The Rendezvous Algorithm: Multiclass Semi-Supervised Learning with Markov Random Walks

Arik Azran AA455@CAM.AC.UK
Engineering Department, University of Cambridge, Cambridge, CB2 1PZ, UK

Abstract

We consider the problem of multiclass classification where both labeled and unlabeled data points are given. We introduce and demonstrate a new approach for estimating a distribution over the missing labels where data points are viewed as nodes of a graph, and pairwise similarities are used to derive a transition probability matrix P for a Markov random walk between them. The algorithm associates each point with a particle which moves between points according to P. Labeled points are set to be absorbing states of the Markov random walk, and the probability of each particle to be absorbed by the different labeled points, as the number of steps increases, is then used to derive a distribution over the associated missing label. A computationally efficient algorithm to implement this is derived and demonstrated on both real and artificial data sets, including a numerical comparison with other methods.

1. introduction

Semi-supervised learning (SSL) is generally concerned with the following problem; given a set of samples $\{s_1, \ldots, s_l, s_{l+1}, \ldots, s_{l+u}\}$ and the labels of the first l samples, $\{y_1, \ldots, y_l\}$, estimate $\{y_{l+1}, \ldots, y_{l+u}\}$, the labels of the rest of the points. The underlying assumption usually made is that the data is scattered such that it is correlated with the labels. For example, Maximum Variance Unfolding (Weinberger & Saul, 2004), Laplacian Eigenmaps (Belkin & Niyogi, 2003) and Laplacian RLS (Belkin et al., 2005) assume the effective number of dimensions occupied by the data is smaller than the input space dimension (the manifold assumption). Low Density Separation (Chapelle & Zien, 2005), Transductive SVM (Joachims, 1999) and Cluster Kernel (Weston et al., 2005) assume the data forms natural groups (the cluster assumption). Kernels by Spectral Transforms (Zhu et al., 2004), Conditional Harmonic Mixing (Burges & Platt, 2006), Quadratic Criterion (Bengio et al., 2006) and Discrete Regularization (Zhou & Schlkopf, 2006) represent the data as a weighted graph on which the labels are propagated. A comprehensive review of current approaches, including an empirical comparison between them, is provided in (Chapelle et al., 2006). Other useful reading include (Zhu, 2006) for a broad literature review and (Seeger, 2002).

In this paper we introduce a new graph based algorithm for learning a distribution for each of the missing labels, which are assumed to belong to one of some $K > 1$ classes. The data is represented as a connected graph and pairwise similarities are used to derive a transition probability matrix P for a Markov random walk between nodes. P is defined such that particles are allowed to leave unlabeled points, but labeled points are set to be absorbing states of the random walk, i.e. a particle that reaches them is trapped. The probability that a missing label y_n is k is then derived from the total probability that a particle which originates at s_n is absorbed by a labeled point with label k as the number of steps grows to infinity. This work follows the spirit of (Szummer & Jaakkola, 2002) which also propagates class labels over the graph using random walks. However, their approach is opposite to ours in the sense that the random walk carries the given labels and propagates them on the graph amongst the unlabeled points, which results in the need to find a good number of steps for the walk to take. Here, we are not concerned with this parameter which is simply set to infinity. This leads to a very simple algorithm which can be implemented efficiently.

We discuss how random walks are used to estimate the missing labels in section 2. We then show in section 3 how it can be implemented efficiently, and discuss connections to other methods in section 4. Experimental results are provided in section 5.

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007.

2. Estimating Missing Labels using Markov Random Walks

The following notation is used in this paper; $\mathcal{I}_l = [1, 2, \ldots, l]$, $\mathcal{I}_u = [l+1, l+2, \ldots, l+u]$ and $\mathcal{I} = \mathcal{I}_l \cup \mathcal{I}_u$ are the indices of the labeled points, the unlabeled points and the entire data set respectively. The total number of samples is $N = l + u$. Each sample is assumed to be drawn from some metric space $\mathcal{S}$ and each label from the set $\mathcal{Y} = [1, 2, \ldots, K]$ for some integer $K > 1$. Our goal is to formulate and study an algorithm for finding a distribution over $\mathcal{Y}$, for each member of $\{y_n\}_{n\in\mathcal{I}_u}$, given the partially labeled data.

Assume $d : \mathcal{S} \times \mathcal{S} \mapsto \mathcal{R}_+$, where $\mathcal{R}_+$ is the set of all non-negative real numbers, is a metric on $\mathcal{S}$ and $w(\cdot;\sigma) : \mathcal{R}_+ \mapsto \mathcal{R}_+$ is a known upper bounded and monotonically decreasing function with parameter σ, which we refer to as the kernel. A popular choice is

$$w_{nm} = w(d(s_n, s_m); \sigma) = \exp\left(-\frac{\|s_n - s_m\|^2}{2\sigma^2}\right) , \quad (1)$$

using the Euclidean distance as a metric and σ as the varinace. The kernel (1) provides us with a numerical measure, a number between 0 and 1, for the similarity of pairs of data points. Note that $w_{nm}w_{mr} = \exp\left(-\frac{\|s_n-s_m\|^2+\|s_m-s_r\|^2}{2\sigma^2}\right)$ has the nature of geodesic distances, that is it measures the similarity between s_n and s_r 'viewed through' s_m, a property that fits in well with our algorithm. In graph based methods each data point is considered to be a node and w, which is used to weigh the edges between nodes, completely ignores the labels. Here we incorporate the given labels by considering a special graph.

Definition 1 (l-Rendezvous Graph) *Given a data set S with l labeled points, some positive integer $M \leq N$ and a metric d, for each $n \in \mathcal{I}_u$ let I_n be the set of indices of the M nearest neighbors of s_n (i.e. $d(s_n, s_m) \leq d(s_n, s_r)$ for all $m \in I_n$ and $r \notin I_n$). The l-Rendezvous graph is defined by considering each s_n to be a node and placing a directed edge from s_n to s_m, with weight W_{nm}, which is set according to*
1. $W_{nm} = w_{nm}$ for all $n \in \mathcal{I}_u$, $m \in I_n$, and
2. $W_{nm} = \delta_{nm}$ for all $n, m \in \mathcal{I}_l$,
where δ_{nm} is 1 if $n = m$ and 0 otherwise.

If the number of nearest neighbors M is chosen to be a fixed (independent of N) small number, then the resulting graph W is highly sparse. Note also that the edges are directed and that W is not symmetric.

Next, each point s_n is associated with a particle x_n that is allowed to move between data points. The location of this particle after taking t steps is denoted as $x_n(t)$. If we define the probability of x_n to move to s_m after taking a single step to be

$$\mathbb{P}(x_n(1) = s_m) = \frac{w_{nm}}{\sum_{r=1}^{N} w_{nr}} = \frac{w_{nm}}{\sum_{r\in I_n} w_{nr}} , \quad (2)$$

then the transition probability from s_n to s_m, for all $n, m = 1, 2, \ldots, N$, is given by the nm'th entry of

$$P = D^{-1}W , \quad (3)$$

where $D = \mathrm{diag}(D_1, \ldots, D_N)$ and $D_n = \sum_{r\in I_n} w_{nr}$.

P is a stochastic matrix which determines the probability for the location of $x_n(1)$ for all n. Now, consider the location of a particle $x_n(t)$ for increasing values of t. If $n \in \mathcal{I}_l$, then $x_n(t) = s_n$ for all t, since by definition s_n is an absorbing state of the markov random walk. If $n \in \mathcal{I}_u$ then $x_n(t)$ moves between points according to P until it reaches a labeled point; once it reached such a point it stays there forever. Notice that with probability one, all particles end up in one of the labeled points as $t \to \infty$, hence the term 'Rendezvous'[1].

Generally, we can expect a particle x_n that begins at some unlabeled point s_n to have higher probability of being trapped in labeled points that are either close to s_n or that have many other unlabeled points between them. In contrast, this particle is not expected to reach a labeled point that is remote and isolated from s_n. This intuition motivates the definition

$$\mathbb{P}(y_n = k) = \sum_{m\in\mathcal{I}_l : y_m = k} \lim_{t\to\infty} \mathbb{P}(x_n(t) = s_m) , \quad (4)$$

for every $k \in \mathcal{Y}$. Equation (4) assigns y_n with the label k with probability given by the total probability that the first labeled point reached by x_n has label k. Note that since a particle that begins its move from a labeled point never leaves it, then (4) is appropriate for both labeled and unlabeled points and that since every particle ends in a labeled point, then $\sum_{k=1}^{K} \mathbb{P}(y_n = k) = 1$ for all n, making it a valid distribution.

A sparse graph might consist of disconnected subgraphs. If any of these subgraphs does not include at least one labeled point as a node, then (4) is no longer a valid distribution. To avoid this we limit the discussion to the following graphs.

Definition 2 (Label Connected Graph) *Let the notation of Definition 1 hold, the graph W is said to be label connected if $\forall\ n \in \mathcal{I}_u$ there exists $m \in \mathcal{I}_l$ and some number of steps t for which $\mathbb{P}(x_n(t) = s_m) > 0$.*

It is not hard to formulate a procedure for checking if a graph is label connected in complexity $O(N)$, e.g.

[1] A prearranged meeting place

choose any labeled node and (1) flag it, (2) flag all its unflagged parents, (3) repeat step 2 for each newly flagged parent, and (4) stop if all unlabeled nodes are flagged, otherwise choose an unflagged labeled point and go to step 1. If the graph is label connected then this iterative process terminates when all unlabeled nodes are flagged, and since each node is only considered once the complexity is linear. If after the algorithm terminates there are still unlabeled nodes that are unflagged then the graph is not label connected.

3. The Rendezvous Algorithm

Equation (4) involves computing $\lim_{t\to\infty}\mathbb{P}(x_n(t) = s_m)$ for all $n \in \mathcal{I}_u$ and $m \in \mathcal{I}_l$. To compute it, recall from Markov chain theory that by applying P to itself t times we obtain P^t, whose nth row is the distribution for the location of $x_n(t)$. So, to compute (4) for all n we need to compute P^∞, which can be done by diagonalizing P. Let $\{\lambda_n, v_n\}_{n=1}^N$ be the right eigenvalues and eigenvectors of P and define V to be the matrix with v_n as its nth column, $Q = V^{-1}$ with q_n as its nth row and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ where without loss of generality we assume $\lambda_n \geq \lambda_{n+1} \forall n$.

Then, applying $P = V\Lambda Q$ to itself t times yields

$$P^t = V\Lambda^t Q = \sum_{n=1}^{N} \lambda_n^t v_n q_n, \tag{5}$$

where the second equality holds by definition. This implies that the eigenvetors of P and P^t are the same for every number of steps t and that the eigenvalues of P^t are given by the power series of the eigenvalues of P. Hence for full graphs, and for every t, P^t can be computed in $O(N^3)$ by first diagonalizing P and then using (5). However, for the graphs considered here this can be simplified. Note that for label connected graphs, W can be written as

$$W = \begin{pmatrix} W_{ll} & W_{lu} \\ W_{ul} & W_{uu} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ W_{ul} & W_{uu} \end{pmatrix}, \tag{6}$$

where $\mathbf{0}$ is a matrix of zeros and $\mathbf{I}$ is the identity matrix, each of the appropriate size. The subscript serves both as the matrix size and as an indication for the points with which it is associated. For example, W_{ul} is of size $u \times l$ and it include the weights of all edges directed from unlabeled to labeled nodes. The second equality is given by definition if W is an l-Rendezvous graph, which is the case here. Correspondingly,

$$P = \begin{pmatrix} P_{ll} & P_{lu} \\ P_{ul} & P_{uu} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ P_{ul} & P_{uu} \end{pmatrix}, \tag{7}$$

where P_{ul} consists of all the transition probabilities from all unlabled to all labeled points etc. Lemma 3 (proved in the Appendix) provides us with a key property of P that allows the derivation of Algorithm1.

Input: Partially labeled data, metric d, kernel w
Output: A distribution over the possible values of each missing label
Algorithm:
0. Set σ (12) and $M = 5$
1. Compute W (Definition 1) and P (3)
2. Compute $\{v_n\}_{n\in\mathcal{I}_l}$, the leading l eigenvectors of P.
3. For every $n \in \mathcal{I}_u$ and $k \in \mathcal{Y}$, compute

$$\mathbb{P}(y_n = k) = \sum_{m\in\mathcal{I}_l : y_m = k} \frac{v_m(n)}{v_m(m)}\ . \tag{8}$$

Algorithm 1: The Rendezvous Algorithm. If W is not label connected (step 1), increase M and recompute W. For hard labeling set $\hat{y}_n = \mathrm{argmax}_{k\in\mathcal{Y}}\,\mathbb{P}(y_n = k)$.

Lemma 3 *Assume W is a label connected l-Rendezvous graph and P is given by (3). Let λ_n be the n'th eigenvalue of P and assume without loss of generality that $\lambda_n \geq \lambda_{n+1}$ for all n. Then,*
1. $\lambda_n = 1$ for all $n = 1, 2, \ldots, l$,
2. $|\lambda_n| < 1$ for all $n = l+1, l+2, \ldots, N$.

Recall that we seek to compute P^∞ which, in our case, is of the form

$$P^\infty = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ P_{ul}^\infty & \mathbf{0} \end{pmatrix}. \tag{9}$$

Combining Lemma 3 with Equation (5), this can also be writen as

$$P^\infty = \sum_{n=1}^{l} v_n q_n. \tag{10}$$

Although generally we need to find $\{v_n\}_{n=1}^N$ to compute $\{q_n\}_{n=1}^l$, the following Lemma shows how in the case of l-Rendezvous graph this can be done more efficiently, where only $\{v_n\}_{n=1}^l$ needs to be computed.

Lemma 4 *Let the definitions and notation of Lemma 3 hold. Then, for every $n \in \mathcal{I}$ and $m \in \mathcal{I}_l$*

$$P_{nm}^\infty = \frac{v_m(n)}{v_m(m)}. \tag{11}$$

Remarkably, Lemma 4 states that to compute (4), all that is needed to find is the first l right eigenvectors of P. In addition, we know that the top entries of these eigenvectors are sparse (see Appendix B) and that each of the corresponding eigenvalues equals 1, which can be used to efficiently initialize the eigensolver. Since

we use a sparse graph, this can be done efficiently in complexity no worst than $O(N^2)$ and has the potential of reducing this comlexity even further.

4. Connections to Other Methods

The problem of unsupervised graph partitioning using random walks was considered in (Meila, 2001; Meila & Shi, 2001) where it was shown that using the leading eigenvectors of P can be viewed as a relaxation of the NP hard problem of minimizing the multiway normalized cut, a multiclass generalization of the normalized cut (Shi & Malik, 2000). In (Azran & Ghahramani, 2006) it was shown that considering P^t instead of P can be used to estimate the number of clusters without losing the powerful theoretical guarantees of the multicut Lemma (Meila, 2001). This is essentially the framework we used here, with the difference that some data points are labeled. Thus, from an unsupervised learning point of view the Rendezvous algorithm can be interpreted as solving a constrained clustering problem where points with the same labels must (and points with different labels must-not) be clustered together by searching for a small balanced multiway cut of the graph which satisfy these constraints. Spectral Graph Transducer (Joachims, 2003) was motivated as a technique for constrained graph partitioning and can be viewed as a modification of the Transductive SVM (Joachims, 1999) which removes the need to set the cut size a priory, or as an optimization problem of a regularized loss function based on the graph Laplacian (Zhu, 2006). Except for sharing similar motivation, SGT and our algorithm differ in the graph representation of the data (symmetric vs. nonsymmetric with absorbing states), the label space (binary vs. multiclass) and in the problem formulation (optimization problem vs. computation of graph properties (4),(8)).

In (Szummer & Jaakkola, 2002) random walks are used directly for multiclass SSL, but as mentioned in the introduction our method completely avoids the crucial problem of choosing the number of steps t and is simpler to implement. In (Zhou & Schlkopf, 2004) the random walk was assigned with some teleporting probability and the commute time was used to classify unlabeled points. Our algorithm is different from this method in setting labeled points to be absorbing states and by allowing any number of classes. Additionally, our method only requires finding a small number l of the leading eigenvectors, whereas (Zhou & Schlkopf, 2004) requires inverting a full size matrix.

An interesting connection can be made with Laplacian based methods. For example, SGT and the grouping algorithm in (Shi & Malik, 2000) are based on the solution to the Laplacian generalized eigenvalue problem $(D - W)u = \mu Du$, which by simple matrix manipulation can be represented as $D^{-1}Wv = (1 - \mu)v$, and the close relation with random walks is immediate. A useful property of the spectral decomposition of the Laplacian $D - W$ is the enforcement of smoothness on functions defined over the graph nodes. This motivated several SSL algorithms, including (Zhu et al., 2003) which consider binary labels and (Zhu et al., 2004) for multiclass problems. It is worth pointing out that our method is not as close as it might seem to methods based on the graph Laplacian such as the above. To see why, recall that we use a nonsymmetric matrix W, which violates the condition for the Laplacian to encourage smooth functions over the graph. The behavior of the Rendezvous algorithm is better understood by considering the diffusion property of random walks on the graph rather than the smoothness property imposed by the graph Laplacian.

5. Experiments

5.1. Nonconvex and Overlapping Clusters

Consider the data set in Figure 1, comprised of a Gaussian nested in two noisy rings. The noise level ranges from low, where the cluster assumption holds and the three groups are clearly separable, up to a level where the group structure completely vanishes. We applied our algorithm to this data set under various noise levels and for different number and location of the labeled points. The experiment setup was as follow: (1) generate 350 data points (20 from the Gaussian, 100 from the inner ring and 230 from the outer ring), (2) label some points, (3) apply the Rendezvous algorithm. This was repeated independently 100 times for each of the following procedures for labeling points in step (2): (i) label a single fixed point for each group, always on the origin for the Gaussian and where the rings intersects with the positive x axis, (ii) label ten fixed points for each group, near the origin for the Gaussian and uniformly spaced on each of the rings, and (iii) randomly choose $\sim 5\%$ of the points from each group (1 from the Gaussian, 5 from the inner ring and 10 from the outer ring) and label them.

The results are presented with boxplots which has lines at the lower quartile, median, and upper quartile values and the whiskers indicate the location of the most distant data point, within a distance of the *interquartile range* (IQR), from the edge of the box. The results for the different schemes of labeling points are presented in the top row of Figure 1. The data plots on the bottom show the labeled points wrapped with a square, the hard labeling by the shape of the point

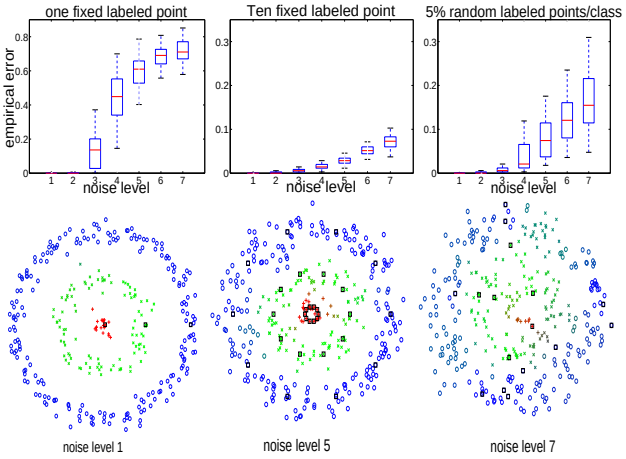

Figure 1. Algorithm 1 applied to data comprised of nested groups with seven different noise levels. The labeled subset was chosen in three different schemes, each demonstrated in the bottom row under the relevant boxplot. Top row - results for 100 independen draws of datasets (note the different scales in the y-axis). Bottom row - results for some individual datasets, showing the labeled points as boxed, the hard labeling as '+','x' or 'o' and the soft labeling a mix of RGB (see text for discussion).

('+','x' or 'o'), and the distribution over $\mathcal{Y}$ for each label by the color (we used $|\mathcal{Y}| = 3$ to allow the representation of the distributions over missing labels as a mix of RGB) for some data sets.

What can be learnt from the results of this experiment? First, if the cluster assumption holds, a single point can be sufficient to propagate the labels with no errors. Second, if the groups overlap, the important thing is that the labeled points 'cover' the region of the input space populated by points which were sampled from the same class. The intuition is that each labeled point has the highest probability of trapping the particles that start their move from nearby points, and thus the location of the labeled points could be more important than their number. Another important property of the results is their variability. Notice how the whiskers of the boxplot extend further as the noise level increases, presenting another piece of evidence for the importance of the location, and not necessarily the number, of the labeled points.

5.2. On the Parameter Setting

The algorithm involves choosing a kernel and setting its parameter σ. For sparse graphs the number of edges leaving each data point, M, also needs to be initialized. We discuss here how the setting of these parameters can be expected to affect the results of the

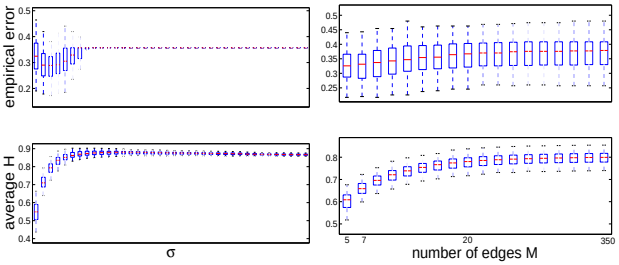

Figure 2. Algorithm 1 applied to the data set in Figure 3 with noise level 7 (no group structure) and 5% labeled points, using a wide range of σ and M. Left - boxplot of the percentage of misclassified points over a hundred different data sets (to test hard labeling), right - boxplot of the empirical entropy (13) for the same data sets (to test soft labeling). See text for a discussion.

algorithm, beginning with M. If the graph is fully connected ($M = N$), then a particle can transition from every unlabeled to any labeled node in a single step. However, if it is located in a distance from all labeled points, then it has higher probability of transitioning to its unlabeled neighboring nodes, and in the following step to their neighbors, until it reaches a labeled point. Hence, allowing each particle to transition only to some small number of neighbors, while ensuring this results in a label connected graph, leads to a natural construction of the graph which is conceptually compatible with our algorithm.

Now, consider the setting of σ for the full graph. If we decrease this parameter to the limit $\sigma \to 0$, then with probability one each particle x_n will only move to the nearest neighbor of s_n, lets call this node s_m, when taking the first step. The second step will be to the nearest neighbor of s_m (which needs not be s_n) etc. This is clearly not a desirable setting since it assigns each missing label with a single class with probability 1, which implies the loss of information that exists in soft classification. In the other extreme, as $\sigma \to \infty$, the graph is no longer a reliable representative of the data. In this case, $P_{mn} = \frac{1}{N}$ for every unlabeled point and there is no recollection to the data structure. The approach implemented in our algorithm is setting the parameters by placing a small number of edges (we usually used $M = 5 \div 10$) leaving each unlabeled node and σ to be of order of the the smallest distances between pairs of points. To formalize it, σ is set to be

$$\sigma = \text{median}(\mathcal{D}) \ , \tag{12}$$

where $\mathcal{D}$ is the set of smallest 1% of all pairwise distances $\{d(s_n, s_m)\}_{n,m\in\mathcal{I}}$.

How much changing σ and M affects the results of the

Algorithm	g241c	g241d	Digit1	USPS	COIL	BCI	Text	g241c	g241d	Digit1	USPS	COIL	BCI	Text
				$l = 10$							$l = 100$			
1-NN	47.88	46.72	13.65	16.66	63.36	49.00	38.12	43.93	42.45	3.89	5.81	17.35	48.67	30.11
SVM	47.32	46.66	30.60	20.03	68.36	49.85	45.37	23.11	24.64	5.53	9.75	22.93	34.31	26.45
MVU+1-NN	47.15	45.56	14.42	23.34	62.62	47.95	45.32	43.01	38.20	2.83	6.50	28.71	47.89	32.83
LEM +1-NN	44.05	43.22	23.47	19.82	65.91	48.74	39.44	40.28	37.49	6.12	7.64	23.27	44.83	30.77
QC+CMN	39.96	46.55	9.80	13.61	59.63	50.63	40.79	22.05	28.20	3.15	6.36	10.03	46.22	25.71
Discrete reg.	49.59	49.05	12.64	16.07	63.38	49.51	40.37	43.65	41.65	2.77	4.68	9.61	47.67	24.00
TSVM	24.71	50.08	17.77	25.20	67.50	49.15	31.21	18.46	22.42	6.15	9.77	25.80	33.25	24.52
SGT	22.76	18.64	8.92	25.36	-	49.59	29.02	17.41	9.11	2.61	6.80	-	45.03	23.09
Cluster-Kernel	48.28	42.05	18.73	19.41	67.32	48.31	42.72	13.49	4.95	3.79	9.68	21.99	35.17	24.38
Data dep. reg.	41.25	45.89	12.49	17.96	63.65	50.12	-	20.31	32.82	2.44	5.10	11.46	47.47	-
LDS	28.85	50.63	15.63	17.57	61.90	49.27	27.15	18.04	23.74	3.46	4.96	13.72	43.97	23.15
Laplacian RLS	43.95	45.68	5.44	18.99	54.54	48.97	33.68	24.36	26.46	2.92	4.68	11.92	31.36	23.57
CHM	39.03	43.01	14.86	20.53	-	46.90	-	24.82	25.67	3.79	7.65	-	36.03	-
Renezvous-L	50.01	49.66	14.65	19.51	59.75	49.83	47.55	46.00	43.76	2.14	7.79	10.66	46.67	31.76
Renezvous-H	50.02	49.68	15.79	19.60	67.12	50.28	48.39	46.03	43.80	2.16	8.20	17.90	47.03	32.26
Renezvous	50.02	49.68	15.19	19.60	64.32	50.19	47.79	46.00	43.77	2.15	8.11	10.66	46.83	31.83

Table 1. Comparison with other methods, showing 12-fold cross-validation performance for $l = 10/100$ labeled points (BCI has a total of 400 points, the rest 1500). For each dataset σ was set to six different values, one was $\min_{n\neq m} d(s_n, s_m)$ and the rest according to (12) where $\mathcal{D}$ comprised $[0.2, 0.6, 1, 2, 4]$ % of the smallest pairwise distances. The row L (H) shows the lowest (highest) error obtained by these settings of σ; the bottom row shows the performance of Algorithm 1 where σ is set automatically (step 0), clearly indicating the algorithm is robust to the exact setting of the sole parameter.

algorithm when applied to the data set in Figure 1 with noise level 7, is described in Figure 2. σ was (nonuniformly) changed over the range $\min_{n\neq m} d(s_n, s_m)$ and $\max_{n,m} d(s_n, s_m)$ and M over the range between 5 and 350 (i.e. N). The results are summarized with two different measurements. First, the percentage of misclassified points (empirical error) measures only the bottom line of the algorithm. Additionally, to measure the sensitivity of the soft classification to the setting of the parameters, we computed the empirical entropy

$$H = \frac{1}{N}\sum_{n=1}^{N} H_n, \tag{13}$$

where $H_n = -\sum_{k=1}^{K} r_{nk} \ln r_{nk}$ and $r_{nk} = \mathbb{P}(y_n = k)$. The results for using a fully connected graph with 5% of the points which are randomly chosen from each class and labeled, for different settings of σ, are shown on the left of Figure 2. It clearly shows a minimum of the empirical error is achieved for a small σ (though not the smallest), which is compatible with (12). It is interesting to notice that as σ grows, the variability of the results decays and every dataset results in exactly the same empirical error. The reason is that large σ results in P_{mn} being approximately equal for all pairs of s_n, s_m, which in turn leads to $r_{n1} \approx 1/16, r_{n2} \approx 5/16$ and $r_{n3} \approx 10/16$ for all n, $H \approx 0.83$ and empirical error that approximately equals $\frac{120}{350} \approx 0.34$. For a small σ, the algorithm performance improves as the number of edges decreases (right side of Figure 2). The reason is that by only allowing particles to move to the nearest neighbors, they have higher probability of ending in a neighboring labeled point rather than a remote one.

5.3. Comparison with Other Methods

In (Chapelle et al., 2006), a collection of state of the art algorithms for SSL are discussed by their authors and applied to a set of benchmark datasets. These datasets, including a detailed description of how they were obtained, are available at `http://www.kyb.tuebingen.mpg.de/ssl-book`, and a brief description only is given here for completeness. The datasets g241c and g241d were generated by some processing of samples from a mixture of Gaussians. Digit1 contains images of '1' and the label is set according to the tilt of the digit in the image. USPS contains 150 images of each of the ten digits from the famous USPS set, with classes '2' ∪ '5' vs. the rest (i.e. classes are imbalanced with relative size 1:4). COIL is a processed version of the Columbia object image library (COIL-100), with six labels. All these data sets are comprised of 1500 data points, each having 241 dimensions. The BCI (brain computer interface) data set includes 400 data points, each living in 117 dimensions representing model parameters fitted to EEG images of a person performing 400 trials of imagining moving the left (class -1) or right (class +1) hand. Lastly, Text is the 5 `comp.*` groups from the Newsgroup data set with labels `ibm` category vs. the rest. It include 1500 data points in 11,960 dimensions.

The results for two sets of experiments, one for smaller ($\sim$ 0.7%) and the other for larger ($\sim$ 7%) ratio of $\frac{l}{N}$, are summarized in Table 1. The performance of all SSL algorithms (with parameters set to optimize the cross-validation error and thus needs to be compared with the row Rendezvous-L) were reported by

their authors in (Chapelle et al., 2006). Overall, the Rendezvous algorithm demonstrates a competitive behavior. In particular, for Digit1 with $l = 100$ it outperforms all of the algorithms and for both setups with COIL it does better than most of them. On the other hand, the results for Text, BCI and USPS can be viewed as average, and when applied to g241c and g241d it recovers almost no class structure. Interestingly, the results stay practically unchanged for different settings of σ (they are somewhat less stable with COIL, probably because it is multiclass). This supports the discussion in section 5.2, especially (12), and suggests our algorithm can be used automatically, which is crucial when dealing with real data.

Important information which was not reported by the authors, and thus is not in the table, is the variability of the results over different choices of the labeled subsets. For example, for Digit1 with $l = 10$ the lowest percentage of misclassified points achieved by the Rendezvous algorithm is 3.36% and the highest is 41.74%. Accompanied by the results in Figure 1, which exhibits similar behavior, this suggests the reason for the large variability is that for labeled subsets for which the empirical error is high, a small number of labeled points of one class might be located away from the boundary between classes while labeled points from the other class are closer to the boundary, which results in many points being mislabeled. This can also be used as an indication to the (in)validity of the cluster assumption.

5.4. On the Convergence of the Algorithm

While a thorough treatment of the asymptotic convergence is beyond the scope of this paper, we provide an experimental evaluation of how the algorithm can be expected to behave as the size of the dataset increases. Consider the dataset in Figure 3 which is comprised of a mixture of four Gaussians. Note that the covariance matrices of the Gaussians are different, which results in the true labeling not being distance related but rather cluster dependent. Note also there is some overlap of the different classes; using Gaussians allowed the computation of the lowest achievable error (Bayes error) which for this case equals $\sim 1\%$. The performance of the algorithm after repeating the experiment 100 times (summarized on the right of Figure 3) suggest the following four observations, some also support relevant previous discussion. First, the variability of the performance decrease as the number of points increases. Second, the algorithm demonstrates fast convergence, where even for a small number of points the results are close to the asymptotic. Third, the empirical error is close to the Bayes error. And fourth, if the cluster assumption holds, then a single point is sufficient to propagate the labels on the graph successfully.

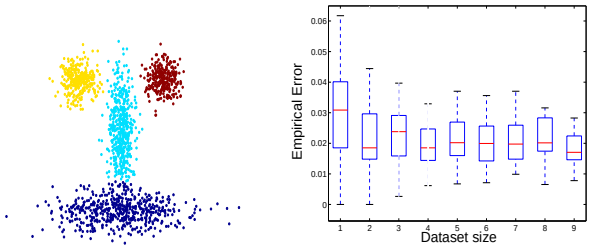

Figure 3. Convergence evaluation of Algorithm1. Right - results for nine dataset sizes ($N = 40 + 200z$ for $z = 0, 2, \ldots, 8$) with only a single labeled point from each class ($l = 4$); for each size Algorithm 1 was applied to 100 independently drawn datasets. Left - an example dataset (with $N = 1040$).

6. Conclusions

We have introduced a new algorithm for estimating a distribution over each of the missing labels in the presence of both labeled and unlabeled data. An efficient implementation was derived and a comparison of the algorithm with other methods was provided. One interesting directions for future work is developing algorithms for a clever selection of the labeled data set. Our experiments clearly indicate that the location of the labeled points within the data set is as important as the size of the labeled set, and thus choosing these points to optimize performance is of great importance. Another direction is choosing the kernel based on the data. In all our experiments we used the very simple Gaussian kernel with the identity covariance matrix, which probably does not exploit all the similarity information conveyed in the data points. Additionally, there is no reason why a single kernel should be used, instead of different kernels for different groups of data points. Another interesting direction is the search for the eigenvectors of P. Since P is highly sparse and we only look for a small number of eigenvectors, all of which correspond to unit eigenvalues and posses some special properties, it is possible that especially efficient methods for finding these eigenvectors can be derived.

A. Proof of Lemma 3

Note that $|\lambda_n v_n(m)| = |\sum_i P_{mi} v_n(i)| \leq \sum_i P_{mi} |v_n(i)| \leq \max_j |v_n(j)| \sum_i P_{mi} = \max_j |v_n(j)|$ $\forall n, m$. Thus, for every n, choosing $m = \text{argmax}_j |v_n(j)|$ results in $|\lambda_n| \leq 1$. Next, recall that if the eigenvalues and eigenvectors of P are $\{\lambda_n, v_n\}_{n=1}^N$, then those of P^t are $\{\lambda_n^t, v_n\}_{n=1}^N$ (see (5)) . Additionally, notice that $P^\infty P^\infty = P^\infty$, which

implies that each columns of P^∞ is also its eigenvector with eigenvalue 1. Finally, since P has exactly l absorbing nodes and is label connected, then with probability 1 every particle ends in one of these nodes as t grows to infinity. This implies that $P^\infty_{nm} = \delta_{nm}$ for all $n \in \mathcal{I}_l$ and $P^\infty_{nm} = 0$ for all $n \in \mathcal{I}$ and $m \in \mathcal{I}_u$ (see (9)), hence P^∞ has l linearly independent columns and is thus of rank l. To conclude, from the condition that $\lambda^\infty_n = 1$ for all $n = 1, 2, \ldots, l$ follows the first part of the Lemma and from the condition that $\lambda^\infty_n = 0$ for all $n = l+1, l+2, \ldots, N$ follows the second part. □

B. Proof of Lemma 4.

Define

$$V = \begin{pmatrix} V_1 & V_2 \\ V_3 & V_4 \end{pmatrix} \tag{14}$$

where V_1, V_2, V_3 and V_4 are of size $l \times l, l \times u, u \times l$ and $u \times u$ respectively. Also, note that each column of P^∞ is an eigenvector of P too (see Appendix A). This, combined with (9), implies that V_1 is diagonal. Additionally, since P has the form of (7), then it is not hard to show that v_n, for each $n \in \mathcal{I}_u$, has zeros on the first l entries, and thus $V_2 = \mathbf{0}$. Thus, using the matrix inversion lemma [2],

$$Q = V^{-1} = \begin{pmatrix} V_1^{-1} & \mathbf{0} \\ -V_4^{-1} V_3 V_1^{-1} & V_4^{-1} \end{pmatrix} . \tag{15}$$

Finally, combining (10) with (15), while keeping in mind that the submatrix V_1 is diagonal, complete the proof of Lemma 4. □

References

Azran, A., & Ghahramani, Z. (2006). Spectral Methods for Automatic Multiscale Data Clustering. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation.*

Belkin, M., Niyogi, P., & Sindhwani, V. (2005). On manifold regularization. *International Workshop on Artificial Intelligence and Statistics (AISTATS).*

Bengio, Y., Delalleau, O., & Roux, N. L. (2006). Label propagation and quadratic criterion. *Semi-Supervised Learning, Eds. O. Chapelle, B. Schlkopf and A. Zien.* MIT Press.

Burges, C., & Platt, J. (2006). Semi-supervised learning with conditional harmonic mixing. *Semi-Supervised Learning, Eds. O. Chapelle, B. Schlkopf and A. Zien.* MIT Press.

Chapelle, O., Schölkopf, B., & Zien, A. (Eds.). (2006). *Semi-Supervised Learning.* Cambridge, MA: MIT Press.

Chapelle, O., & Zien, A. (2005). Semi-supervised classification by low density separation. *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics.*

Joachims, T. (1999). Transductive inference for text classification using support vector machines. *International Conference on Machine Learning (ICML).*

Joachims, T. (2003). Transductive learning via spectral graph partitioning. *International Conference on Machine Learning (ICML).*

Meila, M. (2001). The multicut lemma. *UW Statistics Technical Report 417.*

Meila, M., & Shi, J. (2001). A random walks view of spectral segmentation. *Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS).*

Seeger, M. (2002). *Learning with Labeled and Unlabeled Data* (Technical Report). University of Edinbrough.

Shi, J., & Malik, J. (2000). Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Szummer, M., & Jaakkola, T. (2002). Partially Labeled Classification with Markov Random Walks. *Advances in Neural Information Processing Systems (NIPS),14.*

Weinberger, K. Q., & Saul, L. K. (2004). Unsupervised learning of image manifolds by semidefinite programming. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Weston, J., Leslie, C. S., Ie, E., Zhou, D., Elisseeff, A., & Noble, W. S. (2005). Semi-supervised protein classification using cluster kernels. *Bioinformatics.*

Zhou, D., & Schlkopf, B. (2004). Learning from labeled and unlabeled data using random walks. *Pattern Recognition, Proceedings of the 26th DAGM Symposium.*

Zhou, D., & Schlkopf, B. (2006). Discrete regularization. *Semi-Supervised Learning, Eds. O. Chapelle, B. Schlkopf and A. Zien.* MIT Press.

Zhu, X. (2006). *Semi-Supervised Learning Literature Survey* (Technical Report 1530). Computer Science, University of Wisconsin-Madison.

Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. *Advances in Neural Information Processing Systems (NIPS),16.*

Zhu, X., Lafferty, J., Ghahramani, Z., & kandola, J. (2004). Nonparametric transforms of graph kernels for semi-supervised learning. *International Conference on Machine Learning (ICML).*

[2] $\begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} + A^{-1}BZCA^{-1} & -A^{-1}BZ \\ -ZCA^{-1} & Z \end{pmatrix}$ where A, B, C and D are matrices of appropriate size and $Z = \left(D - CA^{-1}B\right)^{-1}$.