The Fourteenth International Symposium on Artificial Life and Robotics 2009 (AROB 14th '09),
B-Con Plaza, Beppu, Oita, Japan, February 5 - 7, 2009

Graph Based Semi-supervised Learning

Changshui Zhang
Department of Automation
Tsinghua University
Beijing 100084 P.R.China

Fei Wang
Department of Automation
Tsinghua University
Beijing 100084 P.R.China

Abstract

The recent years have witnessed a surge of interests in graph based semi-supervised learning. However, two of the major problems in graph based semi-supervised learning are: (1) how to set the hyperparameter in the Gaussian similarity; (2) how to make the algorithm scalable. In this talk, we will introduce a general framework for graph based learning. First, we proposed a method called linear neighborhood propagation which can automatically construct the optimal graph, second, we introduce a novel multilevel scheme to make our algorithm scalable for large data sets. The applications of our algorithm on various real world problems are also demonstrated.

1 Introduction

Semi-Supervised Learning, 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 semi-supervised learning 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 [6][22] for recent reviews). Moreover, it has been shown recently that the significance of semi-supervised learning is not limited to utilitarian considerations: *humans perform semi-supervised learning too* [8][12][21]. Therefore, to understand and improve semi-supervised learning will not only help us to get a better solver for real world problems, but also help us to to better understand how natural learning come about.

One key point for understanding the semi-supervised learning approaches is the *cluster assumption* [6], which states that [19] (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 [2][9] (note that these methods are also in accordance with the ways that the humans perceive the world [10]). The manifold based methods first assume that the data points (nearly) reside on a low-dimensional manifold (which is called *manifold assumption* in [6]), 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 [1], consequently, numerous *graph based SSL* methods have been proposed in recent years, and graph based *SSL* has been becoming one of the most active research area in semi-supervised learning community [6].

However, in spite of the intensive study of graph based *SSL* methods, there are still some open issues which have not been addressed properly, such as:

1. How to select an appropriate similarity measure between pairwise data automatically;

2. How to speed up these algorithms for handling large-scale dataset (since they usually require the computation of matrix inverse).

To address the first issue, in this talk we will first present a novel method called *Linear Neighborhood Propagation* (*LNP*) [15]. The *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. We prove theoretically that the *Laplacian* matrix of this "pasted" graph can approximate the *Laplacian* matrix of a standard weighted undirected graph. Therefore, this approximated *Laplacian* matrix can be used as a smooth matrix as in standard graph-based semi-supervised learning algorithms.

Second, 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 unsupervised learning can be viewed as a special case of semi-supervised learning, 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 graph based semi-supervised learning algorithms significantly. And we also give a theoretical guarantee on the performance of our algorithm.

2 Linear Neighborhood Propagation

In this section we will present the detailed algorithm of *linear neighborhood propagation*. First let's introduce some notations. $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_l, \mathbf{x}_{l+1}, \cdots, \mathbf{x}_n\}$ 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 $\mathcal{X}_L = \{\mathbf{x}_i\}_{i=1}^{l}$ are labeled as $t_i \in \mathcal{L}$ and the remaining points $\mathcal{X}_U = \{\mathbf{x}_u\}_{u=l+1}^{n}$ 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 [9]. Hence our objective is to minimize

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

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

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

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

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

After all the reconstruction weights are computed, we will construct a sparse matrix $\mathbf{W}$ by $W(i,j) = w_{ij}$. Intuitively, this $\mathbf{W}$ 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 pasted 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 $\mathbf{x}_i$. The label of the unlabeled data point $\mathbf{x}_u$ is determined by the sign of $f_u = f(\mathbf{x}_u)$ (let's 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. Therefore the label of $\mathbf{x}_i$ at time $m+1$ becomes

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

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

$$\mathbf{f}^{m+1} = \alpha \mathbf{W} \mathbf{f}^m + (1-\alpha)\mathbf{t} \qquad (5)$$

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 semi-supervised learning 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 $\mathbf{x}_i$ *strongly influences* $\mathbf{x}_j$ on level l means that

$$w_{ij}^l \geqslant \delta \sum\nolimits_k w_{kj}^l \tag{6}$$

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

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

Let $\mathbf{f}^0 = \mathbf{f}$ be an *classification vector* we want to solve, and $\mathbf{f}^1$ be its *corresponding classification vector* on $\mathcal{G}^1$ (hence the dimensionality of $\mathbf{f}^1$ should be equivalent to n^1, the cardinality of $\mathcal{V}^1$). The same as in other multilevel methods [13], we assume that $\mathbf{f}^0$ can be approximately interpolated from $\mathbf{f}^1$, that is[1]

$$\mathbf{f}^0 \approx \mathbf{P}^{[0,1]} \mathbf{f}^1, \tag{7}$$

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

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

An issue 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 [13], which compute w_{uv}^{l+1} by

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

It can be shown that Eq.(9) can provide a good approximation to Eq.(8) in many cases [11].

3.1.1 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

$$\begin{aligned} \mathcal{J}(\mathbf{f}^s) &= \mathbf{f}^{sT} \mathbf{P}^{[s,s-1]} \cdots \mathbf{P}^{[1,0]} \mathbf{S} \mathbf{P}^{[0,1]} \cdots \mathbf{P}^{[s-1,s]} \mathbf{f}^s \\ &\quad + \gamma \| \mathbf{P}^{[0,1]} \cdots \mathbf{P}^{[s-1,s]} \mathbf{f}^s - \mathbf{y} \|^2, \end{aligned}$$

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

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

Here $\mathbf{I}$ is the $n \times n$ identity matrix. Moreover, we have the following theorem

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

Based on the above theorem, we can compute the *initial classification vector* using Eq.(10), 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.

3.1.2 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 section 3.1, we assume that the classification vector on graph $\mathcal{G}^l$ can be linearly interpolated from $\mathcal{G}^{l+1}$, *i.e.* $\mathbf{f}^l = \mathbf{P}^{[l,l+1]} \mathbf{f}^{l+1}$. Here $\mathbf{P}^{[l,l+1]}$ is an $n^l \times n^{l+1}$ interpolation matrix subject to $\sum_j \mathbf{P}_{ij}^{[l,l+1]} = 1$.

[1] Actually, as we have analyzed after definition 3, 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*[5][13], that's the reason why we apply it here.

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 section **??**), we propose to compute $P_{iI(j)}^{[l,l+1]}$ by

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

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 $\mathbf{P}^{[l,l+1]}$ has full rank.

4 Summary

We present a general framework for graph based semi-supervised learning. The framework first use linear neighborhood propagation to automatically construct the optimal graph, then we apply a multilevel scheme to make the whole algorithm more efficient.

References

[1] Belkin, M., Matveeva, I., Niyogi, P. Regularization and Semi-supervised Learning on Large Graphs. In *Proceedings of the 17th Conference on Learning Theory*, 2004.

[2] Belkin, M., Niyogi, P. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. *Neural Computation*, vol. 15, no. 6, 1373-1396, 2003.

[3] Belkin, M. and Niyogi, P. Semi-Supervised Learning on Riemannian Manifolds. *Machine Learning*, 56: 209-239, 2004.

[4] Blum, A., and Chawla, S. Learning from labeled and unlabeled data using graph mincuts. *Proceedings of the 18th International Conference on Machine Learning*, 2001.

[5] Brandt, A., and Ron, D. Multigrid Solvers and Multilevel Optimization Strategies. in: J. Cong, J.R. Shinnerl (*Eds.*), *Multilevel Optimization and VLSICAD*, Kluwer, 2002.

[6] Chapelle, O., *et al.* (eds.): *Semi-Supervised Learning*. MIT Press: Cambridge, MA. 2006.

[7] Gloub, G. H., Vanloan, C. F. *Matrix Computations*. Johns Hopking UP,Baltimore, 1983.

[8] Graf Estes, K., Evans, J. L., Alibali, M. W., and Saffran, J. R. Can Infants Map Meaning to Newly Segmented Words? Statistical Segmentation and Word Learning. *Psychological Science*. 2006.

[9] Roweis, S. T. and Saul, L. K.: Noninear Dimensionality Reduction by Locally Linear Embedding. *Science*: vol. 290: 2323-2326, 2000.

[10] Seung, H. S. and Lee, D. D. The Manifold Ways of Perception. *Science* vol. 290: 2268-2269, 2000.

[11] Sharon, E., Brandt, A., Basri, R. Fast Multiscale Image Segmentation. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, I:70-77, South Carolina, 2000.

[12] Stromsten, S. B. Classification Learning from Both Classified and Unclassified Examples. *Ph.D. Dissertation*, Stanford. 2002.

[13] Trottenberg, U., Oosterlee, C.W., and Schler, A. *Multigrid*. with guest contributions by Brandt, A., Oswald, P. and Sten, K. San Diego, Calif. London, Academic, 2001.

[14] Vapnik, V. N. *The Nature of Statistical Learning Theory*. Berlin: Springer-Verlag, 1995.

[15] Wang, F., Zhang, C. Label Propagation Through Linear Neighborhoods. In *Proceedings of the 23rd International Conference on Machine Learning*. 2006.

[16] Wang, F., Zhang, C. Label Propagation Through Linear Neighborhoods. In *IEEE Transactions on Knowledge and Data Engineering*. 2006.

[17] Wang, F., Zhang, C. Fast Multilevel Transduction on Graphs. In *The 7th SIAM International Conference on Data Mining*. 2007.

[18] Wertheimer, M. Gestalt Theory. In W. D. Ellis (ed.), *A Source of Gestalt Psychology*, 1-11. New York: The Humanities Press. 1924/1950.

[19] Zhou, D., Bousquet, O., Lal, T. N. Weston, J., & Schölkopf, B. Learning with Local and Global Consistency. *Advances in Neural Information Processing Systems 16*. Thrun, S., Saul, L., and Schölkopf, B. (eds.), pp. 321-328, 2004.

[20] Zhu, X. *Semi-Supervised Learning with Graphs*. Ph.D. Thesis. Language Technologies Institute, School of Computer Science, Carnegie Mellon University. May, 2005.

[21] Zhu, X., Rogers, T., Qian, R., and Kalish, C. Humans Perform Semi-Supervised Classification Too. In *Proc. of the Twenty-Second AAAI Conference on Artificial Intelligence* (*AAAI*), 2007.

[22] Zhu, X. Semi-supervised learning literature survey. Technical Report 1530, Univ. Wisconsin-Madison. 2005.