*. SIAM Press, 1990.
[43] C. Walder and O. Chapelle, "Learning with transformation invariant kernels," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2007, pp. 1561–1568.
[44] F. Wang, S. J. Wang, C. S. Zhang, and O. Winther, "Semi-supervised mean fields," in *International Conference on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 2007, pp. 596–603.
[45] F. Wang, T. Li, G. Wang, and C. Zhang, "Semi-supervised classification using local and global regularization," in *The 23rd AAAI Conference on Artificial Intelligence*, Chicago, Illinois, USA, 2008, pp. 726–731.
[46] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 1, pp. 55–67, 2008.
[47] F. Wang and C. Zhang, "On discriminative semi-supervised classification," in *The 23rd AAAI Conference on Artificial Intelligence*, Chicago, Illinois, USA, 2008, pp. 720–725.
[48] J. Wang and M. F. Cohen, "An iterative optimization approach for unified image segmentation and matting," in *International conference on Computer Vision*, Beijing, China, 2005, pp. 936–943.
[49] H. Wendland, *Scattered Data Approximation*. Cambridge, UK: Cambridge University Press, 2005.
[50] M. Wu, K. Yu, S. Yu, and B. Schölkopf, "Local learning projections," in *International Conference on Machine Learning*, Corvallis, Oregon, USA, 2007, pp. 1039–1046.
[51] M. R. Wu and B. Schölkopf, "A local learning approach for clustering," in *Advances in Neural Information Processing Systems 19*. Cambridge, MA, USA: MIT Press, 2007, pp. 1529–1536.
[52] M. R. Wu and B. Schölkopf, "Transductive classification via local learning regularization," in *International Conference on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 2007, pp. 628–635.
[53] S. M. Xiang, F. P. Nie, C. S. Zhang, and C. X. Zhang, "Spline embedding for nonlinear dimensionality reduction," in *European Conference on Machine Learning*, Berlin, Germany, 2006, pp. 825–832.
[54] J. Yoon, "Spectral approximation orders of radial basis function interpolation on the sobolev space," *SIAM Journal on Mathematical Analysis.*, vol. 33, no. 4, pp. 946–958, 2001.
[55] Z. Zhang and H. Zha, "Principal manifolds and nonlinear dimensionality reduction via tangent space alignment," *SIAM Journal on Scientific Computing.*, vol. 26, no. 1, pp. 313–338, 2004.
[56] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in Neural Information Processing Systems 16*, Vancouver, Canada, 2003, pp. 321–328.
[57] Z. H. Zhou and M. Li, "Tri-training: exploiting unlabeled data using three classifiers," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 11, pp. 1529–1541, 2005.
[58] Z. H. Zhou, D. C. Zhan, and Q. Yang, "Semi-supervised learning with very few labeled training examples," in *AAAI*, Vancouver, Canada, 2007, pp. 675–680.
[59] X. J. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of International Conference of Machine Learning*, Washington DC, USA, 2003, pp. 912–919.
[60] X. J. Zhu, "Semi-supervised learning literature survey," University of Wisconsin-Madison, Tech. Rep. Computer Sciences TR 1530, 2007.



Shiming Xiang received the B.S. degree in mathematics from Chongqing Normal University, China, in 1993, the M.S. degree from Chongqing University, China, in 1996, and the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, China, in 2004. From 1996 to 2001, he was a Lecturer with the Huazhong University of Science and Technology, Wuhan, China. He was a postdoctorate with the Department of Automation, Tsinghua University, Beijing, China, until 2006. He is currently an Associate Professor with the Institute of Automation, Chinese Academy of Sciences. His interests include pattern recognition and machine learning.

Feiping Nie received the B.S. degree in computer science from the North China University of Water Conservancy and Electric Power in 2000, the M.S. degree from Lanzhou University, China, in 2003, and the Ph.D. degree from the Department of Automation, Tsinghua University, China, in 2009. His research interests focus on machine learning and its applications.

Changshui Zhang (M′ 02) received the B.S. degree in mathematics from Peking University, Beijing, China, in 1986, and the Ph.D. degree from Tsinghua University, Beijing, in 1992. In 1992, he joined the Department of Automation, Tsinghua University, and is currently a Professor. His interests include pattern recognition, machine learning, etc. He has authored over 200 papers. Dr. Zhang currently serves on the editorial board of the Pattern Recognition journal.