Dissimilarity in Graph-Based Semi-Supervised Classification

Andrew B. Goldberg
Department of Computer Sciences
University of Wisconsin, Madison
Madison, WI 53705
goldberg@cs.wisc.edu

Xiaojin Zhu
Department of Computer Sciences
University of Wisconsin, Madison
Madison, WI 53705
jerryzhu@cs.wisc.edu

Stephen Wright
Department of Computer Sciences
University of Wisconsin, Madison
Madison, WI 53705
swright@cs.wisc.edu

Abstract

Label dissimilarity specifies that a pair of examples probably have different class labels. We present a semi-supervised classification algorithm that learns from dissimilarity and similarity information on labeled and unlabeled data. Our approach uses a novel graph-based encoding of dissimilarity that results in a convex problem, and can handle both binary and multiclass classification. Experiments on several tasks are promising.

1 INTRODUCTION

Semi-supervised classification learns a classifier from both labeled and unlabeled data by encoding domain knowledge on unlabeled data in the model [3, 11, 19]. In this paper we focus on a particular form of domain knowledge: the *label dissimilarity* between examples. We assume we are given a set of dissimilarity pairs $\mathcal{D} = \{(i,j)\}$. For $(i,j) \in \mathcal{D}$, the two points $\mathbf{x}_i, \mathbf{x}_j$ may be both unlabeled, or one labeled and the other unlabeled. In either case we know they probably do not have the same label. The dissimilarity knowledge can be noisy.

As an example, consider the problem of predicting a person's political view (left, right) from his/her postings to online blogs. The fact that person B quotes person A and uses expletives near the quote is a strong indication that B disagrees with A [9]. Simple text processing thus allows us to create a dissimilarity pair (A,B) to reflect our knowledge that A and B probably have different labels (political views).

Such dissimilarity knowledge has been extensively studied in *semi-supervised clustering*, where such pairs are known as "cannot-links" [1, 6, 13, 14, 18], meaning they cannot be in the same cluster. These methods either directly modify the clustering algorithm, or change the underlying distance metric. Our method is different in that it specifically applies to classification, and works on discriminant functions. Dissimilarity as negative correlation on discriminant functions has been discussed in relational learning with Gaussian processes [4], but their formulation is non-convex and applies only to binary classification. In contrast our formulation is convex and applicable to multiple classes.

Our contribution is a convex method that incorporates both similarity and dissimilarity in semi-supervised learning. We start with graph-based semi-supervised classification methods (e.g., [2, 20]), which allows a natural combination of similarity and dissimilarity. Existing graph-based semi-supervised learning methods encode *label similarity* knowledge, but they cannot handle dissimilarity easily, as we show in Section 2. We define a mixed graph to accommodate both, and define the analog of graph Laplacian. We then adapt manifold regularization [12] to the mixed graph. We extend our method to multiclass classification in Section 3, and present experimental results in Section 4.

2 DISSIMILARITY IN BINARY CLASSIFICATION

Let there be n items, of which l are labeled: $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_l, y_l), \mathbf{x}_{l+1}, \cdots, \mathbf{x}_n\}$. Existing graph-based semi-supervised classification methods assume that a graph over the n items is given. The graph is represented by an $n \times n$ matrix W, where w_{ij} is the non-negative edge weight between items i, j. Similar items have large weights, reflecting the domain knowledge (or assumption) that they tend to have similar labels. Such knowledge can be represented as a penalty term [20] on the discriminant function $f : X \mapsto \mathbb{R}$:

$$\frac{1}{2} \sum_{i,j=1}^{n} w_{ij} (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2. \qquad (1)$$

Minimization of (1) tends to force $f(\mathbf{x}_i) \approx f(\mathbf{x}_j)$ when w_{ij} is large. Therefore existing graph-based methods are able to encode *label similarity* domain knowledge. The penalty (1) can be written in quadratic form $\mathbf{f}^\top \mathcal{L}\mathbf{f}$, where $\mathbf{f} = (f(\mathbf{x}_1), \cdots, f(\mathbf{x}_n))^\top$ and $\mathcal{L}$ is known as the combinatorial graph Laplacian matrix, defined as $\mathcal{L} = D - W$ where D is the diagonal degree matrix with $d_{ii} = \sum_{j=1}^n w_{ij}$.

Existing graph-based methods cannot easily handle dissimilarity, which is the requirement that two items have different labels. A small or zero weight w_{ij} does *not* represent dissimilarity between $\mathbf{x}_i$ and $\mathbf{x}_j$; in fact, a zero edge weight means no preference at all. A negative weight $w_{ij} < 0$ does encourage a large difference between $f(\mathbf{x}_i), f(\mathbf{x}_j)$, but this creates a number of problems. First f needs to be bounded or $\{-\infty, \infty\}$ will be a trivial minimizer. Second, any negative weight in W will make (1), and ultimately the whole semi-supervised problem, non-convex. One has to resort to approximations [10, 15, 16]. It is highly desirable to keep the optimization problem convex.

2.1 MIXED GRAPHS

Let us assume $y \in \{-1, 1\}$ for binary classification. Our key idea is to encode dissimilarity between i, j as $w_{ij}(f(\mathbf{x}_i) + f(\mathbf{x}_j))^2$. Note the summation. This term is zero if $f(\mathbf{x}_i), f(\mathbf{x}_j)$ have the same absolute value but opposite signs, thus encouraging different labels. The trivial case $f(\mathbf{x}_i) = f(\mathbf{x}_j) = 0$ is avoided by competing terms in a risk minimization framework (Section 2.2). The weight w_{ij} remains positive, and represents the strength of our belief in this dissimilarity edge.

Definition 1 *A mixed graph over n nodes has similarity and dissimilarity edges, and is represented by two $n \times n$ matrices S and W. S specifies the edge type: $s_{ij} = 1$ if there is a similarity edge between i, j; $s_{ij} = -1$ if there is a dissimilarity edge. Non-negative weights $w_{ij} \geq 0$ represent the strength of the edge, regardless of its type.*

The graphs in existing graph-based semi-supervised learning methods can be viewed as having an all-one S and the same W. Extending (1) to the mixed graph, we would like to minimize a new penalty term

$$\frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f(\mathbf{x}_i) - s_{ij} f(\mathbf{x}_j))^2. \qquad (2)$$

It handles both similarity and dissimilarity, and is clearly convex in f. Furthermore, we can re-write (2) in a quadratic form.

Proposition 1 *Let $\mathcal{M} = \mathcal{L} + (\mathbf{1} - S) \bullet W$, where $\mathcal{L}$ is the combinatorial graph Laplacian, $\mathbf{1}$ is the all-one matrix, and $\bullet$ is the Hadamard (elementwise) product. Then $\mathcal{M}$ is positive semi-definite, and $\mathbf{f}^\top \mathcal{M}\mathbf{f} = \frac{1}{2}\sum_{i,j} w_{ij}(f(\mathbf{x}_i) - s_{ij} f(\mathbf{x}_j))^2$.*

The matrix $\mathcal{M}$ is the mixed-graph analog of the graph Laplacian $\mathcal{L}$. Like the Laplacian, $\mathcal{M}$ is positive semi-definite, as can be seen from (2). If the graph has no dissimilarity edges, then $\mathcal{M} = \mathcal{L}$.

2.2 MANIFOLD REGULARIZATION WITH DISSIMILARITY

Manifold regularization [2] generalizes graph-based semi-supervised learning with a regularized risk minimization framework. Let $\mathcal{H}$ be the Reproducing Kernel Hilbert Space (RKHS) of a kernel K. Manifold regularization obtains the discriminant function by solving

$$\min_{f \in \mathcal{H}} \sum_{i=1}^{l} c(y_i, f(\mathbf{x}_i)) + \lambda_1 \|f\|_{\mathcal{H}}^2 + \lambda_2 \mathbf{f}^\top \mathcal{L}\mathbf{f}, \qquad (3)$$

where $c()$ is an arbitrary loss function, e.g., the hinge loss for Support Vector Machines (SVMs), or squared loss for Regularized Least Squares (RLS) classifiers. As before, $\mathbf{f}$ is the vector of discriminant function values on the n points. The first two terms in (3) are the same as in supervised learning, while the third term is the additional regularization term for graph-based semi-supervised learning. Because f is defined in $\mathcal{H}$ now, it naturally extends to new test points. Noisy labels are tolerated by the loss function.

The mixed-graph analog of (3) is

$$\min_{f \in \mathcal{H}} \sum_{i=1}^{l} c(y_i, f(\mathbf{x}_i)) + \lambda_1 \|f\|_{\mathcal{H}}^2 + \lambda_2 \mathbf{f}^\top \mathcal{M}\mathbf{f}. \qquad (4)$$

One can solve the optimization problem (4) directly. Alternatively one can view the second and third terms together as regularization by a warped kernel, as proposed in [12]. In this view, one defines a second RKHS $\mathcal{H}'$, which has the same functions as $\mathcal{H}$ but a different inner product: $\langle f, g\rangle_{\mathcal{H}'} = \langle f, g\rangle_{\mathcal{H}} + f^\top M g$, where M is some positive semi-definite matrix on the n points. It follows that $\|f\|_{\mathcal{H}'}^2 = \|f\|_{\mathcal{H}}^2 + \mathbf{f}^\top M\mathbf{f}$. The *supervised* problem $\min_{f \in \mathcal{H}'} \sum_{i=1}^{l} c(y_i, f(\mathbf{x}_i)) + \lambda_1 \|f\|_{\mathcal{H}'}^2$ is then equivalent to our semi-supervised learning problem (4), if we let $M = \frac{\lambda_2}{\lambda_1}\mathcal{M}$. Importantly, it is shown in [12] that the kernel K' for the warped RKHS $\mathcal{H}'$ is related to the original K as follows:

$$k'(x, z) = k(x, z) - \mathbf{k}_x{}^\top (I + MK)^{-1} M \mathbf{k}_z, \qquad (5)$$

where $\mathbf{k}_x = (k(\mathbf{x}_1, \mathbf{x}), \cdots, k(\mathbf{x}_n, \mathbf{x}))^\top$. This allows one to compute the warped kernel K' from some original kernel (e.g., RBF) K and the mixed-graph $\mathcal{M}$. Therefore, to solve (4), we can use K' in conjunction with standard supervised kernel machine software.

3 DISSIMILARITY IN MULTICLASS CLASSIFICATION

It is non-trivial to incorporate dissimilarity into multiclass classification.
1. One-vs-rest does not work with dissimilarity and semi-supervised learning. Suppose, for example, that there are three classes, and that $\mathbf{x}_i, \mathbf{x}_j$ are two unlabeled points whose actual labels are 2 and 3, respectively. Let (i,j) be specified as a dissimilarity edge. In the binary sub-task of class 1 vs. all other classes, however, this dissimilarity edge should become a similarity edge, since $\mathbf{x}_i, \mathbf{x}_j$ are both in the 'rest' meta-class.
2. One-vs-one does not work either. For any particular one-vs-one sub-task (say class 1 vs. 2), it is not clear whether any unlabeled point (say $\mathbf{x}_j$ which actually has class 3) should participate in the one-vs-one semi-supervised learning. If an unlabeled point does not have one of the two labels, its inclusion will likely confuse learning.
3. Using the warped kernel (5) in a standard multiclass kernel machine (e.g., multiclass SVM) does not work. Multiclass methods use k discriminant functions $f_1, \cdots, f_k$, one for each class. The warped kernel incorrectly encourages all discriminant functions to honor $f_\cdot(\mathbf{x}_i) + f_\cdot(\mathbf{x}_j) = 0$, which is unnecessary and potentially harmful.
We found all the above approaches indeed hurt accuracy. These experiments are not reported here.

We therefore need to redesign the multiclass objective in order to incorporate dissimilarity. For simplicity we focus on multiclass SVMs, but our method works for other loss functions too. There are several formulations of multiclass SVMs, e.g., [5, 7, 17]. For our purpose it is important to anchor the discriminant functions around zero. For this reason we start with the formulation in [7]. A k-class SVM is defined as the optimization problem of finding functions $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_k(\mathbf{x}))$ that solve:

$$\begin{aligned} \min \quad & \tfrac{1}{l}\textstyle\sum_{i=1}^{l} L_{i\cdot}(\mathbf{f}(\mathbf{x}_i) - \mathbf{y}_i)_+ + \lambda \sum_{j=1}^{k} \|h_j\|^2_{\mathcal{H}} \\ \text{s.t.} \quad & \textstyle\sum_{j=1}^{k} f_j(\mathbf{x}_i) = 0, \quad i = 1 \cdots l, \end{aligned} \tag{6}$$

where $f_j(\mathbf{x}) = h_j(\mathbf{x}) + b_j$ for $j = 1 \cdots k$; $h_j \in \mathcal{H}$, which is the RKHS of some kernel K; and $b_j \in \mathbb{R}$. There are l labeled training points. L is an $l \times k$ matrix, with the i-th row $L_{i\cdot} = (1, \cdots, 1, 0, 1, \cdots, 1)$ being an all-one vector except the y_i-th element which is zero. y_i is the given label for $\mathbf{x}_i$. The vector $\mathbf{y}_i = (-1/(k-1), \cdots, 1, -1/(k-1), \cdots)^\top$ is an encoding of the label y_i, where the number 1 occurs in the y_i-th position. The plus function is $(z)_+ = \max(0, z)$. Intuitively, (6) means that $\mathbf{f}(\mathbf{x}_i)$ should have elements less than $-1/(k-1)$ for all 'wrong classes'. It is important to note that the elements of $\mathbf{y}_i$ and $\mathbf{f}(\mathbf{x}_i)$ sum to zero.

We exploit this sum-to-zero label encoding to represent dissimilarity as a convex multiclass SVM objective. To simplify the notation, we will restrict ourselves to dissimilarity edges with weight 1. Similarity edges can be added to the formulation easily by using terms like $(\mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_j))^2$ as in [12, 20]. Given a dissimilarity edge $(s,t) \in \mathcal{D}$, the key idea behind our multiclass dissimilarity formula comes from comparing $\mathbf{f}(\mathbf{x}_s), \mathbf{f}(\mathbf{x}_t)$ for the 'good' and 'bad' cases. The 'good' case is when $\mathbf{f}$ takes the nominal encoding $\mathbf{f}(\mathbf{x}_s) = \mathbf{y}_s$ and $\mathbf{f}(\mathbf{x}_t) = \mathbf{y}_t$, and $\mathbf{y}_s \neq \mathbf{y}_t$. By definition $\mathbf{y}_s$ and $\mathbf{y}_t$ have the form $(-1/(k-1), \cdots, 1, -1/(k-1), \cdots)^\top$, where the elements with value 1 must be at different positions. Hence $\mathbf{y}_s + \mathbf{y}_t$ is a vector with two kinds of elements: $(k-2)/(k-1)$ and $-2/(k-1)$. The 'bad' case is when $\mathbf{y}_s = \mathbf{y}_t$, so the elements with value 1 coincide. In this case the sum $\mathbf{y}_s + \mathbf{y}_t$ has two kinds of elements: 2 and $-2/(k-1)$. Comparing 'good' and 'bad', we do not want any element in $\mathbf{f}(\mathbf{x}_s) + \mathbf{f}(\mathbf{x}_t)$ to be larger than $(k-2)/(k-1)$. We are therefore led to the following dissimilarity objective:

$$\sum_{(s,t)\in\mathcal{D}} \sum_{j=1}^{k} \left(f_j(\mathbf{x}_s) + f_j(\mathbf{x}_t) - \frac{k-2}{k-1} \right)_+^p, \tag{7}$$

which is a sum of plus functions raised to the p-th power. The advantages of this definition are that it is convex and simple, and it reduces to our binary SVM dissimilarity formulation when $p = 2, k = 2$.

In standard practice, one can combine (6) and (7) as follows:

$$\begin{aligned} \min \quad & \tfrac{1}{l}\textstyle\sum_{i=1}^{l} L_{i\cdot}(\mathbf{f}(\mathbf{x}_i) - \mathbf{y}_i)_+ + \lambda_1 \sum_{j=1}^{k} \|h_j\|^2_{\mathcal{H}} \\ & + \tfrac{\lambda_2}{|\mathcal{D}|} \textstyle\sum_{(s,t)\in\mathcal{D}} \sum_{j=1}^{k} \left(f_j(\mathbf{x}_s) + f_j(\mathbf{x}_t) - \tfrac{k-2}{k-1} \right)_+^p \\ \text{s.t.} \quad & \textstyle\sum_{j=1}^{k} f_j(\mathbf{x}_i) = 0, \quad i = 1 \cdots n, \end{aligned} \tag{8}$$

where n is the sum of the number of unlabeled points that are involved in any dissimilarity edge, plus the number of labeled points l. The Representer Theorem in [7] needs to be extended to include these unlabeled points [21]. In particular, the minimizing functions for (8) have the form

$$f_j(\mathbf{x}) = \sum_{i=1}^{n} c_{ij} K(\mathbf{x}_i, \mathbf{x}) + b_j \quad \text{for } j = 1, \cdots, k \tag{9}$$

The essential difference to supervised learning is that we now have n rather than l representers in (9).

Using (9), we formulate (8) as a quadratic program. Note $\|h_j\|^2_{\mathcal{H}} = c_{\cdot j}^\top K c_{\cdot j}$, where $K_{st} = K(\mathbf{x}_s, \mathbf{x}_t)$ is the $n \times n$ Gram matrix. We let $p = 1$ in the dissimilarity objective (7). This leads to the primal form

$$\begin{aligned} \min \quad & \tfrac{1}{l} \textstyle\sum_{i=1}^{l} L_{i\cdot}(\mathbf{f}(\mathbf{x}_i) - \mathbf{y}_i)_+ + \lambda_1 \sum_{j=1}^{k} c_{\cdot j}^\top K c_{\cdot j} \\ & + \tfrac{\lambda_2}{|\mathcal{D}|} \textstyle\sum_{(s,t)\in\mathcal{D}} \sum_{j=1}^{k} \left(f_j(\mathbf{x}_s) + f_j(\mathbf{x}_t) - \frac{k-2}{k-1} \right)_+ \\ \text{s.t.} \quad & \textstyle\sum_{j=1}^{k} f_j(\mathbf{x}_i) = 0, \;\; i = 1 \cdots n. \end{aligned} \tag{10}$$

We define an $l \times k$ matrix Y whose i-th row is $\mathbf{y}_i^\top$. Substituting (9) into (10), we obtain

$$\begin{aligned} \min \quad & \tfrac{1}{l} \textstyle\sum_{i=1\cdots l}^{j=1\cdots k} L_{ij}(K_{i\cdot}c_{\cdot j} + b_j - Y_{ij})_+ \\ & + \lambda_1 \textstyle\sum_{j=1}^{k} c_{\cdot j}^\top K c_{\cdot j} \\ & + \tfrac{\lambda_2}{|\mathcal{D}|} \textstyle\sum_{(s,t)\in\mathcal{D}}^{j=1\cdots k} \left((K_{s\cdot} + K_{t\cdot})c_{\cdot j} + 2b_j - \frac{k-2}{k-1} \right)_+ \\ \text{s.t.} \quad & \textstyle\sum_{j=1\cdots k}(K_{i\cdot}c_{\cdot j} + b_j) = 0, \;\; i = 1 \cdots n. \end{aligned} \tag{11}$$

Finally we introduce an $l \times k$ matrix ξ and a $|\mathcal{D}| \times k$ matrix τ as auxiliary variables. With standard reformulation techniques, we rewrite (11) as

$$\begin{aligned} \min \quad & \tfrac{1}{l} \textstyle\sum_{i=1\cdots l}^{j=1\cdots k} L_{ij}\xi_{ij} + \lambda_1 \sum_{j=1}^{k} c_{\cdot j}^\top K c_{\cdot j} \\ & + \tfrac{\lambda_2}{|\mathcal{D}|} \textstyle\sum_{(s,t)\in\mathcal{D}}^{j=1\cdots k} \tau_{stj} \\ \text{s.t.} \quad & K_{i\cdot}c_{\cdot j} + b_j - Y_{ij} \le \xi_{ij}, \;\; i = 1\cdots l, j = 1\cdots k \\ & \xi_{ij} \ge 0, \;\; i = 1\cdots l, j = 1\cdots k \\ & (K_{s\cdot} + K_{t\cdot})c_{\cdot j} + 2b_j - \tfrac{k-2}{k-1} \le \tau_{stj}, \\ & \tau_{stj} \ge 0, \;\; (s,t) \in \mathcal{D}, j = 1\cdots k \\ & \textstyle\sum_{j=1\cdots k}(K_{i\cdot}c_{\cdot j} + b_j) = 0, \;\; i = 1\cdots n, \end{aligned} \tag{12}$$

where the minimization is over c, b, ξ, τ. The quadratic program has $O(nk)$ variables and constraints.

4 EXPERIMENTS

In the following sections, we empirically demonstrate the benefits of incorporating dissimilarity in several classification tasks.

4.1 STANDARD BINARY DATASETS

We first experimented using the standard binary datasets *g50c* and *mac-windows* used in [12] and available with the authors' code at `http://people.cs.uchicago.edu/~vikass/research.html`. As in [12], g50c contains 550 examples containing 50 dimensions, and we use $l = 50$ labeled samples. Mac-windows has 1946 examples with 7511 dimensions, also with $l = 50$.

Ideally, we would like to use dissimilarity information based on domain knowledge. However, without such expertise available to us, we performed "oracle experiments" in which we introduce dissimilarity edges between randomly sampled data points with different labels. Because the edges represent ground-truth dissimilarity, we disallow edges to touch labeled points, to prevent the true labels propagating throughout the unlabeled data. Note that the actual label values are not revealed—just the fact that the points should receive *different* label classifications. Simulating domain knowledge in this manner is common for cannot-link clustering and related work. In Section 4.3, we present results involving 'real' dissimilarity based on domain-specific heuristics.

In this subsection, we introduce dissimilarity in the manifold regularization framework, discussed in Section 2.2. Following [12], we start with a Gaussian base kernel K and encode similarity using k-nearest-neighbor graphs with Gaussian weights. Specifically, the weight between kNN points x_i and x_j is $e^{-\frac{||x_i - x_j||^2}{2\sigma^2}}$, while all other weights are zero. We then add the above dissimilarity edges, and assign them a relatively large weight (see below) to form the mixed-graph matrix $\mathcal{M}$. Our experiments used the resulting warped kernel K' in both SVM and RLS classifiers. The methods were implemented using LIBSVM and a modified version of the code from [12]. We used the same parameter values as [12]. These had been tuned in that paper with 5-fold cross validation using similarity only; our dissimilarity results could become even better with additional parameter tuning.

To compare error rate on unlabeled data used during semi-supervised training, and on new unseen test data, we divided each dataset into four disjoint folds. We then performed 4-fold cross validation, using each fold as a test set once. The test set remains unseen throughout the learning process. The remaining three folds comprised the training set (labeled and unlabeled data). For each train/test split, we trained 10 different classifiers, each time using a different random choice of labeled examples and dissimilarity edges between unlabeled examples. The same random choices are made in all experimental runs, so we can compare results using paired statistical tests. We report classification error rate on the unlabeled training set (*in-sample* performance) and unseen test data (*out-of-sample* performance). Each number is averaged over 4 folds with 10 random trials each. We address two questions in these standard binary dataset experiments:

How does the number of dissimilarity edges influence mean error rate? We experimented first with varying the number of dissimilarity edges in the graph. Since we have high confidence in the oracle edges, we assign each edge a weight equal to the maximal similarity edge weight (close to 1 for our datasets).

Figure 1 shows the effect of changing the number of dissimilarity edges in the g50c and mac-windows datasets. Figures 1(a,b,e,f) present mean in-sample and out-of-sample error rates using 50–12800 dissimilarity edges, as compared to the baseline with 0 dissimilarity edges, using a hinge loss function for $c()$ in (4). They are similar to LapSVMs, but with dissimilarity edges. Figures 1(c,d,g,h) display comparable results using a squared error loss function for $c()$ in (4). These are similar to LapRLS, but with dissimilarity edges. In all plots, we show one standard deviation above and below the error rate curve. The baselines here use only similarity edges in graph-based semi-supervised learning. They are equivalent to LapSVM and LapRLS in [12].

Figure 1 shows the positive impact of dissimilarity edges. The effect is greater for in-sample performance; the in-sample points were directly involved in the kernel deformation, so this benefit is to be expected. Our model also generalizes to out-of-sample test data. To measure statistical significance, we performed two-tailed, paired t-tests, comparing the results using each number of dissimilarity edges to the baseline in each of the subplots. The circled settings are statistically significant at the 0.05 level.

While out-of-sample performance steadily improves in the mac-windows dataset (Figures 1(f,h)), the g50c out-of-sample error benefits less with 6400 or 12800 dissimilarity edges (Figures 1(b,d)). The increase in error rate corresponds with near-zero in-sample error rates, suggesting that the learning algorithm is overfitting the dissimilarity edges. For this small dataset, nearly all of the unlabeled points are touched by one or more of the 6400–12800 dissimilarity edges. (Mac-windows is roughly four times as large, so this is not the case.) It seems the kernel becomes so warped that it fits the g50c unlabeled points perfectly, but becomes less effective in classifying unseen test points. Though we require only $f(\mathbf{x}_i)f(\mathbf{x}_j) < 0$ for $\mathbf{x}_i$ and $\mathbf{x}_j$ to be labeled differently, the dissimilarity terms encourage $f(\mathbf{x}_i) = -f(\mathbf{x}_j)$ for $(i, j) \in \mathcal{D}$. We believe that this unnecessarily stringent requirement is at the root of the observed overfitting when too many dissimilarity terms are included. While the mechanics are still unclear, the inappropriate demand appears to become overwhelming, and generalization error starts to increase.

What is the effect of the weight assigned to dissimilarity edges? In the preceding experiments, we varied the number of dissimilarity edges, but fixed their weights to roughly 1. We next fixed the number of edges at 200, and experimented with varying this weight by a range of multiplicative factors (Figure 2). This effectively places more or less confidence in the dissimilarity edges, compared to the similarity edges. As before, the baseline is either LapSVM or LapRLS, and does not use any dissimilarity.

We observe that in-sample performance tends to benefit from stronger weights on dissimilarity edges (Figures 2(a,c,e,g)). The maximal decrease in mean error rate appears at a weight of approximately 64, above which the error rate rises slightly. In both datasets, above a weight of approximately 100, the out-of-sample error rate (Figures 2(b,d,f,h)) dramatically rises above the baseline. This appears to be another case of overfitting—the kernel deformation relies too heavily on the dissimilarity edges, and much useful similarity information is being ignored. This results in good in-sample performance, at the expense of correct classification of new examples.

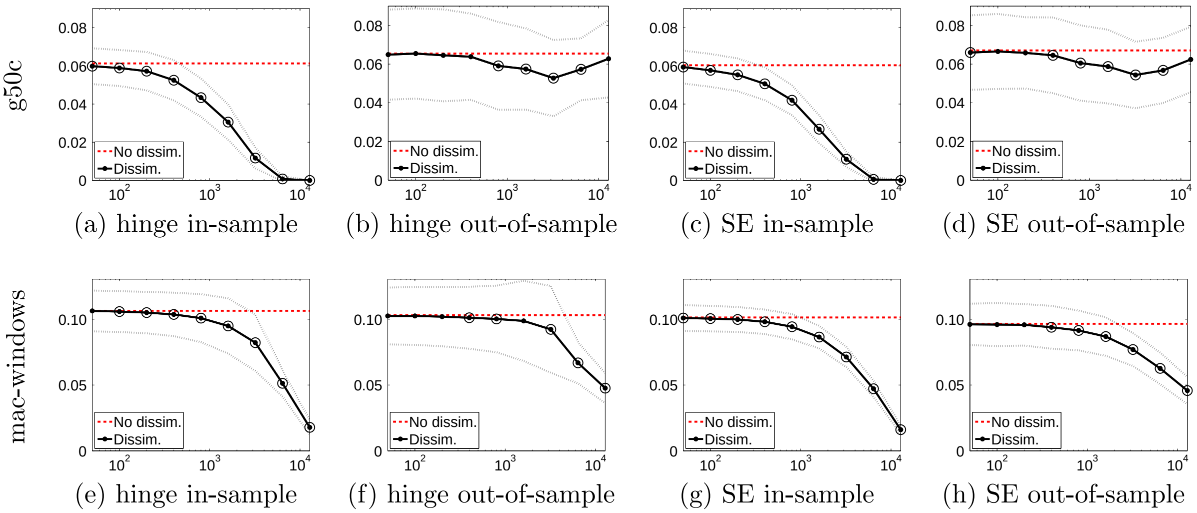
Table 1: Mean error rate with varying numbers of dissimilarity edges in the USPS dataset using the multiclass SVM formulation.

Dissim.	Overall	In-sample	Out-of-sample
baseline 0	24.48	24.48	24.48
10	24.41	20.47	24.40
20	24.32	23.53	24.33
40	24.27	24.17	24.27
80	**23.96**	23.57	**23.99**
160	**23.63**	24.49	**23.48**
320	**23.30**	23.57	**23.20**

4.2 STANDARD MULTICLASS DATASET

We next experimented with dissimilarity in multiclass classification as described in Section 3. We used the standard multiclass dataset *USPS test*, which contains 2007 examples with 256 dimensions, each belonging to one of 10 classes. We used labeled set size $l = 50$. This dataset was also used in [12] and is available at the URL cited above. We solve the quadratic program in (12) using the CPLEX QP solver. We experimented using varying numbers of oracle dissimilarity edges. As before, our dissimilarity edges do not touch labeled points. We consider those examples involved in dissimilarity to be the unlabeled set, and the remaining examples (ignored during training) the unseen test set. We report mean error rates over 10 repeated trials using different random labeled sets and different random unlabeled-unlabeled dissimilarity edges. The λ_1 parameter in (12) was optimized using mean test set performance without any dissimilarity. Thus, we are making the baseline as strong as possible. We arbitrarily set $\lambda_2 = 1$. Careful tuning of this parameter could potentially lead to even better results.

Table 1 presents the overall, in-sample, and out-of-sample mean error rates using the 2-norm SVM formulation (12) with a varying number of dissimilarity edges. Statistically significant reductions in error rate,



(a) hinge in-sample (b) hinge out-of-sample (c) SE in-sample (d) SE out-of-sample

(e) hinge in-sample (f) hinge out-of-sample (g) SE in-sample (h) SE out-of-sample

Figure 1: Varying the number of dissimilarity edges (x-axis) in the g50c dataset (a-d) and the mac-windows dataset (e-h). y-axis is mean error rate across 4 folds with 10 random trials each. 'Hinge' stands for the hinge loss and 'SE' the squared error loss. The baselines are LapSVM and LapRLS respectively, which have no dissimilarity edges. Circled settings are statistically significantly better than the baseline.

compared to the baseline, are indicated in bold face. The 2-norm multiclass SVM formulation uses the dissimilarity edges effectively to lower overall and out-of-sample mean error rate for all amounts of dissimilarity edges that we tested. Note that the baseline has a higher error rate than reported in [12], and this is because we used the multiclass SVM formulation of [7] to allow dissimilarities, not the code in [12].

4.3 POLITICS DATASET

In our final set of experiments, we create real (instead of oracle) dissimilarity edges based on domain knowledge. We experimented with the `politics.com` discussion board text data from [9]. The task here is to predict the political affiliation of the users posting messages on a political discussion board. We restrict ourselves to the 184 users with *left* (96) and *right* (88) political tendencies. The dataset contains the text of several thousand posts. Quoting behavior is annotated in the dataset, so we know who quoted who. Since we are interested in classifying each user (as opposed to each post), we concatenated together all posts (excluding quoted text) written by a user. We removed punctuation and common English words, and applied stemming. We then formed *term frequency-inverse document frequency (TF-IDF)* vectors (see [8]) for each user using word types occurring 10 or more times, which resulted in 8656 unique terms.

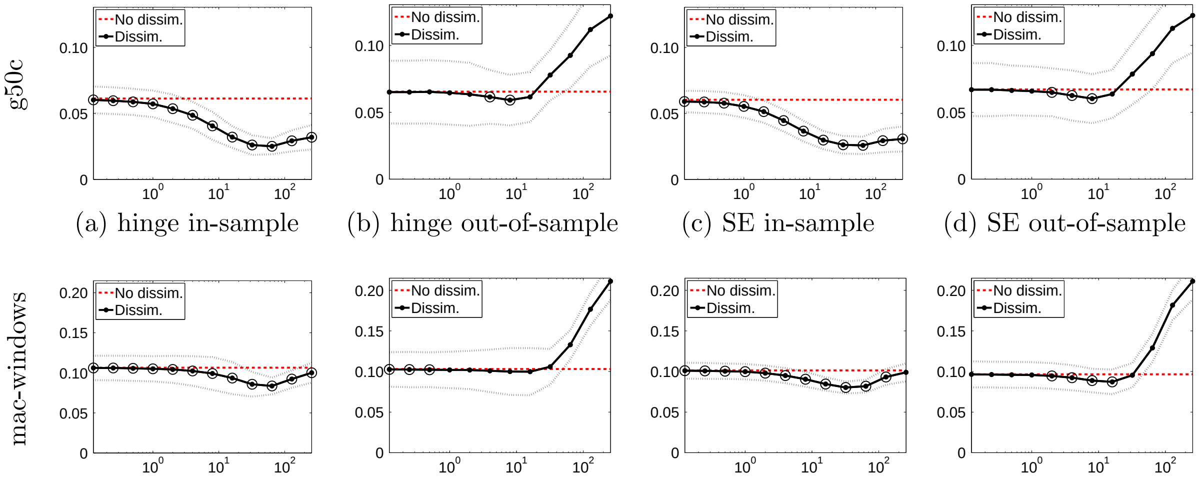
We created dissimilarity edges by the quoting behavior between users. In political discussion boards, users tend to quote posts by users with differing political views [9]. For example, users often debate a controversial issue, quoting and disputing each others' previous claims. We declare disagreement between A and B if B quotes A, and the text adjacent to the quoted text contains two or more question marks or exclamation marks, or two or more consecutive words in all capital letters (i.e., Internet shouting[1]). Consider the following illustrative example taken from the current dataset, where the user *Dixie* has quoted and responded to the user *deshrubinator*:

> *deshrubinator*: "You were the one who thought it should be investigated last week."
> *Dixie*: No I didn't, and I made it clear. You are insane! YOU are the one with NO ****ING RESPECT FOR DEMOCRACY!

We create a dissimilarity edge (A,B) if they have exhibited such seemingly hostile behavior toward each other in more than 2 posts. This thresholding ensures that we have seen multiple pieces of evidence for dissimilarity.

It is worth noting that our dissimilarity edges only need simple text processing, and can be easily defined over unlabeled data (users with unknown political view). For this experiment we do not include similarity edges, partly because the standard cosine similarity on text [8] measures similarity in *topics* (note users from different parties do talk about the same topic), rather than *sentiment*, which is more relevant to the current task. We will investigate high quality similarity edges in future work. Therefore we cannot use LapSVM or LapRLS as our baselines. Instead we

[1] We also require these words to be more than three characters long to avoid false positives from common Internet abbreviations like LOL (laugh out loud).



(a) hinge in-sample (b) hinge out-of-sample (c) SE in-sample (d) SE out-of-sample

(e) hinge in-sample (f) hinge out-of-sample (g) SE in-sample (h) SE out-of-sample

Figure 2: Changing the weight of dissimilarity edges (x-axis) in the g50c dataset (a-d) and the mac-windows dataset (e-h). y-axis is mean error rate across 4 folds with 10 random trials each. Circled settings are statistically significantly better than the baseline.

use the standard supervised SVM and RLS as baselines, respectively. Also note that, unlike our experiments with "oracle" edges, we are now including all such dissimilarity edges, some of which connect labeled and unlabeled examples. The only edges discarded are those between two labeled examples. Our scheme is realistic with noisy, "real" edges.

We used a graph of these dissimilarity edges to warp a linear kernel used in SVM and RLS classification. We set the labeled set size $l = 50$ (out of 184) and ran 10 repeated trials with randomly selected labeled examples. Out of the possible 103 dissimilarity edges derived using the above heuristics, the trials included an average of 93.4 edges (i.e., 9.6 labeled-labeled edges are ignored). On average, 40.7 examples are involved in the dissimilarity edges. Table 2 reports the mean error rate on all unlabeled examples for SVM and RLS classifiers with ("SSL") and without ("Base") dissimilarity edges. The baseline results use unwarped linear kernels. In both classifiers, we observe a statistically significant reduction in error rate ($p < 0.05$ using a two-tailed, paired t-test); it appears that the "real-world" dissimilarity edges aid classification. However upon closer inspection, we also notice the improvement comes mostly from in-sample error reduction, and it does not generalize as well to out-of-sample data like in previous experiments. We suspect this could be due to the high initial error rate.

Finally, as a post-experiment study, we investigated how many of our heuristically derived dissimilarity edges were actually consistent with the true labels. It turns out that 85 out of the 103 edges (83%) are in fact "true" dissimilarity edges. Thus, we have shown that, even if 17% of the dissimilarity edges represent false domain knowledge, we can achieve a significant improvement in overall error rate.

Table 2: Mean error rates for SVM and RLS with and without dissimilarity edges on the politics dataset. Dissimilarity is incorporated through warped kernels. Both differences are statistically significant.

Classifier	Base error rate	SSL error rate	Δ
SVM	45.67 ± 3.28	40.15 ± 4.95	5.5%
RLS	45.60 ± 3.94	37.99 ± 1.88	7.6%

5 Conclusions

We presented a convex algorithm to encode dissimilarity in semi-supervised learning. We demonstrated that when such dissimilarity domain knowledge is available, our algorithm can take advantage of it and improve classification. The major advantage of our dissimilarity encoding formulations (2) and (7) is convexity. However, they probably specify the relation between the discriminant function f at dissimilarity samples $\mathbf{x}_i$ and $\mathbf{x}_j$ more than necessary. For example in the binary case we prefer $f(\mathbf{x}_i) = -f(\mathbf{x}_j)$, while ideally it is sufficient to require $f(\mathbf{x}_i), f(\mathbf{x}_j)$ having opposite signs. Finding computationally efficient encodings for this sufficient condition is a direction for future research.

Acknowledgments

We thank Fernando Pérez-Cruz for helpful discussions on multiclass SVMs. Research supported by NSF grants CCF-0430504, CTS-0456694, CNS-0540147, and Wisconsin Alumni Research Foundation (WARF).

References

[1] Sugato Basu, Mikhail Bilenko, Arindam Banerjee, and Raymond J. Mooney. Probabilistic semi-supervised clustering with constraints. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-Supervised Learning*, pages 71–98. MIT Press, 2006.

[2] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from examples. Technical Report TR-2004-06, University of Chicago, 2004.

[3] Olivier Chapelle, Alexander Zien, and Bernhard Schölkopf, editors. *Semi-supervised learning*. MIT Press, 2006.

[4] W. Chu, V. Sindhwani, Z. Ghahramani, and S. S. Keerthi. Relational learning with gaussian processes. In *Advances in NIPS*, 2006.

[5] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2002.

[6] Nizar Grira, Michel Crucianu, and Nozha Boujemaa. Unsupervised and semi-supervised clustering: a brief survey, 2004. in 'A Review of Machine Learning Techniques for Processing Multimedia Content', Report of the MUSCLE European Network of Excellence (FP6).

[7] Yoonkyung Lee, Yi Lin, and Grace Wahba. Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99:67–81, 2004.

[8] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts, 1999.

[9] Tony Mullen and Robert Malouf. A preliminary investigation into sentiment analysis for informal political discourse. In *Proceedings of the AAAI Workshop on Analysis of Weblogs*, 2006.

[10] Pradeep Ravikumar and John Lafferty. Quadratic programming relaxations for metric labeling and markov random field MAP estimation. In *ICML06, 23rd International Conference on Machine Learning*, Pittsburgh, USA, 2006.

[11] Matthias Seeger. Learning with labeled and unlabeled data. Technical report, University of Edinburgh, 2001.

[12] Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *ICML05, 22nd International Conference on Machine Learning*, 2005.

[13] Jurgen van Gael and Xiaojin Zhu. Correlation clustering for crosslingual link detection. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[14] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained k-means clustering with background knowledge. In *International Conference on Machine Learning (ICML)*, page 577, 2001.

[15] Martin Wainwright, Tommi Jaakkola, and Alan Willsky. MAP estimation via agreement on (hyper)trees: Message passing and linear-programming approaches. *IEEE Transactions on Information Theory*, 51:3697–3717, 2005.

[16] Yair Weiss and William Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, 47, 2001.

[17] J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, 1998.

[18] Eric Xing, Andrew Ng, Michael Jordan, and Stuart Russell. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

[19] Xiaojin Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005. http://www.cs.wisc.edu/∼jerryzhu/pub/ssl_survey.pdf.

[20] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *ICML-03, 20th International Conference on Machine Learning*, 2003.

[21] Xiaojin Zhu and Andrew B. Goldberg. Semi-supervised regression with order preferences. Technical Report TR1578, Dept. of Computer Sciences, University of Wisconsin-Madison, 2006.