Front. Electr. Electron. Eng. China 2011, 6(1): 17–26
DOI 10.1007/s11460-011-0130-6

RESEARCH ARTICLE

Changshui ZHANG, Fei WANG

Graph-based semi-supervised learning

Abstract The recent years have witnessed a surge of interests in graph-based semi-supervised learning (GBSSL). In this paper, we will introduce a series of works done by our group on this topic including: 1) a method called linear neighborhood propagation (LNP) which can automatically construct the optimal graph; 2) a novel multilevel scheme to make our algorithm scalable for large data sets; 3) a generalized point charge scheme for GBSSL; 4) a multilabel GBSSL method by solving a Sylvester equation; 5) an information fusion framework for GBSSL; and 6) an application of GBSSL on fMRI image segmentation.

Keywords graph-based semi-supervised learning (GBSSL), linear neighborhood propagation (LNP), point charge model, fMRI image segmentation

1 Introduction

Semi-supervised learning (SSL), which aims at learning from labeled and unlabeled data, has aroused considerable interests in data mining and machine learning fields since it is usually hard to collect enough labeled data points in practical applications. Various SSL methods have been proposed in recent years, and they have been applied to a wide range of areas including text categorization, computer vision, and bioinformatics (see Refs. [1,2] for recent reviews). Moreover, it has been shown recently that the significance of SSL is not limited to utilitarian considerations: *humans perform semi-supervised learning too* [3–5]. Therefore, to understand and improve SSL will not only help us to get a better solver for real-world problems, but also help us to better understand how natural learning comes about.

Received September 25, 2010; accepted December 15, 2010

Changshui ZHANG (✉), Fei WANG
State Key Laboratory of Intelligent Technologies and Systems, Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation,Tsinghua University, Beijing 100084, China
E-mail: zcs@mail.tsinghua.edu.cn

One key point for understanding SSL approaches is the *cluster assumption* [1], which states that [6]: 1) nearby points are likely to have the same label (local consistency); 2) points on the same structure (such as a cluster or a submanifold) are likely to have the same label (global consistency). It is straightforward to associate cluster assumption with the *manifold analysis* methods developed in recent years [7,8] (note that these methods are also in accordance with the ways that the humans perceive the world [9]). The manifold-based methods first assume that the data points (nearly) reside on a low-dimensional manifold (which is called *manifold assumption* in Ref. [1]), and then try to discover such manifold by preserving some local structure of the dataset. It is well known that graphs can be viewed as discretizations of manifolds [10]; consequently, numerous GBSSL methods have been proposed in recent years, and GBSSL has been becoming one of the most active research areas in SSL community [1].

In this paper, we will present a set of works done by our group on GBSSL including the following.

1) We propose a novel method called linear neighborhood propagation (LNP) [11,12]. LNP algorithm approximates the whole graph by a series of overlapped linear neighborhood patches, and the edge weights in each patch can be solved by a standard quadratic programming procedure. After that, all the edge weights will be aggregated together to form the weight matrix of the whole graph. Then the *Laplacian* matrix on this *pasted* graph is used as a smooth matrix as in standard GBSSL algorithms.

2) We present a fast multilevel graph learning algorithm. In our method, the data graph is first coarsened level by level based on the *similarity* between pairwise data points (which has a similar spirit with grouping, such that for each group, we only select one representative node), then the learning procedure can be performed on a graph with a much small size. Finally, the solution on the coarsened graph will be *refined* back level by level to get the solution of the initial problem. Moreover, as the formulation of unsupervised learning can be viewed as a special case of the formulation of SSL,

we will show that our multilevel method can easily be incorporated into the graph-based clustering methods. Our experimental results show that this strategy can improve the speed of GBSSL algorithms significantly, and we also give a theoretical guarantee on the performance of our algorithm.

3) We present a generalized point charge model (GPCM) for GBSSL [13]. In our model, we treat the labeled data points as point charges; therefore, the remaining unlabeled data points are placed in the electrostatic fields generated by these charges. The labels of these unlabeled data points can be regarded as the electric potentials of the electrostatic field at their corresponding places, which can be efficiently solved by making use of the discrete Green's function.

4) We present a novel GBSSL method for multi-label learning by solving a Sylvester equation (SMSE) [14]. Two graphs are first constructed on instance level and category level, respectively. For the instance level, a graph is defined based on both labeled and unlabeled instances, where each node represents one instance, and each edge weight reflects the similarity between corresponding pairwise instances. Similarly, for the category level, a graph is also built based on all the categories, where each node represents one category, and each edge weight reflects the similarity between corresponding pairwise categories. A regularization framework combining two regularization terms for the two graphs is suggested. The regularization term for the instance graph measures the smoothness of the labels of instances, and the regularization term for the category graph measures the smoothness of the labels of categories. We show that the labels of unlabeled data finally can be obtained by solving a Sylvester equation.

5) We present an information fusion framework for GBSSL problem [15]. We make use of the regularized least-square framework as the basic classifier, which only involves the similarity scores among different instances. We present a similarity score that multiplies different scores based on different distance measures. Particularly the distance measures are not restricted to the Euclidean distance. By adding a weight to each single distance-based score, we propose an EM algorithm to adaptively learn the fusion scores.

6) We present a discriminative GBSSL algorithm and apply it to semi-automated segmentation of brain tumorous tissues [16]. The classifier uses interactive hints to obtain models to classify normal and tumor tissues. A non-parametric Bayesian Gaussian random field in the semi-supervised mode is implemented. Our approach uses both labeled data and a subset of unlabeled data sampling from 2D/3D images for training the model. Fast algorithm is also developed. Experiments show that our approach produces satisfactory segmentation results compared to the manually labeled results by experts.

2 LNP

In this section, we will present the detailed algorithm of LNP. First let us introduce some notations. $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_l, \boldsymbol{x}_{l+1}, \ldots, \boldsymbol{x}_n\}$ represents a set of n data objects in $\mathbb{R}^d$, and $\mathcal{L} = \{1, -1\}$ is the label set (we consider the two-class case for the moment). The first l points $\mathcal{X}_L = \{\boldsymbol{x}_i\}_{i=1}^{l}$ are labeled as $t_i \in \mathcal{L}$, and the remaining points $\mathcal{X}_U = \{\boldsymbol{x}_u\}_{u=l+1}^{n}$ are unlabeled.

We propose to use the neighborhood information of each point to construct $\mathcal{G}$. For computational convenience, we assume that all these neighborhoods are linear, i.e., each data point can be optimally reconstructed using a linear combination of its neighbors [8]. Hence, our objective is to minimize

$$\varepsilon = \sum\nolimits_i \left\| \boldsymbol{x}_i - \sum\nolimits_{i_j:\boldsymbol{x}_{i_j}\in\mathcal{N}(\boldsymbol{x}_i)} w_{ii_j}\boldsymbol{x}_{i_j} \right\|^2, \qquad (1)$$

where $\mathcal{N}(\boldsymbol{x}_i)$ represents the neighborhood of $\boldsymbol{x}_i$, $\boldsymbol{x}_{i_j}$ is the jth neighbor of $\boldsymbol{x}_i$, and w_{ii_j} is the contribution of $\boldsymbol{x}_{i_j}$ to $\boldsymbol{x}_i$. We further constrain $\sum_{i_j\in\mathcal{N}(\boldsymbol{x}_i)} w_{ii_j} = 1$, $w_{ii_j} \geqslant 0$. Obviously, the more similar $\boldsymbol{x}_{i_j}$ to $\boldsymbol{x}_i$, the larger w_{ii_j} will be (as an extreme case, when $\boldsymbol{x}_i = \boldsymbol{x}_{i_k} \in \mathcal{N}(\boldsymbol{x}_i)$, then $w_{ii_k} = 1,\ w_{ii_j} = 0,\ i_j \neq i_k,\ \boldsymbol{x}_{i_j} \in \mathcal{N}(\boldsymbol{x}_i)$ is the optimal solution). Thus, w_{ii_j} can be used to measure how similar $\boldsymbol{x}_{i_j}$ to $\boldsymbol{x}_i$. One issue that should be mentioned here is that usually $w_{ii_j} \neq w_{i_ji}$. It can be easily inferred that

$$\varepsilon_i = \sum\nolimits_{i_j,i_k:\boldsymbol{x}_{i_j},\boldsymbol{x}_{i_k}\in\mathcal{N}(\boldsymbol{x}_i)} w_{ii_j} G^i_{i_ji_k} w_{ii_k}, \qquad (2)$$

where $G^i_{i_ji_k}$ represents the (j,k)th entry of the *local Gram matrix* $\boldsymbol{G}^i$ where $K = |\mathcal{N}(\boldsymbol{x}_i)|$ is the size of $\boldsymbol{x}_i$'s neighborhood. Thus, the reconstruction weights of each data object can be resolved by the following n standard quadratic programming problems

$$\begin{aligned} &\min\nolimits_{w_{ii_j}} \sum\nolimits_{i_j,i_k:\boldsymbol{x}_{i_j},\boldsymbol{x}_{i_k}\in\mathcal{N}(\boldsymbol{x}_i)} w_{ii_j} G^i_{i_ji_k} w_{ii_k}, \\ &\text{s. t.} \quad \sum_{i_j} w_{ii_j} = 1,\ w_{ii_j} \geqslant 0. \end{aligned} \qquad (3)$$

After all the reconstruction weights are computed, we will construct a sparse matrix $\boldsymbol{W}$ by $W(i,j) = w_{ij}$. Intuitively, this $\boldsymbol{W}$ can be treated as the weight matrix of $\mathcal{G}$. And the way we construct the whole graph is to first shear the whole graph into a series of overlapped linear patches, and then past them together.

After the graph has been constructed, we have to make use of it to predict the labels of the unlabeled vertices. Here we label propagation scheme, which can iteratively propagate the labels of the labeled data to the remaining unlabeled data $\mathcal{X}_U$ on the constructed graph.

Let $\mathcal{F}$ denote the set of classifying functions defined on $\mathcal{X}$, $\forall\ f \in \mathcal{F}$ can assign a real value f_i to every point

$\boldsymbol{x}_i$. The label of the unlabeled data point $\boldsymbol{x}_u$ is determined by the sign of $f_u = f(\boldsymbol{x}_u)$ (let us only consider the two-class case for the time being).

In each propagation step, we let each data object *absorbs* a fraction of label information from its neighborhood, and *retains* some label information of its initial state. This is similar to the strategy in Ref. [6]. Therefore, the label of $\boldsymbol{x}_i$ at time $m+1$ becomes

$$f_i^{m+1} = \alpha \sum\nolimits_{j:\boldsymbol{x}_j \in \mathcal{N}(\boldsymbol{x}_i)} w_{ij} f_j^m + (1-\alpha) t_i, \quad (4)$$

where $0 < \alpha < 1$ is the fraction of label information that $\boldsymbol{x}_i$ receives from its neighbors. Let $\boldsymbol{t} = (t_1, t_2, \ldots, t_n)^{\mathrm{T}}$ with $t_i \in \mathcal{L}$ $(i \leqslant l)$, $t_u = 0$ $(l+1 \leqslant u \leqslant n)$. $\boldsymbol{f}^m = (f_1^m, f_2^m, \ldots, f_n^m)^{\mathrm{T}}$ is the prediction label vector at iteration t and $\boldsymbol{f}^0 = \boldsymbol{t}$. Then we can rewrite our iteration equation as

$$\boldsymbol{f}^{m+1} = \alpha \boldsymbol{W} \boldsymbol{f}^m + (1-\alpha)\boldsymbol{t}. \quad (5)$$

We will use Eq. (5) to update the labels of each data object until convergence; here "convergence" means the predicted labels of the data will not change in several successive iterations.

3 A multilevel scheme

Below we will introduce a novel multilevel scheme [17] for SSL on graphs. The scheme is composed of three phases: 1) graph coarsening; 2) initial classification; 3) solution refining.

3.1 Graph coarsening

In the following, we will describe the first coarsening step. Starting from graph $\mathcal{G}^0 = \mathcal{G}$ (the superscript represents the level of graph scale), we first split $\mathcal{V}^0 = \mathcal{V}$ into two sets, $\mathcal{C}^0$ and $\mathcal{F}^0$, subject to $\mathcal{C}^0 \cup \mathcal{F}^0 = \mathcal{V}^0$, $\mathcal{C}^0 \cap \mathcal{F}^0 = \Phi$. The set $\mathcal{C}^0$ will be used as the node set of the coarser graph of the next level, i.e., $\mathcal{V}^1 = \mathcal{C}^0$. And the nodes in $\mathcal{C}^0$ are called $\mathcal{C}$-*nodes*, which is defined as:

Definition 1 ($\mathcal{C}$-nodes and $\mathcal{F}$-nodes) Given a graph $\mathcal{G}^l = (\mathcal{V}^l, \mathcal{E}^l)$, we split $\mathcal{V}^l$ into two sets, $\mathcal{C}^l$ and $\mathcal{F}^l$, satisfying $\mathcal{C}^l \cup \mathcal{F}^l = \mathcal{V}^l$, $\mathcal{C}^l \cap \mathcal{F}^l = \Phi$, $\mathcal{C}^l = \mathcal{V}^{l+1}$. And each node in $\mathcal{C}^l$ must satisfy one of the following conditions:

1) It is labeled;

2) It *strongly influences* at least one node in $\mathcal{F}^l$ on level l.

We will call the nodes in $\mathcal{C}^l$ $\mathcal{C}$-*nodes*, and the nodes in $\mathcal{F}^l$ $\mathcal{F}$-*nodes*.

Here *strongly influence* means

Definition 2 (Strongly influence) A node $\boldsymbol{x}_i$ *strongly influences* $\boldsymbol{x}_j$ on level l means that

$$w_{ij}^l \geqslant \delta \sum\nolimits_k w_{kj}^l, \quad (6)$$

where $0 < \delta < 1$ is a control parameter, and w_{ij}^l is the weight of the edge linking $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ on $\mathcal{G}^l$.

In fact, $z_{ij}^l = w_{ij}^l / \sum_k w_{kj}^l$ measures how much $\boldsymbol{x}_j$ depends on $\boldsymbol{x}_i$. Since $\boldsymbol{x}_j$ only connects to its neighborhood, a larger z_{ij} implies a larger dependency of $\boldsymbol{x}_j$ to $\boldsymbol{x}_i$. Intuitively, if $\boldsymbol{x}_j$ depends too much on $\boldsymbol{x}_i$, then we only need to retain $\boldsymbol{x}_i$. The normalization is to make z_{ij} a relative measure which is independent of the data distributions.

Let $\boldsymbol{f}^0 = \boldsymbol{f}$ be a *classification vector* we want to solve, and $\boldsymbol{f}^1$ be its *corresponding classification vector* on $\mathcal{G}^1$ (hence, the dimensionality of $\boldsymbol{f}^1$ should be equivalent to n^1, the cardinality of $\mathcal{V}^1$). The same as in other multilevel methods [18], we assume that $\boldsymbol{f}^0$ can be approximately interpolated from $\boldsymbol{f}^1$, that is[1)]

$$\boldsymbol{f}^0 \approx \boldsymbol{P}^{[0,1]} \boldsymbol{f}^1, \quad (7)$$

where $\boldsymbol{P}^{[0,1]}$ is the interpolation matrix of size $n^0 \times n^1$ $(n^0 = n)$, subject to $\sum_j \boldsymbol{P}_{ij}^{[0,1]} = 1$. Moreover, we have the following theorem.

Theorem 1 The edge weights on graph $\mathcal{G}^{l+1}$ can be computed from the edge weights on $\mathcal{G}^l$ by

$$w_{uv}^{l+1} = \frac{1}{2} \sum\nolimits_{i,j} w_{ij}^l \left(P_{jv}^{[l,l+1]} - P_{iv}^{[l,l+1]} \right) \cdot \left(P_{iu}^{[l,l+1]} - P_{ju}^{[l,l+1]} \right). \quad (8)$$

An issue that should be addressed here is that for computational efficiency, the above coarsening weight equation can be somewhat simplified to the following *iterated weighted aggregation* strategy [18], which compute w_{uv}^{l+1} by

$$w_{uv}^{l+1} = \frac{1}{2} \sum\nolimits_{i,j} P_{iu}^{[l,l+1]} w_{ij}^l P_{jv}^{[l,l+1]}. \quad (9)$$

It can be shown that Eq. (9) can provide a good approximation to Eq. (8) in many cases [20].

3.2 Initial classification

Assuming the data graph $\mathcal{G}$ has been coarsened recursively to some level s, then the semi-supervised classification problem defined on $\mathcal{G}^s$ is to minimize

$$\mathcal{J}(\boldsymbol{f}^s) = \boldsymbol{f}^{s\mathrm{T}} \boldsymbol{P}^{[s,s-1]} \ldots \boldsymbol{P}^{[1,0]} \boldsymbol{S} \boldsymbol{P}^{[0,1]} \ldots \boldsymbol{P}^{[s-1,s]} \boldsymbol{f}^s + \gamma \| \boldsymbol{P}^{[0,1]} \ldots \boldsymbol{P}^{[s-1,s]} \boldsymbol{f}^s - \boldsymbol{y} \|^2,$$

1) Actually, as we defined in Definition 1, the nodes in $\mathcal{V}^0/\mathcal{V}^1$ are largely dependent on the nodes in $\mathcal{V}^1$. Therefore, what we define in Eq. (7) is just to model such a dependence rule. The interpolation rule is simple and efficient, and it has also been widely used in the multilevel or multigrid methods for solving *partial differential equations* [18,19]; that's the reason why we apply it here

where $\boldsymbol{P}^{[i,i-1]} = \left(\boldsymbol{P}^{[i-1,i]}\right)^{\mathrm{T}}$, and $\boldsymbol{S}$ is the smoothness matrix. Therefore, let $\partial\mathcal{J}(\boldsymbol{f}^s)/\partial\boldsymbol{f}^s = 0$, then

$$\begin{aligned}\frac{\partial\mathcal{J}(\boldsymbol{f}^s)}{\partial\boldsymbol{f}^s} &= (\boldsymbol{L}^s)\,\boldsymbol{f}^s - \gamma\boldsymbol{P}^{[s,s-1]}\cdots\boldsymbol{P}^{[1,0]}\boldsymbol{y} = 0\\ &\Longrightarrow \boldsymbol{f}^s = \gamma\,(\boldsymbol{L}^s)^{-1}\,\boldsymbol{P}^{[s,s-1]}\cdots\boldsymbol{P}^{[1,0]}\boldsymbol{y}.\end{aligned}$$

Moreover, we have the following theorem.

Theorem 2 The matrix $\boldsymbol{L}^s = \boldsymbol{P}^{[s,s-1]}\cdots\boldsymbol{P}^{[1,0]}(\boldsymbol{S} + \gamma\boldsymbol{I})\boldsymbol{P}^{[0,1]}\cdots\boldsymbol{P}^{[s-1,s]}$ is invertible.

Based on the above theorem, we can compute the *initial classification vector*, in which we only need to compute the inverse of an $n^s \times n^s$ matrix, and usually n^s is much smaller than n.

3.3 Solution refining

Having achieved the *initial classification vector* from Eq. (10), we have to refine it level by level to get a classification vector on the initial graph $\mathcal{G}^0 = \mathcal{G}$. As stated in Sect. 3.1, we assume that the classification vector on graph $\mathcal{G}^l$ can be linearly interpolated from $\mathcal{G}^{l+1}$, i.e., $\boldsymbol{f}^l = \boldsymbol{P}^{[l,l+1]}\boldsymbol{f}^{l+1}$. Here, $\boldsymbol{P}^{[l,l+1]}$ is an $n^l \times n^{l+1}$ interpolation matrix subject to $\sum_j \boldsymbol{P}_{ij}^{[l,l+1]} = 1$.

Based on the simple geometric intuition that the label of a point should be similar to the label of its neighbors (which is also consistent with the cluster assumption we introduced in Sect. 1), we propose to compute $P_{iI(j)}^{[l,l+1]}$ by

$$P_{iI(j)}^{[l,l+1]} = \begin{cases} w_{ij}^l / \sum_{k\in\mathcal{C}^l} w_{ik}^l, & i \notin \mathcal{C}^l,\\ 1, & i = j,\\ 0, & \boldsymbol{x}_i \in \mathcal{C}^l,\ i \neq j. \end{cases} \tag{10}$$

In the above equation, subscripts i, j, k are used to denote the index of the nodes in $\mathcal{V}^l$. We assume that node j has been selected as a $\mathcal{C}$-*node*, and $I(j)$ is the index of j in $\mathcal{V}^{l+1}$. It can be easily inferred that $\boldsymbol{P}^{[l,l+1]}$ has full rank.

4 A generalized point charge model for GB-SSL

First let us see the analogy between SSL and point charge models. In SSL, we are given some labeled and unlabeled data points; the goal is to predict the label of those unlabeled data points (*transduction*) and even the unseen out-of-sample data points (*induction*). Viewing the whole system as a *electrostatic field* model [21], we can treat the labeled data points as source point charges, which generate an electric field over the whole data space. Associated with each place in the electric field, there is an *electric potential* [21], which is a scalar, and varies smoothly with respect to the electric field. Similarly, in SSL, we also require that the data labels vary smoothly with respect to the intrinsic data manifold; then it is reasonable to regard the labels of the points as the electric potentials at their corresponding places[1)]. Therefore, we can predict the labels of the unlabeled points by computing the superposed electric potentials of their corresponding places.

An issue to be addressed here is that in electrostatic fields, the electric potentials vary smoothly in the physical space, which is usually a three-dimensional Euclidean space; however, in SSL, the intrinsic data manifold is usually *Riemannian*. Therefore, we should describe the data manifold and "transplant" the physical theorems in electrostatic fields to the data manifold. More concretely, let f be the label prediction function such that $f(\boldsymbol{x}_i)$ returns the "soft" label of $\boldsymbol{x}_i$, then the *Gauss's law* [21] tells us that f should satisfy the following *Poisson's equation*

$$\Delta_{\mathcal{M}} f = c\rho, \tag{11}$$

where ρ is the distribution of the "label sources", which can be modelled using the *Dirac delta function* as $\rho = \sum_{i=1}^{l} t_i\delta(\boldsymbol{x} - \boldsymbol{x}_i)$ (where $\boldsymbol{x}, \boldsymbol{x}_i$ represents the positions of the data points on $\mathcal{M}$), $\Delta_{\mathcal{M}}$ is the *Laplace Beltrami operator* on the data manifold $\mathcal{M}$ [7], and $c \neq 0$ is a constant. Since the multiplication of f with any constant will not affect the final decision of the data labels, we will simply drop c in the following. Then f can be solved using a properly defined Green's function [21], which can be realized with the help of the unlabeled data points.

Since in SSL scenario, the dataset can be viewed as being sampled the data manifold, then the data manifold $\mathcal{M}$ will degenerate to the data graph $\mathcal{G}$, and the Laplace-Beltrami operator $\Delta_{\mathcal{M}}$ on $\mathcal{M}$ will become the Laplacian matrix, i.e., *combinatorial Laplacian* $\boldsymbol{L}_c$ [22] or *normalized Laplacian* $\boldsymbol{L}_n$ [6], defined on $\mathcal{G}$ [7]. Following Ref. [23], we define the *discrete Green's function* defined on graphs as

$$\boldsymbol{G}_1 = \sum_{i>0}(1/\lambda_i)\boldsymbol{v}_i\boldsymbol{v}_i^{\mathrm{T}}, \quad \boldsymbol{G}_2 = \sum_{i>0}(1/\tau_i)\boldsymbol{u}_i\boldsymbol{u}_i^{\mathrm{T}}, \tag{12}$$

where $i = 1, 2, \ldots, n-1$, $0 = \lambda_0 \leqslant \lambda_1 \leqslant \cdots \leqslant \lambda_{n-1}$ are the eigenvalues of $\boldsymbol{L}_c$, and $\{\boldsymbol{v}_i\}_{i=0}^{n-1}$ are their corresponding eigenvectors; $0 = \tau_0 \leqslant \tau_1 \leqslant \cdots \leqslant \tau_{n-1}$ are the eigenvalues of $\boldsymbol{L}_n$, and $\{\boldsymbol{u}_i\}_{i=0}^{n-1}$ are their corresponding eigenvectors. We give the Green's function on a synthetic data set in Fig. 1, which clearly illustrates the cluster structure contained in the data set. Intuitively, $\boldsymbol{G}(i,j)$[2)] represents the "label potential" at position j when there is a unit label source at position i (here the positions of the data points are defined on the data manifold). Then

1) Although data labels are discrete while electric potentials are continuous, we can relax the data labels to be continuous as in Ref. [22] and regard the electric potentials as the relaxed "soft" labels

2) In the rest of this letter, we will use $\boldsymbol{G}$ as the unified notation for the Green's functions, i.e., $\boldsymbol{G}_1$ or $\boldsymbol{G}_2$

we can predict the labels of the unlabeled data points by

$$\begin{aligned} f(\boldsymbol{x}) &= \int G(\boldsymbol{x},\boldsymbol{s})\rho(\boldsymbol{s})\mathrm{d}\boldsymbol{s} \\ &= \int G(\boldsymbol{x},\boldsymbol{s})\sum_{i=1}^{l} t_i\delta(\boldsymbol{s}-\boldsymbol{x}_i)\mathrm{d}\boldsymbol{s} \\ &= \sum_{i=1}^{l} G(\boldsymbol{x},\boldsymbol{x}_i)t_i, \end{aligned} \tag{13}$$

which can be rewritten in its matrix form as $\boldsymbol{f} = \boldsymbol{G}\boldsymbol{\rho}$, where $\boldsymbol{\rho}_i = \begin{cases} t_i, & \boldsymbol{x}_i \text{ is labeled as } t_i \\ 0, & \boldsymbol{x}_i \text{ is unlabeled} \end{cases}$ is the "label source" vector. Then the final data labels can be determined by the sign of $\boldsymbol{f}$. Till now, we have restricted the classification problem to be a two-class problem since a point charge can only have two types of charges, positive or negative (since the different types of charges correspond to the different classes). For multi-class problems, we can *generalize* the traditional point charge model to C types of charges and apply an *indicator* to represent each type of charge, e.g., the $C\times 1$ vector $(0,0,1,\ldots,0)^{\mathrm{T}}$ denotes a unit charge of type 3. Then our algorithm can be easily extended to multi-class problems by $\boldsymbol{F} = \boldsymbol{G}\boldsymbol{\Psi}$, where $\boldsymbol{\Psi}$ is an $n\times C$ matrix with its (i,j)th entry $\boldsymbol{\Psi}_{ij} = \begin{cases} 1, & \boldsymbol{x}_i \text{ is labeled as class } j \\ 0, & \text{otherwise} \end{cases}$ and the final label of $\boldsymbol{x}_i$ can be determined by $\arg\max_j \boldsymbol{F}_{ij}$ $(1\leqslant i\leqslant n,\ 1\leqslant j\leqslant C)$.

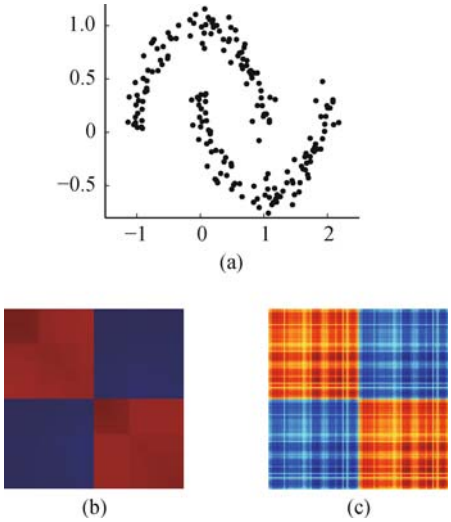

Fig. 1 An illustration of the Green's function. (a) shows the original two-moon dataset; (b) and (c) show the Green's function for this dataset, such that the color for pixel (i,j) represents the value of $\boldsymbol{G}_1(i,j)$ (or $\boldsymbol{G}_2(i,j)$). The larger (smaller) that value, the closer the corresponding pixel tends to red (blue). (a) Two-moon dataset; (b) Green's function $\boldsymbol{G}_1$; (c) Green's function $\boldsymbol{G}_2$

5 Multi-label GBSSL by SMSE

Multi-label learning refers to the problem where each data point is associated with multiple labels. Traditional GBSSL methods only construct a graph on the instance level, which is appropriate when there are no correlations among categories. However, category correlations often exist in a typical multi-label learning scenario. Therefore, in order to make use of the correlation information, we have another graph constructed on category level too [14]. Let $G' = (V', E')$ denote the category graph with k nodes, where each node represents one category. We define a $k\times k$ symmetric weight matrix $\boldsymbol{W}'$ as the following equation

$$W'_{ij} = \exp\left[-\lambda(1-\cos(c_i,c_j))\right], \tag{14}$$

where λ is a hyperparameter, c_i is a binary vector whose elements are set to be one when the corresponding training instances belong to the ith category and zero otherwise and $\cos(c_i,c_j)$ computes the Cosine Similarity between c_i and c_j.

Define $\boldsymbol{F} = [\boldsymbol{f}_1,\boldsymbol{f}_2,\ldots,\boldsymbol{f}_n]^{\mathrm{T}} = [\boldsymbol{g}_1,\boldsymbol{g}_2,\ldots,\boldsymbol{g}_k]$, where k is the number of categories. Then we can define an energy on the category graph as

$$E'(\boldsymbol{g}) = \frac{1}{2}\sum_{i,j=1}^{k} W'_{ij}\|\boldsymbol{g}_i-\boldsymbol{g}_j\|^2. \tag{15}$$

This can also be viewed as a regularization term that measures the smoothness of the labels of categories. Then our problem becomes

$$\min_{\boldsymbol{f}} \sum_{i=1}^{l}\|\boldsymbol{f}_i-\boldsymbol{t}_i\|^2+\beta\mathrm{tr}\left(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{L}\boldsymbol{F}\right)+\gamma\mathrm{tr}\left(\boldsymbol{F}\boldsymbol{H}\boldsymbol{F}^{\mathrm{T}}\right), \tag{16}$$

where $\boldsymbol{L}$ is the regular graph Laplacian constructed on the instances, and $\boldsymbol{H}$ is the graph Laplacian constructed on the category graph. Taking the first order derivative of the objective with respect to $\boldsymbol{F}$, we can get the following Sylvester equation

$$(\beta\boldsymbol{L}+\boldsymbol{J})\boldsymbol{F}+\gamma\boldsymbol{F}\boldsymbol{H} = \boldsymbol{J}\boldsymbol{T}, \tag{17}$$

which is a standard Sylvester equation that can be solved via existing techniques.

6 Semi-supervised information fusion for classification

Most classification work can be divided into two stages: feature extraction and classifier design. Classification based on combination of different features is a good way to improve the accuracy, since it can reduce the disadvantage that the single feature cannot describe all kinds of genres. Semi-supervised classification is a good choice to reduce the amount of manual labeling work and improve the classification accuracy rate. We propose a method that can adaptively fuse different features for the semi-supervised classification problem. We present

a similarity score that multiplies different scores based on different distance measures. Particularly the distance measures are not restricted to the Euclidean distance. By adding a weight to each single distance-based score, we propose an EM algorithm to adaptively learn the fusion scores.

Suppose we have l labeled data and u unlabeled data. Given the training data set $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_l, \boldsymbol{x}_{l+1}, \ldots, \boldsymbol{x}_{l+u}\}$ and $\{y_1, y_2, \ldots, y_l\}$, where $\boldsymbol{x}_i \in X$ is the feature and $y_i \in Y$ is the label, we not only want to estimate the soft labels of the training data points $\{f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_{l+u})\}$, but also want to know what the soft label $f(\boldsymbol{x})$ of arbitrary new test point $\boldsymbol{x}$ in the input space X is. To incorporate the additional information of unlabeled data, we use the following regularization framework [24]:

$$f^*(\boldsymbol{x}) = \arg\min_{f\in\mathcal{H}_K} \int_{X\times Y} V(y, f(\boldsymbol{x}))\mathrm{d}P(\boldsymbol{x}, y) + \lambda_1||f||_K^2 + \lambda_2||f||_I^2, \quad (18)$$

where $\int_{X\times Y} V(y, f(\boldsymbol{x}))\mathrm{d}P(\boldsymbol{x}, y)$ is the *expected risk*, the loss function is $V(y, f(\boldsymbol{x}))$ and $P(\boldsymbol{x}, y)$ is the joint distribution. The solution $f^*(\boldsymbol{x})$ is a function $X \to \mathcal{R}$ which lies in a bounded convex subset of reproducing kernel Hilbert space (RKHS) $\mathcal{H}_K$ is defined by a positive-definite kernel function $K : X\times X \to \mathcal{R}$. The kernel function satisfies the Mercer condition. $||f||_K^2$ is the traditional Tikhonov regularization term in RHKS, and $||f||_I^2$ is the regularization term based on manifold analysis. λ_1 and λ_2 are the parameters which control the tradeoffs of these two terms.

Suppose there are several types of feature, the fusion work is to use the information provided by these features to gain more accurate rates. We denote $D_k\left(\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\right)$ as the kth distance measure of two music tracks: track i and track j. Note that $\boldsymbol{x}_i^{(k)}$ and $\boldsymbol{x}_j^{(k)}$ are their corresponding features, which are not restricted to vectors in Euclidean space. To show the information provided by the kth distance measure, we define a score that related to $D_k(\cdot,\cdot)$:

$$S_k\left(\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\right) = \exp\left\{-\frac{D_k\left(\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\right)}{2\sigma_k^2}\right\}. \quad (19)$$

$S_k(\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)})$ is the exponential negative distance score, which means if the distance between two music tracks is small, they have a large similarity score. σ_k is the width of the Gaussian kernel, and its inverse $\beta_k = 1/(2\sigma_k^2)$ can be seen as the weight that controls the distance scaling.

Note that the kernel function $K(\cdot,\cdot)$ and the graph weight $W(\cdot,\cdot)$ have the same form, which can be interpreted in the type of the similarity score. A mechanism to fuse the multiple distance information can then be naturally designed, which multiplies different similarity scores:

$$\begin{aligned} S(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \prod\nolimits_k S_k(i,j) \\ &= \exp\left\{-\sum\nolimits_k \frac{D_k\left(\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\right)}{2\sigma_k^2}\right\} \\ &= \exp\left\{-\sum\nolimits_k \beta_k D_k\left(\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\right)\right\}. \end{aligned} \quad (20)$$

We can see that multiplying similarity scores is equivalent to the sum of weighted distance measure, and the weight β_k controls the importance of distance measure $D_k(\cdot,\cdot)$.

In empirical Bayesian language, the framework can be seen as a maximum of a posterior (MAP) estimation of the weight vector $\boldsymbol{\alpha}$:

$$P(\boldsymbol{\alpha}|\Theta, \boldsymbol{y}, \boldsymbol{X}) \propto P(\boldsymbol{y}|\boldsymbol{\alpha}, \boldsymbol{X})P(\boldsymbol{\alpha}|\Theta, \boldsymbol{X}), \quad (21)$$

where the likelihood $P(\boldsymbol{y}|\boldsymbol{\alpha}, \boldsymbol{X})$ corresponds to the exponential negative loss function, and the prior $P(\boldsymbol{\alpha}|\Theta, \boldsymbol{X})$ corresponds to the exponential negative regularization term. This estimation is computed with the fixed fusion parameters $\Theta = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ and fixed regularization controlling parameters $\{\lambda_1, \lambda_2\}$.

We use the least-squares loss function, thus the noise between soft label f and label y is regarded as Gaussian $P(y|f) = \exp\{-\frac{1}{2}(y-f)^2/l\}$. The likelihood is then

$$P(\boldsymbol{y}|\boldsymbol{\alpha}, \boldsymbol{X}) \propto \prod_i^l \exp\left\{-\frac{1}{2}\frac{(y_i - \sum_{j=1}^{l+u}\alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_i))^2}{l}\right\}. \quad (22)$$

Since the prior of the weight vector $\boldsymbol{\alpha}$ has the Gaussian form:

$$P(\boldsymbol{\alpha}|\Theta, \boldsymbol{X}) \propto \exp\left\{-\frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}(\lambda_1\boldsymbol{K} + \lambda_2\boldsymbol{K}\boldsymbol{L}\boldsymbol{K})\boldsymbol{\alpha}\right\}, \quad (23)$$

the posterior of weight vector $\boldsymbol{\alpha}$ is also Gaussian:

$$\begin{aligned} &N((\boldsymbol{J}\boldsymbol{K} + \lambda_1 l\boldsymbol{I} + \lambda_2 l\boldsymbol{L}\boldsymbol{K})^{-1}\boldsymbol{y}, \\ &\quad (\boldsymbol{K}\boldsymbol{J}\boldsymbol{K} + \lambda_1 l\boldsymbol{K} + \lambda_2 l\boldsymbol{K}\boldsymbol{J}\boldsymbol{K})^{-1}), \end{aligned} \quad (24)$$

where

$$\boldsymbol{y} = [y_1, y_2, \ldots, y_l, 0_{l+1}, \ldots, 0_{l+u}]^{\mathrm{T}}$$

is an $\mathcal{R}^{l+u}$ vector. $\boldsymbol{J}$ is a diagonal matrix, which is

$$\boldsymbol{J} = \mathrm{diag}(1_1, 1_2, \ldots, 1_l, 0_{l+1}, \ldots, 0_{l+u}).$$

$(\boldsymbol{K})_{(l+u)\times(l+u)}$ is the Gram matrix which satisfies $\boldsymbol{K}_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$. It is being centered at

$$\boldsymbol{\alpha}^* = (\boldsymbol{J}\boldsymbol{K} + \lambda_1 l\boldsymbol{I} + \lambda_2 l\boldsymbol{L}\boldsymbol{K})^{-1}\boldsymbol{y},$$

and the covariance is

$$(\boldsymbol{K}\boldsymbol{J}\boldsymbol{K} + \lambda_1 l\boldsymbol{K} + \lambda_2 l\boldsymbol{K}\boldsymbol{J}\boldsymbol{K})^{-1}.$$

To learn the fusion parameters Θ, we use the following maximum likelihood (ML) estimation:

$$P(\boldsymbol{y}|\Theta, \boldsymbol{X}) = \int P(\boldsymbol{y}|\boldsymbol{\alpha}, \boldsymbol{X})P(\boldsymbol{\alpha}|\Theta, \boldsymbol{X})\mathrm{d}\boldsymbol{\alpha}. \quad (25)$$

It is intractable to compute the above integral. We address the problem as maximizing a lower bound using Jensen's inequality:

$$\begin{aligned}
&\log P(\boldsymbol{y}|\Theta,\boldsymbol{X})\\
&\quad=\log\int P(\boldsymbol{y}|\boldsymbol{\alpha},\boldsymbol{X})P(\boldsymbol{\alpha}|\Theta,\boldsymbol{X})\mathrm{d}\boldsymbol{\alpha}\\
&\quad\geqslant\int q(\boldsymbol{\alpha})\log\frac{P(\boldsymbol{y}|\boldsymbol{\alpha},\boldsymbol{X})P(\boldsymbol{\alpha}|\Theta,\boldsymbol{X})}{q(\boldsymbol{\alpha})}\mathrm{d}\boldsymbol{\alpha}\\
&\quad=\int P(\boldsymbol{\alpha}|\Theta^{\mathrm{old}},\boldsymbol{y},\boldsymbol{X})\log\frac{P(\boldsymbol{y}|\boldsymbol{\alpha},\boldsymbol{X})P(\boldsymbol{\alpha}|\Theta^{\mathrm{new}},\boldsymbol{X})}{P(\boldsymbol{\alpha}|\Theta^{\mathrm{old}},\boldsymbol{y},\boldsymbol{X})}\mathrm{d}\boldsymbol{\alpha}\\
&\quad\triangleq J\left(\Theta^{\mathrm{new}},\Theta^{\mathrm{old}}\right),
\end{aligned}\tag{26}$$

where $q(\boldsymbol{\alpha})=P(\boldsymbol{\alpha}|\Theta^{\mathrm{old}},\boldsymbol{y},\boldsymbol{X})$. If $q(\boldsymbol{\alpha})$ is solved by factorization or some approximation methods, it is the variational EM method. In our approach, $q(\boldsymbol{\alpha})$ can be computed as an exact Gaussian form Eq. (24). Thus, our algorithm is an EM algorithm.

Then we can update the fusion parameters. We regard the weights $\boldsymbol{\alpha}$ as the "hidden" variables in the EM framework, and estimate the fusion parameters Θ in each iteration. In the E-step, the posterior Gaussian distribution of "hidden" variables $P(\boldsymbol{\alpha}|\Theta,\boldsymbol{y},\boldsymbol{X})$ is calculated due to some fixed fusion parameters Θ, then the new lower bound is calculated by using expectation on this distribution; In the M-step, the lower bound is maximized with respect to the fusion parameters.

E-step Given the parameters, we estimate the posterior probability of "hidden" variables $P(\boldsymbol{\alpha}|\Theta^{\mathrm{old}},\boldsymbol{y},\boldsymbol{X})$.

M-step Given $P(\boldsymbol{\alpha}|\Theta^{\mathrm{old}},\boldsymbol{X},\boldsymbol{X})$ computed in the E-step, we maximize the lower bound (Eq. (26)) with respect to the parameters. The update is based on gradient descent, which makes $\partial J(\Theta^{\mathrm{new}},\Theta^{\mathrm{old}})/\partial\Theta^{\mathrm{new}}=0$.

The E-step and M-step are alternated until convergence.

7 Semi-supervised brain tumor image segmentation

Magnetic resonance (MR) imaging has been proven to be a useful noninvasive technique for assisting in clinical diagnoses and in evaluating therapy results, due to its high contrast resolution and ability to provide rich information about human soft tissue. Segmentation of MR brain images is the first step of quantitative analysis. Computer-assisted brain tumor segmentation is one of the most important and hardest issues in segmenting abnormities. There are two main problems.

First, automatic measurement of the volume and variation of these tumorous tissues is not easy. The distribution of normal tissue intensity is complex, and overlapping exists among different types of tissues. In addition, the cerebral tumorous tissues in MR brain images may vary in size, shape and location. They are usually accompanied by edema. Other tissues, such as hemorrhage, necrosis and cystic components, may also appear in the tumorous region. Therefore, the boundaries of the tumorous tissues can be rather blurry.

Second, there are a great many pixels (such as $256\times256\times124$) for 3D MR images. Consequently, segmentation will have high computational complexity and require large memory storage. This problem can be solved by applying the 2D methods sequently, and even experts segment the images in this way. However, this will lose some geometric information.

We propose a method based on graph regularization. It is a semi-supervised inductive method, which uses labeled data in one image and a subset of unlabeled data sampling from 2D/3D images to classify the remains. It can directly segment 3D data by sampling the unlabeled data from 3D images rather than by segmenting 2D images sequently.

We denote one input data point as a feature vector $\boldsymbol{x}_i$, and $\boldsymbol{X}_N\triangleq\{\boldsymbol{x}_i\}_{i=1}^N$ is the observed data set including both labeled and unlabeled data. For the semi-supervised problem, we attempt to extend the labels of the labeled to the unlabeled, whose labels are set to zeros initially. The label of $\boldsymbol{x}_i$ is given by t_i $(i=1,2,\ldots,N)$, and $\boldsymbol{x}_N=(t_1,t_2,\ldots,t_N)^{\mathrm{T}}$. We also denote the training data set as $D=\{\boldsymbol{x}_i,t_i\}_{i=1}^N$. For inductive problems, we want to estimate the label t_{N+1} of a new point $\boldsymbol{x}_{N+1}$. For MR images, each voxel $\boldsymbol{x}_i=\left(\boldsymbol{x}_i^{(c)},\boldsymbol{x}_i^{(p)}\right)$ in 3D space is represented by a six-dimensional feature vector: three spatial features and three intensity's features of T1, T2 and PD weighted images. $\boldsymbol{x}^{(p)}$ is the coordinate vector (X,Y,Z) of the voxel, and $\boldsymbol{x}^{(c)}$ is the vector of intensities $(I_{\mathrm{T1}},I_{\mathrm{T2}},I_{\mathrm{PD}})$ of T1, T2 and PD images. We label some of the pixel/voxel as the labeled points, which are the hints obtained by very simple human-computer interaction. The unlabeled data are obtained by randomly sampling from the 2D/3D image data. We use the training data, which include both labeled and unlabeled data, to train a classification model. For all the image data, the classification model can get a prediction whether a pixel/voxel belongs to tumorous or normal tissues. The segmentation is done by classifying all the image data.

By computing the mode of posterior $P(\boldsymbol{y}_N|\boldsymbol{t}_N,\Theta)$ as the estimation of $\boldsymbol{y}_N$, which is the negative logarithm of $P(\boldsymbol{y}_N|\boldsymbol{t}_N,\Theta)=P(\boldsymbol{y}_N,\boldsymbol{t}_N|\Theta)/P(\boldsymbol{t}_N)$, we define the Gaussian density function of $\boldsymbol{y}_N$ as

$$\Psi(\boldsymbol{y}_N)\approx-\log P(\boldsymbol{t}_N|\boldsymbol{y}_N)-\log P(\boldsymbol{y}_N|\Theta),\tag{27}$$

where $P(\boldsymbol{t}_N)$ is omitted since it is a constant with respect to $\boldsymbol{y}_N$. $P(\boldsymbol{y}_N|\Theta)$ is the prior of latent variables, which is modeled based on graph regularization:

$$P(\boldsymbol{y}_N|\Theta)=\frac{1}{Z_{\mathrm{r}}}\exp\left\{-\frac{S(\boldsymbol{y}_N)}{\mu}\right\},\tag{28}$$

where Z_{r} is the normalization constant, and the smoothness function $S(\boldsymbol{y}_N)$ is

$$S(\boldsymbol{y}_N) = \frac{1}{2}\boldsymbol{y}_N^{\mathrm{T}}(\boldsymbol{I} - \boldsymbol{S})\boldsymbol{y}_N, \tag{29}$$

where $\boldsymbol{S} = \boldsymbol{S}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{D}^{-\frac{1}{2}}$, $\boldsymbol{D} = \mathrm{diag}(D_{11}, D_{22}, \ldots, D_{NN})$ and $(\boldsymbol{W})_{ij} = W_{ij}$. $D_{ii} = \sum_j W_{ij}$, and W_{ij} is the weight function associated with the edges on graph, which satisfies: $W_{ij} \geqslant 0$ and $W_{ij} = W_{ji}$.

$P(\boldsymbol{t}_N|\boldsymbol{y}_N)$ is the new conditional probability *extended Bernoulli model*:

$$P(t_i|y_i) = \frac{1-\lambda}{1+\exp(-t_i y_i)}\delta_{\{t_i \neq 0\}} + \lambda\delta_{\{t_i=0\}}, \tag{30}$$

where δ is the indicator function. $\delta_{\{t_i=0\}}$ means: if $t_i = 0$, $\delta = 1$, and if $t_i \neq 0$, $\delta = 0$.

For training the model, which estimates the latent variables from the posterior probability $P(\boldsymbol{y}_N|\boldsymbol{t}_N, \Theta)$, we make use of the Laplace approximation [25] for the posterior probability with fixed hyper-parameters. By differentiating Eq. (27) with respect to $\boldsymbol{y}_N$, we have

$$\nabla\Psi = \boldsymbol{\alpha}_N + \boldsymbol{g}_N, \quad \nabla\nabla\Psi = \boldsymbol{\Pi}_N + \boldsymbol{K}_N^{-1}, \tag{31}$$

where

$$\begin{aligned} \boldsymbol{g}_N &= \nabla_{\boldsymbol{y}_N}\left(-\log P\left(\boldsymbol{y}_N|\Theta\right)\right) = \boldsymbol{K}_N^{-1}\boldsymbol{y}_N, \\ \boldsymbol{K}_N^{-1} &= \nabla\nabla_{\boldsymbol{y}_N}\left(-\log P\left(\boldsymbol{y}_N|\Theta\right)\right) = \boldsymbol{\Delta} = \boldsymbol{I} - \boldsymbol{S}, \end{aligned} \tag{32}$$

and

$$\begin{aligned} \boldsymbol{\alpha}_N &= \nabla_{\boldsymbol{y}_N}\left(-\log P\left(\boldsymbol{t}_N|\boldsymbol{y}_N\right)\right) \\ &= \left(-\frac{t_i}{1+\exp(t_i y_i)}\right)_i, \\ \boldsymbol{\Pi}_N &= \nabla\nabla_{\boldsymbol{y}_N}\left(-\log P\left(\boldsymbol{t}_N|\boldsymbol{y}_N\right)\right) \\ &= \mathrm{diag}\left[\frac{t_i^2\exp(t_i y_i)}{\left(1+\exp(t_i y_i)\right)^2}\right]. \end{aligned} \tag{33}$$

To find the expectation of the approximated Gaussian density Ψ in Eq. (27), the Newton-Raphson iteration is adopted:

$$\boldsymbol{y}_N{}^{new} = \boldsymbol{y}_N - (\nabla\nabla\Psi)^{-1}\nabla\Psi. \tag{34}$$

Since $\nabla\nabla\Psi$ is always positive definite[1)], Eq. (27) is a convex problem. When it converges to an optimal $\hat{\boldsymbol{y}}_N$, $\nabla\Psi$ will be a zero vector. The posterior probability $P(\boldsymbol{y}_N|\boldsymbol{t}_N, \Theta)$ is approximated as Gaussian, being centered at the estimated $\hat{\boldsymbol{y}}_N$. The inverse of the posterior covariance is $\nabla\nabla\Psi$.

Although we have a clear and compact formulation to realize the semi-supervised induction, there still exists a problem that the training computational complexity is scaling to $O(N^3)$, where N is the number of the training data. A simple way to reduce the $O(N^3)$ computational requirement and $O(N^2)$ memory requirement is to express the solution as sampled examples from the training set. A successful usage of sampling approach has been proposed by Williams and Seeger [26], which is called the Nyström method. It is also used to speed up the normalized cut algorithm in a modified way [27].

Suppose we have N points in the training data set, the goal of the Nyström method is to use the random partition of M ($M \ll N$) and $(N-M)$ points to approximate the original $N \times N$ matrix. It has been proved that the matrix $\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{D}^{-\frac{1}{2}}$ can be decomposed as $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}$, where $\boldsymbol{\Lambda}$ is an $M \times M$ matrix [27]. In the training phase, the computer solves the inverse of $\nabla\nabla\Psi$ when it computes the Newton-Raphson iteration to find the minimum of Ψ (Eq. (34)). We rewrite the Hessian matrix as

$$\begin{aligned} \nabla\nabla\Psi &= \boldsymbol{\Pi}_N + \boldsymbol{K}_N^{-1} \\ &= \boldsymbol{\Pi}_N + \boldsymbol{I} - \hat{\boldsymbol{D}}^{-\frac{1}{2}}\hat{\boldsymbol{W}}\hat{\boldsymbol{D}}^{-\frac{1}{2}} \\ &= \boldsymbol{\Pi}_N + \boldsymbol{I} - \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}. \end{aligned}$$

Therefore, by applying the Woodbury formula [28], we have

$$\begin{aligned} &(\nabla\nabla\Psi)^{-1} \\ &\quad = (\boldsymbol{\Pi}_N + \boldsymbol{I})^{-1} \\ &\qquad + (\boldsymbol{\Pi}_N + \boldsymbol{I})^{-1}\boldsymbol{U}\left[\boldsymbol{I} - \boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{\Pi}_N + \boldsymbol{I})^{-1}\boldsymbol{U}\right]^{-1} \\ &\qquad\quad \boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{\Pi}_N + \boldsymbol{I})^{-1}. \end{aligned} \tag{35}$$

Thus, the computational requirement of Newton-Raphson iteration is $O(M^2N)$, because it only needs to compute the singular value decomposition (SVD) of an $M \times M$ sub-matrix. The memory usage is $O(MN)$ since it only stores: 1) the weight matrix of the sampling points, 2) the weights between sampling and the remaining points in the training set.

To find whether a voxel, which is not in the observed training set, is tumorous or normal, is equivalent to estimating the label t_{N+1} of a new point $\boldsymbol{x}_{N+1}$. Given the estimated hyper-parameters and labels, the prediction function is to compute the integral over the latent variable of a new point:

$$P(t_{N+1}|D, \Theta) = \int P(t_{N+1}|y_{N+1})P(y_{N+1}|D, \Theta)\mathrm{d}y_{N+1}. \tag{36}$$

The second factor in Eq. (36) can be obtained by further integration:

$$P(y_{N+1}|D, \Theta) = \int P(y_{N+1}|\boldsymbol{y}_N, \Theta)P(\boldsymbol{y}_N|\boldsymbol{t}_N, \Theta)\mathrm{d}\boldsymbol{y}_N. \tag{37}$$

The segmentation examples are shown in Fig. 2.

1) For the positive semi-definite case, we can add extra regularization as the jitter noise

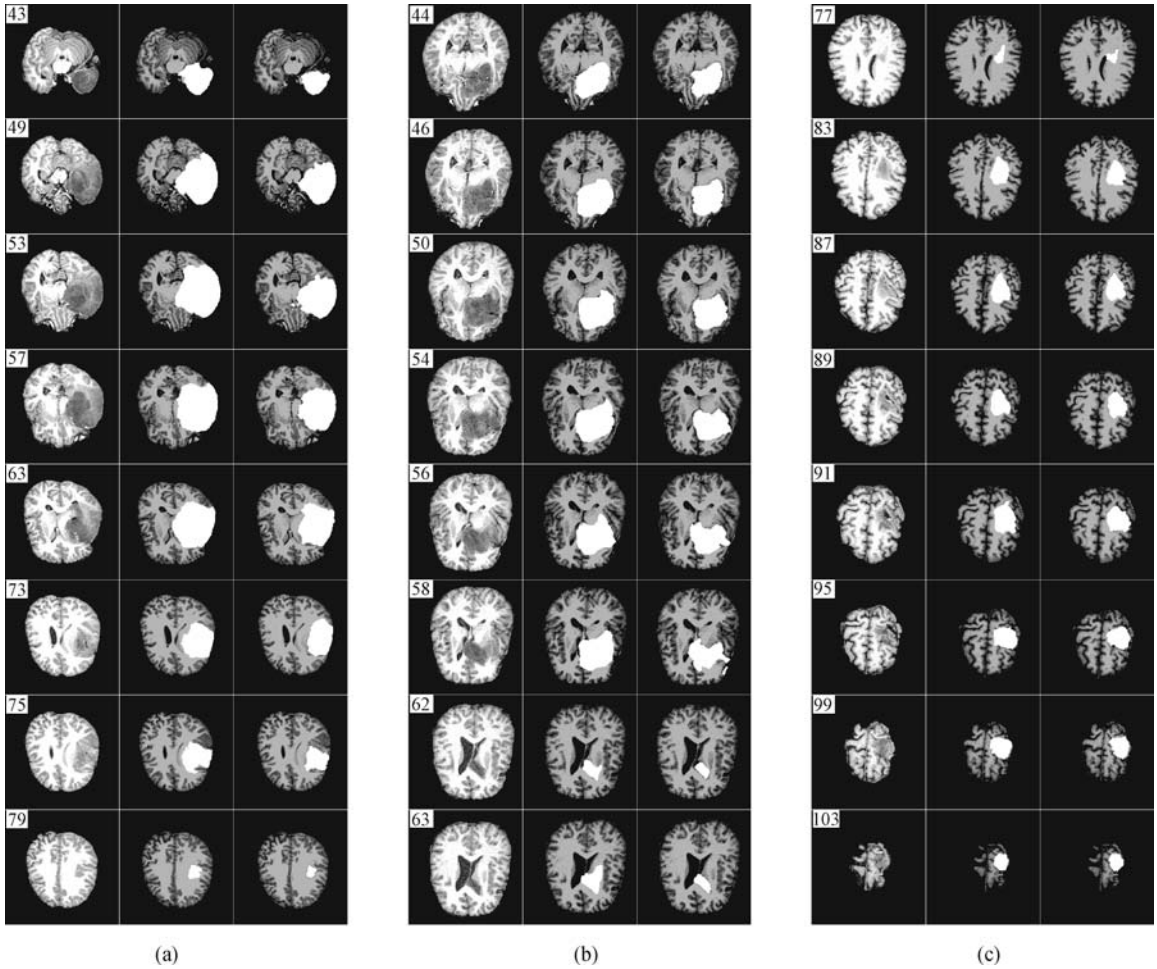

Fig. 2 3D segmentation results. The left columns of each sub-figure are the original T1 weighted images. The middle columns are the hand-guided standard ground truth Hand_{mid}. The right columns are the results of SSGPI. (a) Patient 1; (b) patient 2; (c) patient 3

8 Conclusions

GBSSL is an important problem that has been becoming more and more popular in recent years. In this paper, we summarized a series of works that developed in our group in recent years, including both theoretical foundation and applications.

Acknowledgements This paper was supported by the National Natural Science Foundation of China (Grant Nos. 60835002 and 61075004).

References

1. Chapelle O, Schölkopf B, Zien A. Semi-Supervised Learning. Cambridge: MIT Press, 2006
2. Zhu X. Semi-supervised learning literature survey. Technical Report 1530, Univ. Wisconsin-Madison. 2005
3. Graf E K, Evans J L, Alibali M W, Saffran J R. Can infants map meaning to newly segmented words? Statistical segmentation and word learning. Psychological Science, 2007, 18(3): 254–260
4. Stromsten S B. Classification learning from both classified and unclassified examples. Dissertation for the Doctoral Degree. Palo Alto: Stanford University, 2002
5. Zhu X, Rogers T, Qian R, Kalish C. Humans perform semi-supervised classification too. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence. 2007, 864–869
6. Zhou D, Bousquet O, Lal T N, Weston J, Schölkopf B. Learning with local and global consistency. In: Thrun S, Saul L, Schölkopf B, eds. Advances in Neural Information Processing Systems. 2004, 16: 321–328
7. Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation, 2003, 15(6): 1373–1396
8. Roweis S T, Saul L K. Noninear dimensionality reduction by locally linear embedding. Science, 2000, 290(5500): 2323–2326
9. Seung H S, Lee D D. The manifold ways of perception. Science, 2000, 290(5500): 2268–2269
10. Belkin M, Matveeva I, Niyogi P. Regularization and semi-supervised learning on large graphs. In: Proceedings of the 17th Conference on Learning Theory. 2004, 624–638
11. Wang F, Zhang C. Label propagation through linear neighborhoods. In: Proceedings of the 23rd International Conference on Machine Learning. 2006, 985–992
12. Wang F, Zhang C. Label propagation through linear neigh-

borhoods. IEEE Transactions on Knowledge and Data Engineering, 2008, 20(1): 55–67
13. Wang F, Zhang C. Semi-supervised learning based on generalized point charge models. IEEE Transactions on Neural Networks, 2008, 19(7): 1309–1311
14. Chen G, Song Y, Wang F, Zhang C. Semi-supervised multi-label learning by solving a sylvester equation. In: Proceedings of the 8th SIAM Conference on Data Mining. 2008, 410–419
15. Song Y, Zhang C. Content based information fusion for semi-supervised music genre classification. IEEE Transaction on Multimedia, 2008, 10(1): 145–152
16. Song Y, Zhang C, Lee J, Wang F, Xiang S, Zhang D. Semi-supervised discriminative classification with application to tumorous tissues segmentation of MR brain images. Pattern Analysis and Applications, 2009, 12(2): 99–115
17. Wang F, Zhang C. Fast multilevel transduction on graphs. In: Proceedings of the 7th SIAM International Conference on Data Mining. 2007, 157–168
18. Trottenberg U, Oosterlee C W, Schüler A. Multigrid. San Diego: Academic, 2001
19. Brandt A, Ron D. Multigrid solvers and multilevel optimization strategies. In: Cong J, Shinnerl J R, eds. Multilevel Optimization and VLSICAD, 2002, 1–69
20. Sharon E, Brandt A, Basri R. Fast multiscale image segmentation. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition. 2000, 1: 70–77
21. Miah M A W. Fundamentals of Electromagnetics. New Delhi: Tata McGraw-Hill Publishing Co Ltd, 1982
22. Zhu X. Semi-supervised learning with graphs. Dissertation for the Doctoral Degree. Pittsburgh: Carnegie Mellon University, 2005
23. Chung F R K, Yau S T. Discrete green's functions. Journal of Combinatorial Theory (A), 2000, 91(1): 191–214
24. Belkin M, Niyogi P, Sindhwani V. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. Journal of Machine Learning Research, 2006, 1(1): 1–48
25. Williams C, Barber D. Bayesian classification with gaussian processes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998, 20(12): 1342–1351
26. Williams C K I, Seeger M. Using the Nyström method to speed up kernel machines. In: Proceedings of Advances in Neural Information Processing Systems, Cambridge: MIT Press, 2001: 682–688
27. Fowlkes C, Belongie S, Chung F, Malik J. Spectral grouping using the Nyström method. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26(2): 214–225
28. Press W, Teukolsky S, Vetterling W, Flannery B. Numerical Recipes in C. 2nd ed. Cambridge: Cambridge University Press, 1992

Prof. Changshui ZHANG received the BS degree in mathematics from Peking University, China, in 1986, and the M.S. and Ph.D. Degrees in control science and engineering from Tsinghua University, Beijing, China, in 1989 and 1992, respectively. In 1992, he joined the Department of Automation, Tsinghua University, and is currently a Professor. His research interests include pattern recognition, machine learning, etc. He has authored more than 200 papers. He is currently an associate Editor of the Pattern Recognition journal.

Dr. Fei WANG got his BE degree in automation from Xidian University, China, 2003, and the Ph.d. degree in control science and engineering from Tsinghua University, Beijing, China, in 2008. After that, he spent one year in School of Computing and Information Sciences, Florida International University, Miami, Florida, U.S. and another year in Department of Statistical Science, Cornell University, Ithaca, New York, U.S., as postdoc research associates. He is currently with the Healthcare Transformation Group, IBM T. J. Watson Research Center, Hawthorne, New York, U.S. His major research interests include data mining, machine learning and medical informatics. He has published over 70 papers on top venues of the relevant field.