DIRICHLET MIXTURES OF GRAPH DIFFUSIONS FOR SEMI SUPERVISED LEARNING

Christian Walder

Informatics and Mathematical Modelling
Technical University of Denmark
DK-2800 Kongens Lyngby.

ABSTRACT

Graph representations of data have emerged as powerful tools in the classification of partially labeled data. We give a new algorithm for graph based semi supervised learning which is based on a probabilistic model of the process which assigns labels to vertices. The main novelty is a non parametric mixture of graph diffusions, which we combine with a Markov random field potential. Markov chain Monte Carlo is used for the inference, which we demonstrate to be significantly better in terms of predictive power than the *maximum a posteriori* estimate. Experiments on bench-mark data demonstrate that while computationally expensive our approach can provide significantly improved predictions in comparison with previous approaches.

1. INTRODUCTION

In transductive semi-supervised learning (s.s.l.) we are given examples $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m$, only some of which (the *labeled* examples) come with corresponding categorical class labels y_i, and we wish to infer the class labels of the unlabeled examples. A large amount of work has been done here, for a survey of the different methods we recommend [1, 2], as well as the nice basic discussion of the problem in [3]. Due to space limitations we can only summarize here the insights in the existing literature which are most relevant in motivating our algorithm.

Discriminative models are not applicable to the s.s.l. setting, since if we model the conditional $p(y|\boldsymbol{x})$ directly then each y_i is independent of the other $\boldsymbol{x}_j$ given $\boldsymbol{x}_i$. Even complete knowledge of the marginal $p(\boldsymbol{x})$ is irrelevant when inferring the label of a given point. Hence, for s.s.l. it is necessary to employ generative models [3], *i.e.* those which model class conditional densities. Existing discriminative models need to be modified to benefit from unlabeled data, for example by including new likelihood or loss terms for the unlabeled examples. The transductive support vector machine (s.v.m.) [4] is an algorithm which can be seen from this perspective, since it equivalent to a normal s.v.m. with additional loss terms for the unlabeled examples. The popularity of the s.v.m. is widely attributed in part to the convexity of the formulation. The transductive s.v.m. does not have this advantage however, since the unlabeled loss term must favor classifying the unlabeled points with a large margin on either one of two disjoint sides of the decision boundary. The inevitability of this type of non-convexity has lead to the development of various dedicated non-convex optimization strategies for s.s.l., as reviewed by [5].

We propose an algorithm which represents the data as a graph and models the class conditional densities over the vertices. Our approach is most closely related to the method of [6], which samples from a discrete Markov random field distribution over the labels. We essentially combine that idea with a model of the class conditional densities as a Dirichlet mixture of probability mass functions derived from a Laplacian graph diffusion. This diffusion mixture model could also be used to model probabilities on any graph, including the hyper-link graph of the world wide web.

The paper is organized as follows. We review graph based regularization in section 2, including graph construction, the graph Laplacian, and the diffusion process we use to construct generic mixture components for our mixture model. In section 3 we introduce our probabilistic model for the labeling of graph vertices based on these components, first as an analogous continuous model in subsection 3.1, then in the discrete form in subsection 3.2. In section 4 we describe our sampling algorithm, before providing results in section 5 and concluding in section 6.

2. GRAPHS AND AN ANALOGY WITH $\mathbb{R}^N$

We define a *graph* $G = (V, E)$ to be a finite set of *vertices* $V := v_1, v_2, \ldots, v_m$ and a set $E \subseteq V \times V$ of *edges*. We consider weighted graphs, so that there is a function w which takes an $e \in E$ and maps it to an associated *edge weight* $w(e) \geq 0$, and restrict to the case of symmetric weights, so that $w([v,u]) = w([u,v]), \forall\, [u,v] \in E$. w may be interpreted as measuring the strength of the connection or similarity between the vertices constituting an edge. For convenience we define the *degree* of a vertex by $d(v) = \sum_{u:[u,v]\in E} w([u,v])$. Given data points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m \subseteq \mathbb{R}^d$, a typical method of constructing a graph for s.s.l. is

the following. Associate the $\boldsymbol{x}_i$ one to one with the vertices v_i. Connect two vertices by an edge e iff the data point associated with either one is a *k-nearest neighbor* of that associated with the other. Define $w([v_i, v_j]) = (1 - \delta_{i,j}) \exp(-\omega \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$,for all $[v_i, v_j] \in E$, where δ is the Kronecker delta, and both k and ω are parameters which we discuss in section 5. Such a graph is effectively free of self connections, since $w([v, v]) = 0, \forall u \in V$.

2.1. Regularisation on Graphs

We provide a brief motivating overview of discrete regularization, for a more precise and detailed exposition we recommend [7]. Typically in graph based s.s.l. one defines a function space $\mathcal{H}(V)$ along with an inner product $\langle f, g \rangle_{\mathcal{H}(V)} := \sum_{v \in V} f(v) g(v)$. Many supervised learning algorithms can now be applied to the semi-supervised case by replacing their usual function space with $\mathcal{H}(V)$. For example a common approach solves for

$$f^\star = \arg \min_{f \in \mathcal{H}(V)} \lambda \Omega(f) + \sum_{i \in \mathcal{L}} \left(f(\boldsymbol{v}_i) - y_i\right)^2, \quad (1)$$

where $\mathcal{L}$ is the labeled set and Ω is defined via the graph Laplacian. This least-squares approach is already rather effective given an appropriate graph based regularizer Ω. This idea has been extended in a number of ways, see *e.g.* [1, 2].

2.2. A Discrete/Continuous Analogy

It turns our that many typical choices of regularization operator Ω permit direct analogies to the continuous case, in which $\mathcal{H}(V)$ is identified with $\mathbb{R}^n$. In particular, letting the i-th element of $\boldsymbol{f}$ be $f(v_i)$, and letting L stand for both the operator $\mathcal{H}(V) \to \mathcal{H}(V)$ and the $m \times m$ matrix, the commonly used normalized graph Laplacian L is defined by $\langle f, Lf \rangle_{\mathcal{H}(V)} = \boldsymbol{f}^\top L \boldsymbol{f}$, which by definition equals

$$\sum_{[u,v] \in E} w([u, v]) \left(f(u) - f(v)\right)^2 / \left(\sqrt{d(u) d(v)}\right),$$

implying $L = \text{diag}\,(S\boldsymbol{e}) - S$ and, for W composed of elements $w([u, v])$, $S = \text{diag}\,(W\boldsymbol{e})^{-\frac{1}{2}}\, W \text{diag}\,(W\boldsymbol{e})^{-\frac{1}{2}}$. Multiplication by L is the graph analogy of applying the usual continuous Laplacian operator on $\mathbb{R}^n$, with the two formally coinciding point-wise in the limit under certain conditions [8]. This connection is well known, and motivates two key components of our model as analogies of standard techniques used in $\mathbb{R}^n$. One of these is the choice of discrete mixture components, discussed separately in the following subsection 2.4. The other more immediate one which we discuss in the following subsection 2.3 is the function regularizer, which we use to define a prior over functions on the graph.

2.3. Priors Over Functions on the Graph

A classic regularizers in the continuous case the second order thin plate energy given by $\langle f, \nabla^2 f \rangle$. By our analogy, this leads naturally to a prior distribution over functions on the graph defined by the density $p(\boldsymbol{f}|L) \propto \exp(-\gamma \boldsymbol{f}^\top L \boldsymbol{f})$, which is Gaussian in $\boldsymbol{f}$ and may be thought of as the graph analogy of a Gaussian process prior over continuous functions, with inverse covariance matrix L.

2.4. Diffusions as Mixture Components

As discussed in section 1, s.s.l. cannot occur in discriminative models, so we choose to model the class conditional densities over the vertices of the graph. Generic mixture components are appropriate for this, and since a Gaussian density on $\mathbb{R}^n$ can be defined in terms of the continuous Laplacian operator via the heat equation $\frac{\partial}{\partial \tau} \psi = \mu \nabla^2 \psi$, as our discrete mixture components we replace the Laplacian in this definition with the graph Laplacian, to obtain the graph heat kernel $K_\tau := \exp(-\tau L/2)$, where $\exp(A) := \lim_{s \to \infty} (I + A/s)^s$ is the matrix exponential, and the symmetric matrix K contains the values of the heat kernel between all pairs of vertices [9]. This is the result of a diffusion process defined by applying the discrete analog of the (continuous) heat equation $\frac{\partial}{\partial t} K_\tau = L K_\tau$ to a function which at $\tau = 0$ vanishes on all but one vertex. Hence, as our m discrete probability mass function (p.m.f.)'s we choose the normalized columns of K_τ, so that for ξ distributed according to the j-th such p.m.f., $p(\xi = i) := (D_\tau)_{i,j}$ where $D_\tau := \text{diag}\,(K_\tau \mathbf{1})^{-1} K$, and $\mathbf{1}$ is a vector of ones.

3. DIRICHLET MIXTURE OF DIFFUSIONS

3.1. Analogous Continuous Model

To motivate our model we start with an analogous but more familiar one in which the data generating distribution is supported on $\mathbb{R}^n$ rather than V, the vertices of our graph. In recent years infinite mixtures models have become a popular choice for flexible density modeling from generic mixture components. For example we may build a class conditional infinite Gaussian mixture model including the labeling process y_i for k classes by

$$\begin{aligned} y_i|\eta_1, \eta_2, \cdots, \eta_k &\sim \text{Discrete}(\eta_1, \eta_2, \cdots, \eta_k) \\ G_i &\sim \text{DP}(\alpha, H) \\ \boldsymbol{m}_i|y_i, G_1, G_2, \ldots, G_k &\sim G_{y_i} \\ \boldsymbol{x}_i|\boldsymbol{m}_i, \sigma &\sim \text{Normal}(\boldsymbol{x}_i|\boldsymbol{m}_i, \sigma), \end{aligned} \quad (2)$$

where $\text{DP}(\alpha, H)$ denotes the Dirichlet process with concentration parameter α and base measure H. We use the notation $\text{Normal}(\boldsymbol{x}_i|\boldsymbol{m}_i, \sigma)$ for the normal or Gaussian distribution in $\boldsymbol{x}_i$ with mean $\boldsymbol{m}_i$ and isotropic variance σ, and

Discrete$(\eta_1, \eta_2, \cdots, \eta_k)$ for the discrete distribution with probability masses $\eta_1, \eta_2, \ldots, \eta_k$. Hence each class conditional density is given by an infinite Gaussian mixture model with its own parameters.

3.2. Discrete Model

To model the labelling process of a fixed graph, we propose a graph analog of (2) based on the analogy developed in subsection 2.2. This means we identify $\mathbb{R}^n$ with the vertices of the graph, so that rather than sampling points $\boldsymbol{x}_i \in \mathbb{R}^n$ from an infinite mixture of Gaussian probability density functions, we generate vertex indices $\xi_i \in \{1, 2, \ldots, m\}$ of the graph by sampling from a finite mixture of the diffusion p.m.f.'s defined in subsection 2.4. Further identifying the means $\boldsymbol{m}_i$ with vertex indices β_i (indexing the diffusion p.m.f.'s which generated the ξ_i) leads to

$$\begin{aligned} y_i|\eta_1, \ldots, \eta_k &\sim \text{Discrete}(\eta_1, \cdots, \eta_k) \\ G_i &\sim \text{Dirichlet}(\alpha/m, \ldots, \alpha/m) \\ \beta_i|y_i, G_1, \ldots, G_k &\sim G_{y_i} \\ \xi_i|\beta_i, \tau &\sim \text{Discrete}((D_\tau)_{1,\beta_i}, \ldots, (D_\tau)_{m,\beta_i}). \end{aligned} \tag{3}$$

The above finite Dirichlet-Multinomial is known to converge to the DP of (2) under an infinite limit [10]. Instead of the infinite mixture as in subsection 3.1, this class conditional density is a finite mixture of the m p.m.f.'s stored in the columns of D_τ. This is a generative model for the labeling of a given graph. If the graph itself is constructed from the (labeled and unlabeled) data however, then the overall inference procedure should be considered transductive, and since the graph grows with the data, non parametric.

3.3. Markov Random Field Potential

In s.s.l. problems where most y_i are unobserved, this model may lead to undesirably large overlap in the class conditional densities. The model discourages only by way of the *clustering* tendency of the Dirichlet process, which favors fewer components and hence non-overlapping distributions. In this section we propose a modification of the model which is designed to reduce this problem. The problem is a typical failure mode of mis-matched generative models, and has been encountered before in the literature. The difficulty appears to be especially acute in s.s.l. problems where the majority of the y_i are unobserved, exacerbating the problem of a poorly modeled predictive distribution $p(y_i|\boldsymbol{x}_i, \Theta)$. This is ironic given that purely discriminative models (which may model the predictive distribution more reliably) do not apply to the semi-supervised case.

A natural prior to discourage this in the continuous case is the Gaussian process. Let us define k binary class indicator variables $b^{(1)}, b^{(2)}, \ldots b^{(k)}$ by $b_i^{(j)} := \mathbb{I}(y_i = j)$, where $\mathbb{I}$ is the zero/one indicator function. We take the simple approach of introducing a new factor in the (continuous analogy of the) joint likelihood, namely $\prod_{i=1}^k p_{b_i \sim \text{GP}(\Gamma)}(b_i)$. Following subsection 3.3 the graph analogy is

$$p_{\text{MRF}}(\boldsymbol{y}|L, \gamma) \propto \prod_{i=1}^{k} \exp(-\gamma \boldsymbol{b}^{(i)\top} L \boldsymbol{b}^{(i)}), \tag{4}$$

where the $\boldsymbol{b}^{(i)}$ are similarly defined binary class indicator vectors, and L is the graph Laplacian matrix.

3.4. Class Balancing Potential

Another particularly important issue in s.s.l. is class balancing as it is possible for most y_i to take on the same value in cases with very few labeled examples. This is especially dangerous after the inclusion of (4), which encourages such behavior. Hence we incorporate the further clique potential

$$p_{\text{BAL}}(\boldsymbol{y}|\boldsymbol{\eta}) \propto \prod_{j=1}^{k} \frac{\eta_j^{c_j}}{c_j!}, \tag{5}$$

which is the p.m.f. of Multinomial$(\eta_1, \eta_2, \ldots, \eta_k, m)$ evaluated at $c_1, c_2, \ldots, c_k$, where c_j is the class count: the number of distinct i for which $y_i = j$. Since the c_j are a (deterministic) function of the $\boldsymbol{y}$, this implies a density over $\boldsymbol{y}$. The y_i are no longer independent, but exchangeable. Although this distribution has presumably been studied, we are not aware of any such work. The prior over $\boldsymbol{y}$ it induces in our setting encodes a greater certainty about the class proportions c_j. It is easy to see that the implied distribution for $\boldsymbol{y}$ satisfies $p(y_i|\boldsymbol{y}_{\star \setminus i}) \propto \frac{\eta_{y_i}}{c'_{y_i}+1}$ where $\boldsymbol{y}_{\setminus i}$ is $\boldsymbol{y}$ without y_i and c'_{y_i} is the number of times y_i occurs in $\boldsymbol{y}_{\setminus i}$. Hence this distribution renders already seen events (realizations of the y_i) less likely to be seen in the future, in loosely speaking the opposite manner to the Dirichlet process, for example.

3.5. Final Model

The modifications (4) and (5) lead the final model we propose for s.s.l. which we now summarize for the sake of clarity. The variables are α, the Dirichlet parameter, L, the graph Laplacian and sufficient statistics for the graph, τ, the graph diffusion constant, $\boldsymbol{y}$, the vertex class labeling, $\boldsymbol{\xi}$, the observed (arbitrary, unique and categorical) vertex indices, $\boldsymbol{\beta}$, the vertex to mixture component assignments, γ, the parameter in (4), and $\boldsymbol{\eta}$, the class frequency proportions. Hence $p(\boldsymbol{y}, \boldsymbol{\beta}|\boldsymbol{\xi}, \boldsymbol{\eta}, \alpha, \gamma, L, D_\tau)$ is proportional to the following product of un-normalised clique potentials:

$$p_{\text{BAL}}(\boldsymbol{y}|\boldsymbol{\eta})\, p_{\text{MRF}}(\boldsymbol{y}|L, \gamma) \prod_j p_{\text{DM}}\Big(\boldsymbol{\beta}^{(j)}|\alpha\Big) \prod_i p(\xi_i|\beta_i, \tau),$$

where we recall $p(\xi_i|\beta_i,\tau) = (D_\tau)_{i,\beta_i}$ is the likelihood term for the graph diffusion mixture components (defined in subsection 3.2). $p_{\text{BAL}}(\boldsymbol{y}|\boldsymbol{\eta})$ and $p_{\text{MRF}}(\boldsymbol{y}|L,\gamma)$ are defined in (4) and (5) respectively. By $\boldsymbol{\beta}^{(j)}$ we mean the sub-vector formed by restricting $\boldsymbol{\beta}$ to those indices i for which $y_i = j$. Finally, p_{DM} is the p.m.f. of an m-dimensional Dirichlet-Multinomial implied by (3). Since the length of $\boldsymbol{\beta}^{(j)}$ is c_j, one can show that

$$p_{\text{DM}}\Big(\boldsymbol{\beta}^{(j)}|\alpha\Big) = \frac{\prod_{i=1}^{m}\Gamma(c_{j,i}+\frac{\alpha}{m})}{\Gamma(\frac{\alpha}{m})^m}\frac{\Gamma(\alpha)}{\Gamma(c_j+\alpha)}, \quad (6)$$

where $c_{j,i}$ is the number of distinct indices i' for which $y_{i'} = j$ and $\beta_{i'} = i$. Hence our definition of $\boldsymbol{\beta}^{(j)}$ is precise enough since as sufficient statistics we need only the frequencies of the values represented in each of the $\boldsymbol{\beta}^{(j)}$, as we can see above and in (7). The β_i will tend to take on a relatively small number of distinct values, just like the Gaussian means $\boldsymbol{m}_i$ of subsection 3.1.

4. INFERENCE ALGORITHM

To perform inference we sample the β_i and the unobserved y_i, treating all other variables as observed or fixed. The method we propose is a straightforward application of the Markov chain Monte Carlo (m.c.m.c.) idea, employing both Gibbs samples and Metropolis-Hastings moves. In particular, in each iteration of the sampler we Gibbs sample each y_i and β_i as described next, and then propose two different types of Metropolis-Hastings moves. Sampling each y_i and β_i individually leads to slow convergence so we block sample each (y_i,β_i) pair for the unlabeled data, sampling the β_i individually only for the labeled data for which y_i is observed. It is easy to check that the required conditional $p\left(y_i,\beta_i|*\setminus\{y_i,\beta_i\}\right)$ proportional to

$$\frac{\eta_y}{c'_y+1}\exp\left(-2\gamma\boldsymbol{b}^{(y)\top}L_{:,i}\right)\frac{c'_{y,\beta}+\alpha/m}{c'_y+\alpha}\,(D_\tau)_{i,\beta}, \quad (7)$$

where on the right hand side all of the y and β are shorthand for y_i and β_i. The c'_j above is as in subsection 3.4, while $c'_{j,l}$ is the number of distinct indices for which the entry in $\boldsymbol{y}_{\setminus i}$ (defined in 3.4) is j and the corresponding entry in the similarly defined $\boldsymbol{\beta}_{\setminus i}$ is l. The conditional in β_i for the labeled set is given by substituting the observed y_i in (7), computed efficiently by doing book-keeping on the term $L\boldsymbol{b}^{(i)}$.

In addition to Gibbs sampling each of the β_i and unobserved y_i, we employ two different m.c.m.c. moves, which we describe in the two following sub-sections. Recall that m is the number of points and k the number of classes, while c_y and $c_{y,\beta}$ are the class and mixture component counts (as defined after (5) and (6), respectively). We introduce the notation $\mathcal{Y} = \{1,2,\ldots,k\}$, $\mathcal{B} := \{1,2,\ldots,m\}$, and denote by $\mathcal{C}_{j,l} := \{i \in \mathcal{B} : y_i = j \wedge \beta_i = l\}$ what we refer to throughout as a *mixture component*.

4.1. Mixture Component Re-sampling

The idea is to select *from* and *to* vertices l and l', as well as a class j, and to propose re-assigning $\beta_i \leftarrow l'$ for all $i \in \mathcal{C}_{j,l}$. The particular procedure we employ is as follows.

1. Choose a vertex l and class label j randomly from $\mathcal{B}\times\mathcal{Y}$ with probability proportional to $c_{j,l}$.

2. Define $\mathcal{E}_j = \{l \in \mathcal{B} : c_{j,l} = 0\}$, the mixture components explaining no points from class j. Choose a new vertex l' randomly from $\mathcal{E}_j \cup l$ with probability proportional to $(D_\tau)_{l',l}$.

3. Choose a number uniformly at random from $[0,1]$. If it is less than the Kanji-like acceptance ratio

$$\mathcal{A}_{\text{move}}(l,l',j) = \frac{(D_\tau)_{l,l'}}{(D_\tau)_{l',l}}\prod_{i\in\mathcal{C}_{j,l}}\frac{(D_\tau)_{i,l'}}{(D_\tau)_{i,l}},$$

then accept by re-assigning $\beta_i \leftarrow l'$ for all $i \in \mathcal{C}_{j,l}$.

4.2. Mixture Component Re-Labeling

Here we select a vertex l and a class j and propose to relabel $y_i \leftarrow j'$ for all $i \in \mathcal{C}_{j,l}$. Since relabeling a single component with large assignment count $c_{j,l}$ may be too unlikely due to p_{BAL} of (5), we propose to relabel up to R components simultaneously:

1. Choose r uniformly at random from $\{1,2,\ldots,R\}$.

2. Define $\mathcal{R} = \{1,2,\ldots,r\}$ and choose r (mixture components, label) pairs $(j'_i,l'_i), i \in \mathcal{R}$ randomly without replacement from $\mathcal{B}\times\mathcal{Y}$ with probability $\propto c_{j'_i,l'_i}$.

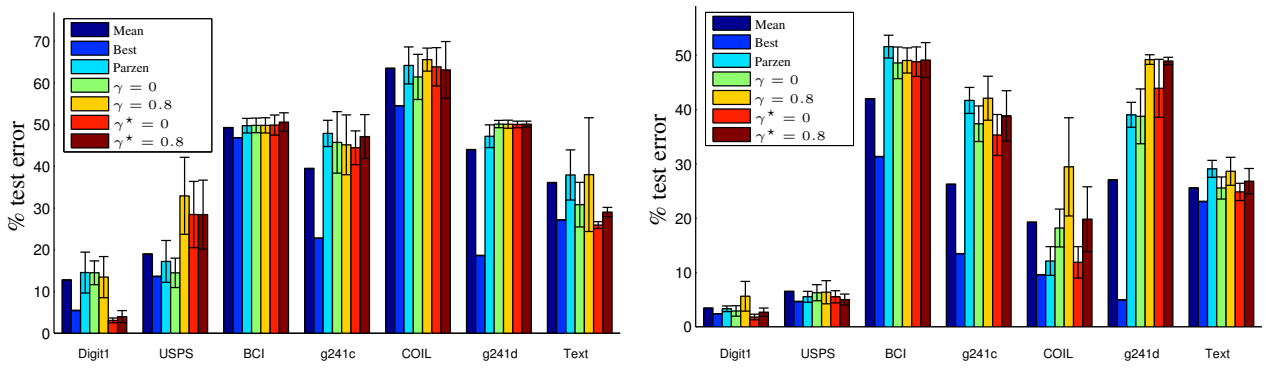
3. Repeat for all $i \in \mathcal{R}$ choosing y'_i at random from $\mathcal{Y}$ with probability proportional to $\eta_{y'_i}$.

4. Construct the proposal $\boldsymbol{y}^\star$ by initializing $\boldsymbol{y}^\star \leftarrow \boldsymbol{y}$ and relabeling the r mixture components. That is, repeating $\boldsymbol{y}^\star_w \leftarrow y'_i$ for all $i \in \mathcal{R}$ and for all $w \in \mathcal{C}_{j'_i,l'_i}$.

5. Choose a number uniformly at random from $[0,1]$. If it is less than the acceptance ratio

$$\mathcal{A}_{\text{relabel}}\left(\boldsymbol{y},\boldsymbol{y}^\star\right) = w\cdot\frac{p_{\text{BAL}}(\boldsymbol{y}^*|\boldsymbol{\eta})\,p_{\text{MRF}}(\boldsymbol{y}^*|L,\gamma)}{p_{\text{BAL}}(\boldsymbol{y}|\boldsymbol{\eta})\,p_{\text{MRF}}(\boldsymbol{y}|L,\gamma)},$$

where $w = \prod_{i\in\mathcal{R}}\left(\frac{\eta_{j_i}}{\eta_{y'_i}}\right)^{c_{j_i,l_i}}$ is the ratio of proposal densities, then accept by re-assigning $\boldsymbol{y} \leftarrow \boldsymbol{y}^\star$.



(a) 10 labeled points

(b) 100 labeled points

Fig. 1. Mean (colored bar height) and standard deviation (error bar width) percentage errors for (a) 10 and (b) 100 labeled points out of 1500. A star in the legend indicates the m.a.p. estimates rather than means. *mean* and *best* are the mean and best results of eleven other methods and *Parzen* is the graph Parzen window classifier.

5. EXPERIMENTS AND DISCUSSION

We provide results on the benchmark data, and investigate the role of p_{BAL} as well as the parameters γ and α.

Benchmark Test. We tested on the six two-class and one six-class benchmark problems of [2] as follows. Each bar in Figure 1 is a mean (standard deviation) over the twelve test splits supplied with the data sets, for each of the two supplied cases: 10 (top half of each table) and 100 (bottom half) labeled points, out of a total of 1500 points. The Dirichlet diffusion parameter α of (3) was set to 0.5, and the values 0 and 0.8 were tried for the parameter γ of (4). Note that for $\gamma = 0$ the Markov random field term plays no role. The class proportions η_j were set to known values, *i.e.* uniform for all but *USPS* which is imbalanced 4:1.

We used 10^5 iterations after 10^3 for burn in (one iteration being a Gibbs sweep followed by one attempt at each of the two Metropolis-Hastings moves), for a total computation time of around ten hours per split. Omitting the Metropolis-Hastings moves could lead to orders of magnitude degradations in convergence speed. We provide results for both the sample posterior mean of the unobserved labels y_i, as well as an *maximum a posteriori* (m.a.p.) estimate of them (denoted by a star in Figure 1). To compute the m.a.p. estimate, we kept track of the most likely state visited by the Markov chain, as a starting point for a final refinement by sequentially optimizing each of the conditional distributions used for Gibbs sampling, with the unobserved (β_i, y_i) pairs optimized jointly. We also include *best* and *mean* results reported by [2] for a pool of eleven different methods implemented and tested by their respective authors as part of that study. The sheer number of methods involved makes the *best* figures extremely competitive. No single model is expected to perform well on all data, so the relative aspects of our results are perhaps most informative. Also note that sets *g241c* and *g241n* are artificial and designed to test (and break) certain reasonable assumptions.

As a baseline we also include results for, by our discrete/continuous analogy, the graph analog of a Parzen window classifier. This models each class conditional density by a sum of diffusion p.m.f.'s (as represented by the columns of D_τ), one centered on each data point from that class. For a fair comparison we also re-weighted the densities according to the mixing proportions η_j. This is already competitive in terms of benchmark set performance, and is almost identical to the *discrete regularization* method described in [2] and equivalent to (1). Indeed, our Parzen results are all within one standard deviation of the *Discrete Regularization* results given by [2].

To construct the graph from the vectorial data, we followed the procedure outlined in section 2, with ω set to the mean squared distance over all pairs of points. Model and hyper-parameter selection can be problematic in s.s.l. as there may be too few labeled points for simple validation set type procedures to be reliable. Hence, for the 10 labels case we simply fixed the number k of nearest neighbors connected in the graph to be 10, and fixed the diffusion coefficient τ to 5. For the 100 labels case we used a leave one out estimate of the test error of the Parzen window method to choose the k and τ by grid search.

An important comparison in Figure 1 is between the $\gamma = 0$ version of our method and the Parzen method, since this most clearly isolates the strength of the latent Dirichlet-Multinomial diffusion mixtures in comparison to a straightforward label diffusion with the same graph and diffusion parameters. Our new method is equal or better on all but the *COIL* 100 labels case. The $\gamma = 0.8$ results are overall better still, and we obtain significantly best overall performance

on the *Digit1* and *Text* data sets, compared with all methods in the original study. Moreover our method is within a standard deviation of the best on many problems, and fails to obtain average results (those within a standard deviation of *mean*) only on the artificial data sets *g241c* and *g241d* and the highly noisy *BCI* dataset. We also see that the m.a.p. estimate is worse than or the same as the mean on all but the *COIL* problem (for the case of 100 labels only). As expected, even when the mean errors are close, the m.a.p. estimate tends to have much higher variance than the mean.

Parameter Exploration. To better understand the parameters α and γ we computed mean test errors (on the unlabeled points) for a grid of values of these parameters. We repeated twenty times taking 100 points from the *Digit1* data set, and randomly labeled five points per class. To verify the importance of the class balancing prior p_{BAL} of subsection 3.4, we repeated the experiment with the term replaced by the original one of the model of subsection 3.2, namely $p(\boldsymbol{y}) = \prod_{j=1}^{m} \eta_{y_j}$. The result in Figure 2 shows a severe degradation in the performance of the unbalanced model for large values of γ. We verified from the samples that this is due to the tendency of p_{MRF} to cause many of the unobserved y_i to take on the same value. There is also a tendency for smaller values of α (*i.e.* more parsimonious graph mixture models) to give rise to better performance, demonstrating the effectiveness of the generative model of subsection 3.2 in combination with the Markov random field term p_{BAL}. Finally, note that while in this example a large value of γ is optimal, this was not always the case on the benchmark sets as in Figure 1. In particular, we see that for the *USPS* dataset with 10 labels, the $\gamma = 0$ setting results in a large improvement in predictive power.

6. CONCLUSIONS

We proposed an algorithm for s.s.l. which utilizes established graph based regularization tools in a different manner to previous approaches. We define a probabilistic model for the labeling of the graph and perform inference using sampling, thereby avoiding the non-convex optimization characteristic of s.s.l. problems. In this way, at the cost of more computation we are able to obtain significantly better predictions compared to a rather large pool of previous methods on a standardised benchmark set. Our approach is based on a new model component, the non-parametric mixture of graph diffusions with centers distributed according to a Dirichlet-Multinomial distribution over the vertices of the graph. This component is applicable to many other problems involved with modeling random quantities on the vertices of a graph, as in world wide web hyper-link data, and community detection problems.

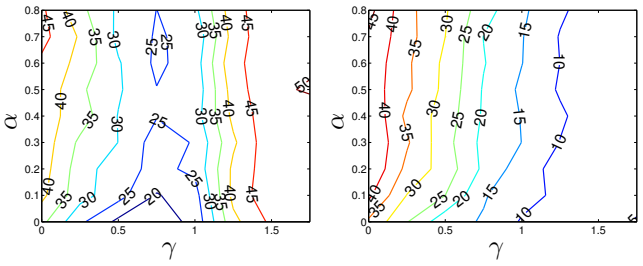


Fig. 2. Mean % error for a range of α (vertical axis) and γ (horizontal axis) both with (right) and without (left) the class balancing prior p_{BAL} of subsection 3.4.

7. REFERENCES

[1] Xiaojin Zhu, "Semi-supervised learning literature survey," Tech. Rep. 1530, Computer Sciences, University of Wisconsin-Madison, 2007.

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

[3] Matthias Seeger, "Learning with labeled and unlabeled data," Tech. Rep., Univ. Edinburgh, Dec. 2002.

[4] Vladimir Vapnik, *Statistical Learning Theory*, John Wiley and Sons, inc., New York, 1998.

[5] Olivier Chapelle, Vikas Sindhwani, and Sathiya S. Keerthi, "Optimization techniques for semi-supervised support vector machines," *JMLR*, vol. 9, pp. 203–233, 2008.

[6] Gad Getz, Noam Shental, and Eytal Domany, "Learning with partially classified training data," in *Learning with Partially Classified Training Data ICML 2005 Workshop*, Bonn, Germany, 2005.

[7] D. Zhou and B. Schölkopf, *Discrete Regularization*, pp. 221–232, Adaptive computation and mach. learning. MIT Press, Cambridge, Mass., USA, 11 2006.

[8] M Hein, J-Y Audibert, and U von Luxburg, "Graph laplacians and their convergence on random neighborhood graphs," *JMLR*, vol. 8, May 2007.

[9] Risi Imre Kondor and John D. Lafferty, "Diffusion kernels on graphs and other discrete input spaces," in *ICML*, Claude Sammut and Achim G. Hoffmann, Eds. 2002, pp. 315–322, Morgan Kaufmann.

[10] Tom Griffiths and Zoubin Ghahramani, "Infinite latent feature models and the indian buffet process," in *Advances in Neural Information Proc. Systems 18*, 2005.