ROBUST GRAPH-BASED IMAGE CLASSIFIER LEARNING WITH NEGATIVE EDGE WEIGHTS

Weng-Tai Su [*], *Gene Cheung* [#], *Chia-Wen Lin* [*]

[*] National Tsing Hua University, [#] National Institute of Informatics

ABSTRACT

We study semi-supervised learning for image classifiers from a graph signal processing (GSP) perspective. Specifically, by viewing a binary classifier as a graph-signal in a high-dimensional feature space, we cast classifier learning as a signal restoration problem via a classical maximum a posteriori (MAP) formulation. Unlike previous graph-signal restoration works, we consider in addition edges with negative weights expressing dissimilarity between samples. We make two key contributions by interpreting a graph as an electrical circuit. First, for graph construction we show how "effective resistance" can guide node pair selection for negative edge insertions. Second, for classification that tolerates a small rejection rate, we define generalized smoothness on graphs that promotes ambiguity in the classifier signal, so that unsure estimated samples can be rejected. We show that generalized graph-signal smoothness is equivalent to satisfying Kirchhoff's current law (KCL) at a given node—this explains why negative edges should not be used to compute generalized smoothness on graphs. Finally, we propose an algorithm based on iterative reweighted least squares (IRLS) that solves the posed MAP problem efficiently. Simulation results show that our algorithm outperforms both SVM variants and graph-based classifiers using positive-edge graphs noticeably.

Index Terms— image classifier, graph signal processing, signal restoration

1. INTRODUCTION

Image classification from extracted features is an important and challenging problem. We focus on *semi-supervised learning*: given partially observed labels (possibly noisy) as input, train a classifier to appropriately assign labels to unclassified samples also. Among many approaches is a class of graph-based methods [1, 2, 3] that treat each sample as a node in a graph and connect it to other nodes using undirected edges, with weights that reflect distances in a high-dimensional feature space. See Fig. 1 for an example of a 8-node graph in a 2D feature space. A graph representation of the data means that properties of the graph spectrum (*e.g.*, low frequencies that are eigenvectors of the graph Laplacian matrix) can be exploited for label assignment via spectral graph theory [4].

In this paper, we extend previous graph-based works by considering in addition *negative edge weights* for binary image classifier learning. Common formulations in graph signal processing (GSP) [5] use positive edge weights that reflect inter-node *similarity*; large edge weight $w_{i,j} > 0$ means samples x_i and x_j should be similar. In contrast, negative edge weights can express *dissimilarity*: $w_{i,j} = -1$ means x_i and x_j should be different, *i.e.*, $|x_i - x_j|$ should be large. Incorporating pairwise dissimilarities into a graph should intuitively benefit image classification. For example, if edge weight $w_{1,2} = -1$ in Fig. 1, then from the graph $\mathcal{G}$ itself, one already expects x_1 and x_2 to be assigned opposite labels in a binary classifier.

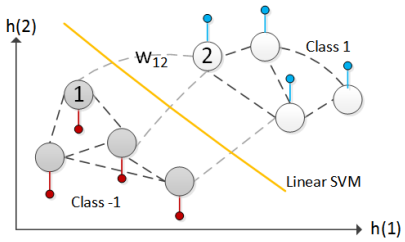
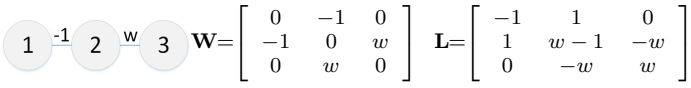


Fig. 1. Example of a graph classifier and linear SVM in 2-dimensional feature space. Graph $\mathcal{G}$ contains nodes $\mathcal{N}$ representing samples, and edges $\mathcal{E}$ with weights $w_{i,j}$ reflecting feature space distances. The classifier graph-signal takes on binary values: 1 (blue spikes) and -1 (red spikes).

To study negative edges, we view a binary image classifier as a graph-signal in a high-dimensional feature space and cast classifier learning as a signal restoration problem via a classical *maximum a posteriori* (MAP) formulation. Specifically, we make two key contributions by interpreting a graph as an electrical circuit, also done in [6, 7, 8]. First, towards optimal classification performance, we show how *effective resistance* [8]—a circuit concept that maps a network of connected resistors to a single equivalent one—can guide node pair selection for negative edge insertions during graph construction. Second, for classification that tolerates a small rejection rate, we define *generalized smoothness on graphs*—an extension of *total generalized variation* (TGV) [9] to the graph-signal domain—that promotes a suitable amount of ambiguity in the classifier signal, so that unsure estimated samples can be rejected. We show that generalized graph-signal smoothness is equivalent to satisfying *Kirchhoff's current law* (KCL) at a given node. This KCL interpretation helps explain why negative edges should not be used to compute generalized smoothness on graphs.

Having constructed a graph and defined graph-signal smoothness priors, we define a MAP problem for signal restoration. Because the graph Laplacian matrix $\mathbf{L}$ with negative edges can be indefinite, we perturb $\mathbf{L}$ by $\mathbf{\Delta}$ so that $\mathbf{L} + \mathbf{\Delta}$ is positive semi-definite (PSD), resulting in a stable signal prior. Finally, we propose an algorithm based on *iterative reweighted least squares* (IRLS) [10] that efficiently solves the posed MAP problem. Simulation results show that our algorithm outperforms SVM variants, a well-known robust classifier in the machine learning literature called *RobustBoost* [11], and graph-based classifiers using positive-edge graphs noticeably.

The outline of the paper is as follows. We first overview related works in Section 2. We then review basic GSP concepts and present our graph construction strategy based on effective resistance in Section 3. We define a generalized smoothness prior in Section 4. In Section 5, we present a label noise model and present an efficient



(a) 3-node graph (b) adjacency $\mathbf{W}$ (c) graph Laplacian $\mathbf{L}$

Fig. 2. Example of a 3-node graph with negative edges.

algorithm to solve the posed MAP problem. Finally, we present experimental results and conclusions in Section 6 and 7, respectively.

2. RELATED WORK

There exist a wide range of approaches for classifier learning with label noise, including theoretical (*e.g.*, label propagation in [12]) and application-specific (*e.g.*, inference algorithm based on multiplicative update rule [13]). In this paper, similar to previous works [1, 2, 3] we build a *graph-based classifier*, where each sample is represented as a node in a high-dimensional feature space and connects to other nearby nodes. Compared to previous graph-based classifiers, our novelties are as follows. First, we learn a classifier via a classical MAP formulation but include *negative* edge weights that reflect *dissimilarity*. This requires a careful negative edge insertion strategy, which we design based on effective resistance by interpreting a graph as an electrical circuit. Second, we show how generalized graph smoothness—extending TGV [9] to the graph-signal domain—can be interpreted intuitively as Kirchhoff's current law and used to promote ambiguity in the classifier solution.

Graph-signal smoothness priors have been used for image restoration problems such as denoising [14, 15, 16], interpolation [17] and JPEG de-quantization [18, 19]. The common assumption is that the desired signal is smooth with respect to an appropriate graph with non-negative edge weights that reflect inter-pixel similarity. In contrast, by considering negative edges we incorporate also dissimilarity information into the graph.

One alternative GSP approach is based on algebraic theory in traditional digital signal processing that relies on the shift operator [20, 2]. More concretely, instead of the graph Laplacian matrix $\mathbf{L}$, the adjacency matrix $\mathbf{W}$ is used as the variation operator to define signal smoothness and graph frequencies. As an example, a smoothness prior $\|\mathbf{x} - \mathbf{W}\mathbf{x}\|_p^p$, $p \in \mathbb{I}^+$, was proposed in [2]. When edge weights are negative, however, such smoothness prior can become insensible. Consider the three-node graph in Fig. 2. Assuming $p = 2$, the smoothness prior when $w = -1$ is:

$$\|\mathbf{x} - \mathbf{W}\mathbf{x}\|_2^2 = \|(\mathbf{I} - \mathbf{W})\mathbf{x}\|_2^2 = \left\| \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \right\|_2^2$$
$$= (x_1 + x_2)^2 + (x_1 + x_2 + x_3)^2 + (x_2 + x_3)^2$$

It is not clear why minimizing different subset sums is sensible, despite having two negative edges that signify dissimilarity.

This observation motivates our current study to use $\mathbf{x}^T\mathbf{L}\mathbf{x}$ as the appropriate smoothness prior; for the same example,

$$\mathbf{x}^T\mathbf{L}\mathbf{x} = -1(x_1 - x_2)^2 + w(x_2 - x_3)^2 \tag{1}$$

which promotes a *large* difference between node 1 and 2, and promotes a large or small difference between node 2 and 3 depending on the sign of w. This is clearly a more sensible prior. Of course, direct use of $\mathbf{x}^T\mathbf{L}\mathbf{x}$ can lead to numerical problems, and thus $\mathbf{L}$ must be first perturbed appropriately. We will discuss this in later sections.

Recently, the control community have studied the conditions where negative edges would cause a graph Laplacian to be indefinite [6, 7]. We will leverage a theoretical result from [7] in a later section for negative edge insertion during graph construction.

3. GRAPH CONSTRUCTION

3.1. Graph Definition

A graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$ has a set $\mathcal{V}$ of N nodes and a set $\mathcal{E}$ of M edges. Each edge $(i, j) \in \mathcal{E}$ connecting nodes i and j is undirected with weight $w_{i,j}$, which can be positive *or* negative. A negative $w_{i,j}$ means that connected samples are *dissimilar*—the samples are expected to have very different values. A graph-signal $\mathbf{x} \in \mathbb{R}^N$ on $\mathcal{G}$ is a discrete signal of dimension N—one label x_i for each node (sample) i in $\mathcal{V}$.

3.2. Graph Spectrum

Given edge weight (adjacency) matrix $\mathbf{W}$, we define a diagonal *degree matrix* $\mathbf{D}$, where $d_{i,i} = \sum_j w_{i,j}$. A *combinatorial graph Laplacian matrix* $\mathbf{L}$ is $\mathbf{L} = \mathbf{D} - \mathbf{W}$ [5]. Because $\mathbf{L}$ is symmetric, it can be eigen-decomposed into (Spectral Theorem):

$$\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T \tag{2}$$

where $\mathbf{\Lambda}$ is a diagonal matrix containing real eigenvalues λ_k, and $\mathbf{V}$ is an eigen-matrix composed of orthogonal eigenvectors $\mathbf{v}_i$ as columns. If $w_{i,j}$ are non-negative, then $\mathbf{L}$ must be *positive semi-definite* (PSD), meaning that $\lambda_k \geq 0, \forall k$ and $\mathbf{x}^T\mathbf{L}\mathbf{x} \geq 0, \ \forall \mathbf{x}$. Non-negative eigenvalues λ_k can be interpreted as *graph frequencies*, and eigenvectors $\mathbf{v}_k$ interpreted as corresponding graph frequency components. Together they define the *graph spectrum* for graph $\mathcal{G}$.

In this paper, we consider also negative $w_{i,j}$, and thus eigenvalues λ_k can be negative and $\mathbf{L}$ can be indefinite. It is then hard to interpret $\mathbf{L}$'s eigenvalues λ_k as frequencies.

3.3. Graph-Signal Smoothness Prior

For graph $\mathcal{G}$ with positive edge weights, signal $\mathbf{x}$ is considered *smooth* if each label x_i on node i is similar to labels x_j on neighboring nodes j with large $w_{i,j}$. In the graph frequency domain, it means that $\mathbf{x}$ contains mostly low graph frequency components; *i.e.*, coefficients $\boldsymbol{\alpha} = \mathbf{V}^T\mathbf{x}$ are zeros or very small for high frequencies. The smoothest signal is the constant vector $\mathbf{1}$—the first eigenvector $\mathbf{v}_1$ for $\mathbf{L}$ corresponding to the smallest eigenvalue $\lambda_1 = 0$.

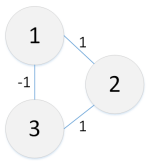
Mathematically, we can write that a signal $\mathbf{x}$ is smooth if its *graph Laplacian regularizer* $\mathbf{x}^T\mathbf{L}\mathbf{x}$ is small [14, 15, 16]. Graph Laplacian regularizer can be expressed as:

$$\mathbf{x}^T\mathbf{L}\mathbf{x} = \sum_{(i,j)\in\mathcal{E}} w_{i,j}\,(x_i - x_j)^2 = \sum_k \lambda_k\,\alpha_k^2 \tag{3}$$

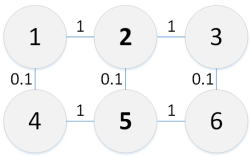
Because $\mathbf{L}$ is PSD, $\mathbf{x}^T\mathbf{L}\mathbf{x}$ is lower-bounded by 0.

When edge weights can be negative, then $\mathbf{L}$ can be indefinite, which means that there exists an eigenvector $\mathbf{v}_1$ corresponding to a negative value $\lambda_1 < 0$. The consequence is that $\mathbf{x}^T\mathbf{L}\mathbf{x}$ in (3) is not a stable prior for minimization, as $\mathbf{x} = \infty\mathbf{v}_1$ would be a pathological optimal solution to a minimization problem.

Our solution is to perturb $\mathbf{L}$ so that it is PSD again. Specifically, we define a *generalized graph Laplacian* [21] $\mathbf{L}_g = \mathbf{L} + |\lambda_1|\mathbf{I}$, where $\lambda_1 < 0$ is the smallest eigenvalue of $\mathbf{L}$, and $\mathbf{I}$ is the identity matrix. $|\lambda_1|\mathbf{I}$ effectively increases each eigenvalue of $\mathbf{L}$ by $|\lambda_1|$, so that $\mathbf{L}_g$ is PSD. The resulting eigenvalues are non-negative, and thus can again be interpreted as graph frequencies. Note that $\mathbf{L}_g$ can still be eigen-decomposed using the *same* eiganvectors $\mathbf{V}$. We will use $\mathbf{x}^T\mathbf{L}_g\mathbf{x}$ as the new smoothness prior in the sequel.



(a) 3-node graph



(b) 6-node graph

Fig. 3. Example of a 3-node graph with one negative edge and a 6-node graph with two clusters of three nodes each.

3.4. Graph Construction with Negative Edges

Towards the best possible classification performance, graph construction with negative edges requires a great deal of care. We now detail our construction method. We first construct a graph $\mathcal{G}$ with nodes $\mathcal{V}$ representing N samples. For each sample i, we compute a *feature vector* $\mathbf{h}_i$ of some dimension D. Then we can compute *positive* edge weight $w_{i,j}$ using a Gaussian kernel, as similarly done in previous graph-based classifiers [1, 2, 3]:

$$w_{i,j} = \exp\left(-\frac{(\mathbf{h}_i - \mathbf{h}_j)^T \mathbf{\Xi} (\mathbf{h}_i - \mathbf{h}_j)}{\sigma_h^2}\right) \tag{4}$$

where σ_h is a parameter. $\mathbf{\Xi}$ is a $D \times D$ diagonal matrix, where $\Xi_{i,i} \in \mathbb{R}^+$ is a feature parameter for the i-th feature. We can then assign positive edge weights $w_{i,j}$ to ω nearest neighbors j of node i, while the rest of the nodes have no edges[1] to i.

We next augment graph $\mathcal{G}$ with negative edges. In general, more negative edges means that graph Laplacian $\mathbf{L}$ may potentially become more indefinite, thus requiring a larger perturbation $|\lambda_1|\mathbf{I}$ to make $\mathbf{L} + |\lambda_1|\mathbf{I}$ PSD. Thus in practice we add only a small number of negative edges to $\mathcal{G}$, that nonetheless leads to noticeable improvement in classification results, as shown in Section 6.

3.4.1. Edge Consistency

One key challenge in inserting a negative edge is that it can be *inconsistent* with existing positive edges in the graph $\mathcal{G}$. As an illustration, we see that in Fig. 3(a), a negative edge $(1, 3)$ with weight -1 is not consistent with the two positive edges $(1, 2)$ and $(2, 3)$ with weight 1, because nodes 1 and 3 cannot be dissimilar and nodes $1, 2, 3$ be similar at the same time.

To ensure edge consistency, before we insert a negative edge, we first perform *clustering* [22] based on edge graph weights into two node groups, so that a negative edge can be drawn from a node in one group to a node in the opposing group. As an example, in Fig. 3(b), we see a 6-node graph with two groups of three nodes each: $\{1, 2, 3\}$ and $\{4, 5, 6\}$. We next devise a strategy to identify a node in each group for negative edge insertion.

3.4.2. Effective Resistance

It has been demonstrated [6, 7, 8] that insights can be obtained by interpreting a graph as an *electircal circuit*, where a positive edge weight is viewed as *conductance* (the inverse of *resistance*). We will do so here to select nodes for negative edge insertion.

We first review a theorem in [7], stating that for a graph Laplacian $\mathbf{L}$ to be definite, the absolute value of a negative edge weight $w^-_{i,j}$ must be no larger than its *effective resistance* $R_{i,j}(\mathcal{G}^+)$ computing from the graph $\mathcal{G}^+$ with only positive edges, *i.e.*,

$$|w^-_{i,j}| \leq R_{i,j}(\mathcal{G}^+)^{-1} \tag{5}$$

The effective resistance $R_{i,j}(\mathcal{G}^+)$ between nodes i and j in a graph $\mathcal{G}^+$ with positive edges can be computed as [8]:

$$R_{i,j}(\mathcal{G}^+) = Q_{i,i} + Q_{j,j} - 2Q_{i,j} \tag{6}$$

where $\mathbf{Q}$ is the Moore-Penrose pseudo-inverse of the graph Laplacian $\mathbf{L}^+$ for graph $\mathcal{G}^+$.

Recall from Section 3.3 that an indefinite $\mathbf{L}$ is perturbed by $|\lambda_1|\mathbf{I}$ such that $\mathbf{L} + |\lambda_1|\mathbf{I}$ is PSD. Clearly, an indefinite $\mathbf{L}$ with a smaller magnitude $|\lambda_1|$ would require less perturbation, and thus is more desirable. From (5), a node pair (i, j) with a large $R_{i,j}(\mathcal{G}^+)^{-1}$ (small effective resistance $R_{i,j}(\mathcal{G}^+)$) is more likely to tolerate an inserted negative edge with weight $w^-_{i,j}$; *i.e.*, the resulting $|\lambda_1|$ is likely smaller. It turns out that connecting centroids of clusters often leads to small effective resistance. We thus assign negative edges into $\mathcal{G}$ as follows. We find $2k$ clusters in $\mathcal{G}$ if k negative edges are targeted for insertion. For each cluster, we identify a centroid, and we connect pairs of centroids that are furthest in weighted feature distance with edges of negative weights (*e.g.* -1).

4. GENERALIZED GRAPH-SIGNAL SMOOTHNESS

We next describe a generalized version of the graph-signal smoothness prior (3) for classifier signal reconstruction.

4.1. Positive Edges for Generalized Smoothness

Like TGV for images [9], we can define a higher-order notion of smoothness for graph-signals using *positive* edge weights. Specifically, graph Laplacian $\mathbf{L}^+$ defined using only positive edges is related to the second derivative of continuous functions [5], and so $\mathbf{L}^+\mathbf{x}$ computes the second-order difference on graph-signal $\mathbf{x}$. As an example, the 3-node line graph in Fig. 2 with all edge weights equal to 1 has the following $\mathbf{L}^+$:

$$\mathbf{L}^+ = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} \tag{7}$$

Using the second row $\mathbf{L}^+_{2,:}$ of $\mathbf{L}^+$, we can compute the second-order difference at node x_2:

$$\mathbf{L}^+_{2,:}\mathbf{x} = -x_1 + 2x_2 - x_3 \tag{8}$$

On the other hand, the definition of second derivative[2] of a function $f(x)$ is:

$$f''(x) = \lim_{h \to 0} \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} \tag{9}$$

We see that (8) and (9) are computing the same quantity (with a sign change) in the limit.

Hence if $|\mathbf{L}^+\mathbf{x}|$ is small, then the second-order difference of $\mathbf{x}$ is small, or the first-order difference of $\mathbf{x}$ is smooth or changing slowly. In other words, the *gradient* of the signal is smooth with respect to the graph. We express this notion by stating that the square of the l_2-norm of $\mathbf{L}^+\mathbf{x}$ is small:

$$\|\mathbf{L}^+\mathbf{x}\|_2^2 = \mathbf{x}^T(\mathbf{L}^+)^T\mathbf{L}^+\mathbf{x} = \mathbf{x}^T(\mathbf{L}^+)^2\mathbf{x} = \sum_i (\lambda_i^+)^2\alpha_i^2 \tag{10}$$

where (10) is true since $\mathbf{L}^+$ is symmetric by definition.

[1] If this relationship is not symmetric, *i.e.*, if i is one of ω closest neighbors to j but j is not one of ω closest neighbors to i, then we keep edge (i, j) of weight $w_{i,j}$ anyway. Thus each node has $\geq \omega$ neighbors.

[2] https://en.wikipedia.org/wiki/Second_derivative

4.2. Negative Edges for Generalized Smoothness

We demonstrate now that including negative edges when computing generalized smoothness can be problematic. Consider again the three-node line graph in Fig. 2, where $w = 1$. The corresponding second row of the graph Laplacian $\mathbf{L}$ is:

$$\mathbf{L}_{2,:} = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} \tag{11}$$

Hence when we compute the generalized smoothness $|\mathbf{L}\mathbf{x}|$ at x_2, we get $|\mathbf{L}_{2,:}\mathbf{x}| = |x_1 - x_3|$; *i.e.,* the generalized smoothness at x_2 does not actually depend on the value of x_2! We next provide an intuitive explanation why negative edge weights should not be used next.

4.3. Circuit Interpretation of Generalized Smoothness

We again interpret an undirected weighted graph $\mathcal{G}$ as an electrical circuit to understand generalized smoothness for graph-signals. Suppose we interpret an edge (i, j) as a wire between nodes i and j, and an edge weight $w_{i,j}$ as *conductance* (equivalently, $1/w_{i,j}$ as the *resistance*) between its two endpoints. Let x_i and x_j represent the voltage at the two endpoints. According to *Ohm's law*[3], the *current* $c_{i,j}$ between the two nodes is the voltage difference at the entpoints times the conductance:

$$c_{i,j} = w_{i,j}(x_i - x_j) \tag{12}$$

By *Kirchhoff's current law*[4] (KCL), the net sum of the currents flowing into a node is zero. Applying KCL to node 2 in the three-node line graph in Fig. 2 connected by weights $w_{1,2}$ and $w_{2,3}$, we can write:

$$w_{1,2}(x_1 - x_2) + w_{2,3}(x_3 - x_2) = 0 \tag{13}$$

using Ohm's law (12).

If we desire a signal $\mathbf{x}$ to satisfy this condition maximally, we can minimize the absolute value of this current sum:

$$\min_{\mathbf{x}} \left| \begin{bmatrix} -w_{1,2} & (w_{1,2} + w_{2,3}) & -w_{2,3} \end{bmatrix} \mathbf{x} \right| = \left| \mathbf{L}^{+}_{2,:}\mathbf{x} \right| \tag{14}$$

This is in fact the generalized graph-signal smoothness condition we discussed earlier. Thus we can conclude the following: *a graph-signal* $\mathbf{x}$ *on graph* $\mathcal{G}$ *that is perfectly generalized smooth, i.e.* $|\mathbf{L}^{+}\mathbf{x}| = \mathbf{0}$*, is a voltage signal on* $\mathcal{G}$ *that satisfies KCL.*

This electrical circuit interpretation also provides an argument why negative edges should not be considered for generalized smoothness. As done in [8], a generalized graph Laplacian $\mathbf{L}_g$ with diagonal element $L_{i,i} \geq \sum_{j|j\neq i} L_{i,j}$ can be considered a conductance matrix, where an edge (i, j) has branch conductance $-L_{i,j}$ and node i has shunt conductance $L_{i,i} - \sum_{j|j\neq i} L_{i,j} \geq 0$. For such a resistive circuit, Ohm's law is applicable directly and thus KCL is meaningful. However, a negative conductance $L_{i,j} < 0$ (equivalently, a negative resistance) means the circuit is no longer resistive, and Ohm's law is not applicable. As a result, KCL—by extension generalized graph-signal smoothness—is no longer meaningful.

4.4. Interpretation of Smoothness Priors for Classifiers

We interpret the two smoothness terms in the context of binary classification. We know that the *true* signal $\mathbf{x}$ is indeed *piecewise constant* (PWC); each true label x_i is binary, and labels of the same class cluster together in the same region. The graph-signal smoothness term in (3), analogous to the total variation (TV) prior in image restoration, promotes a PWC signal $\hat{\mathbf{x}}$ during reconstruction, as empirically demonstrated in previous graph-signal restoration works [14, 15, 16, 18, 19]. Hence the smoothness prior is appropriate.

Recall that the purpose of TGV [9] is to avoid over-smoothing a *ramp* (linear increase / decrease in pixel intensity) in an image, which would happen if only a TV prior is used. A ramp in the reconstructed signal $\hat{\mathbf{x}}$ in our classification context would mean an assignment of label other than -1 and 1, which can reflect the *confidence level* in the estimated label; *e.g.*, a computed label $\hat{x}_i = 0.3$ would mean the classifier has determined that event i is more likely to be 1 than -1, but the confidence level is not high. We can thus conclude that *the generalized smoothness prior can promote an appropriate amount of ambiguity in the classification solution instead of forcing the classifier to make hard binary decisions.*

5. OPTIMIZATION

We now formulate the classifier learning problem with noisy labels. We first describe our chosen label noise model. Given the graph-signal smoothness notions previously discussed, we then formulate the optimization problem with both priors. Finally, we propose an efficient algorithm to solve the posed problem.

5.1. Label Noise Model

To model binary label noise, we adopt a uniform noise model [23], where the probability of observing $y_i = x_i$, $1 \leq i \leq K$, is $1 - p$, and p otherwise; *i.e.*,

$$Pr(y_i|x_i) = \begin{cases} 1-p & \text{if } y_i = x_i \\ p & \text{o.w.} \end{cases} \tag{15}$$

This model is motivated by the following observation in social media analysis: when labels are assigned manually by non-experts via *crowd-sourcing* [23], workers are often unreliable (*e.g.*, a non-expert is not competent in a label assignment task but pretends to be, or he simply assigns label randomly to minimize mental effort). Thus observations $\mathbf{y}$ may result in label errors that are uniform and independent.

The probability of observing a noise-corrupted $\mathbf{y}$ given ground truth $\mathbf{x}$ is hence:

$$Pr(\mathbf{y}|\mathbf{x}) = p^k(1-p)^{K-k}, \qquad k = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_0 \tag{16}$$

where $\mathbf{H} \in \{0, 1\}^{K\times N}$ is a binary matrix that extracts K components from $\mathbf{x}$ corresponding to $\mathbf{y}$. (16) serves as the likelihood term for this noise model (15). The negative log of this likelihood is:

$$-\log Pr(\mathbf{y}|\mathbf{x}) = k \underbrace{(\log(1-p) - \log(p))}_{\gamma} - K\log(1-p) \tag{17}$$

Because the second term is a constant for fixed K and p, we can ignore it during minimization.

5.2. Iterative Reweighted Least Squares Algorithm

We can now combine the likelihood (16) and the two graph-signal smoothness priors together to define an optimization objective:

$$\min_{\mathbf{x}} \ \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_0 \, \gamma \ + \ \sigma_0^{-2} \, \mathbf{x}^T \, \mathbf{L}_g \, \mathbf{x} \ + \ \sigma_1^{-2} \, \mathbf{x}^T \, (\mathbf{L}^{+})^2 \, \mathbf{x} \tag{18}$$

To solve (18), we employ the following strategy. We first replace the l_0-norm in (18) with a weighted l_2-norm:

$$\min_{\mathbf{x}} \ (\mathbf{y}-\mathbf{H}\mathbf{x})^T\mathbf{B}(\mathbf{y}-\mathbf{H}\mathbf{x})\gamma + \sigma_0^{-2} \, \mathbf{x}^T \, \mathbf{L}_g \, \mathbf{x} + \sigma_1^{-2} \, \mathbf{x}^T \, (\mathbf{L}^{+})^2 \, \mathbf{x} \tag{19}$$

[3] https://en.wikipedia.org/wiki/Ohm%27s_law

[4] https://en.wikipedia.org/wiki/Kirchhoff%27s_circuit_laws

where $\mathbf{B}$ is a $K \times K$ diagonal matrix with weights $b_1, \ldots, b_K$ on its diagonal. In other words, the fidelity term is now a weighted sum of label differences: $(\mathbf{y}-\mathbf{H}\mathbf{x})^T\mathbf{B}(\mathbf{y}-\mathbf{H}\mathbf{x}) = \sum_{i=1}^{K} b_i(y_i - \mathbf{H}_{i,:}\mathbf{x})^2$.

The weights b_i should be set so that the weighted l_2-norm mimics the l_0-norm. To accomplish this, we employ the *iterative reweighted least squares* (IRLS) strategy [10], which has been proven to have superlinear local convergence, and solve (19) iteratively, where the weights $b_i^{(t+1)}$ of iteration $t+1$ is computed using solution $x_i^{(t)}$ of the previous iteration t, *i.e.*,

$$b_i^{(t+1)} = \frac{1}{(y_i - \mathbf{H}_{i,:}\mathbf{x}^{(t)})^2 + \epsilon} \tag{20}$$

for a small $\epsilon > 0$ to maintain numerical stability. Using this weight update, we see that the weighted quadratic term $(\mathbf{y} - \mathbf{H}\mathbf{x})^T\mathbf{B}(\mathbf{y} - \mathbf{H}\mathbf{x})$ mimics the original l_0-norm $\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_0$ in the original objective (18) when the solution $\mathbf{x}$ converges.

5.2.1. Closed-Form Solution per Iteration

For a given $\mathbf{B}$, objective (19) is an unconstrained quadratic programming problem with three quadratic terms. One can thus derive a closed-form solution by taking the derivative with respect to $\mathbf{x}$ and equating it to zero, resulting in:

$$\mathbf{x}^* = \left(\gamma\mathbf{H}^T\mathbf{B}\mathbf{H} + \sigma_0^{-2}\mathbf{L}_g + \sigma_1^{-2}(\mathbf{L}^+)^2\right)^{-1}\gamma\mathbf{H}^T\mathbf{B}^T\mathbf{y} \tag{21}$$

5.3. Interpreting Computed Solution $\hat{\mathbf{x}}$

After the IRLS algorithm converges to a solution $\hat{\mathbf{x}}$, we interpret the classification results as follows. We perform thresholding by a pre-defined value τ on $\hat{\mathbf{x}}$ to divide it into three parts, including the rejection option for ambiguous labels:

$$x_i = \begin{cases} 1, & x_i^* > \tau \\ \text{Rejection}, & -\tau \le x_i^* \le \tau \\ -1, & x_i^* < -\tau. \end{cases} \tag{22}$$

Typically, threshold τ is set per application requirement. Eliminating more ambiguous labels leads to a larger rejection rate and a smaller classification error rate.

6. EXPERIMENTATION

6.1. Experiment Setup

6.1.1. Datasets for Training and Testing

To evaluate the performances of different classification methods, we selected three two-class datasets. The first is a face gender dataset provided in [24], which consists of 7900 face images (395 individuals, 20 images per individual). We extract the Local Binary Pattern (LBP) features to represent the faces for gender classification.

The second is a "faces and non-faces" dataset [25] that contains 2429 face images and 4548 non-face images, where Histogram of Oriented Gradients (HOG) features are extracted to represent the face and non-face images.

The third is the "Phoneme" dataset [26] that provides the values of five categorical attributes to distinguish nasal sounds from oral sounds.

For our experiments, we randomly sampled 400, 200, and 300 instances from the first, second, and third datasets, respectively, and used 70% of the samples as training data and 30% as testing data. We repeated the process 100 times for each dataset and then calculated the average performance of the 100 experiments in terms of classification error rate.

Table 1. Classification error rates in the Face gender dataset for competing schemes under different training label error rates (the numbers in the parentheses of the last row indicate the rejection rates)

% label noise	0%	5%	10%	15%	20%
SVM-Linear	17.65%	18.22%	18.77%	19.59%	21.60%
SVM-RBF	12.14%	12.16%	12.83%	16.30%	24.01%
RobustBoost	9.15%	11.09%	14.36%	17.36%	20.68%
Graph-Pos	13.15%	13.62%	14.38%	15.39%	16.54%
Proposed	**1.44%**	**2.96%**	**4.46%**	**5.88%**	**8.07%**
Proposed-Rej	0.21% (9.64%)	0.59% (9.36%)	1.06% (9.51%)	2.31% (9.55%)	3.91% (9.56%)

Table 2. Classification error rates in the face / non-face dataset for competing schemes under different training label error rates (the numbers in the parentheses of the last row indicate the rejection rates)

% label noise	0%	5%	10%	15%	20%
SVM-Linear	20.80%	21.84%	24.97%	28.24%	31.66%
SVM-RBF	20.80%	21.58%	23.96%	26.95%	30.03%
RobustBoost	22.83%	24.29%	25.87%	29.57%	31.24%
Graph-Pos	19.18%	20.02%	21.78%	23.35%	25.05%
Proposed	**18.72%**	**19.85%**	**21.50%**	**23.14%**	**24.92%**
Proposed-Rej	15.22% (9.28%)	16.05% (9.95%)	17.53% (9.84%)	19.40% (9.90%)	20.89% (9.91%)

6.1.2. Graph Construction

To construct a graph with negative edge weights for our proposed methods, we first constructed a graph with positive edge weights. For each sample (node), we found its three nearest neighbors according to the Euclidean distances between the node and its neighbors, and connected these nodes using edges with positive weights that are normalized to [0,1] using the Gaussian kernel given in (4). We subsequently performed clustering on the graph and found the centroids (chosen from labeled nodes) of clusters as explained in Section 3.4. We then paired the cluster centroids with different labels, and assigned a negative edge weight between each pair with a value normalized to $[-1,0]$ where the magnitude is inversely proportional to the Euclidean distance between the pair. In the experiments, we added two negative edges in each constructed graph.

6.1.3. Comparison Schemes

We tested our proposed algorithm against five schemes: i) linear SVM, ii) SVM with an RBF kernel (named SVM-RBF), iii) a more robust version of the famed AdaBoost called RobustBoost [11] that claims robustness against label noise, and iv) a graph classifier with the graph-signal smoothness prior (3) where the edge weights of the graph are all positive (named Graph-Pos).

We implemented two variants of our proposed graph classifier: i) our proposed perturbation method (named Proposed) using the combinatorial graph Laplacian $\mathbf{L}_g$ but without the generalized smoothness term (*i.e.*, $\sigma_1^{-2} = 0$ in (18) and $\tau = 0$ in (22)), and vii) the pro-

Table 3. Classification error rates in the Phoneme dataset for competing schemes under different training label error rates (the numbers in the parentheses of the last row indicate the rejection rates)

% label noise	0%	5%	10%	15%	20%
SVM-Linear	21.83%	23.35%	24.55%	25.05%	25.64%
SVM-RBF	16.63%	16.84%	17.48%	17.72%	19.34%
RobustBoost	12.81%	14.91%	17.94%	19.33%	21.50%
Graph-Pos	13.22%	14.91%	16.79%	18.17%	20.70%
Proposed	**11.80%**	**13.38%**	**15.76%**	**17.09%**	**19.31%**
Proposed-Rej	9.52% (9.73%)	11.16% (9.61%)	13.44% (9.57%)	14.64% (9.51%)	16.75% (9.36%)

posed method in (18) with rejection (named Proposed-Rej) where the rejection rate is controlled to be within 9–10% by parameter tuning.

6.2. Performance Evaluation

To test the robustness of different classification schemes against label error, we randomly selected a portion of samples from the training set and reversed their labels. All the classifiers were then trained using the same set of features and labels. Each test set was classified by the classifiers and the results are compared with the ground-truth labels. The resulting classification error rates for the three datasets using different classifiers are presented in Tables 1–3, where the percentage of randomly erred training labels ranges from 0% to 20%. The comparisons show that our proposed scheme achieved lower classification error when compared to five competing schemes under almost all training label error rates. The parameters $(\gamma, \sigma_0, \sigma_1, \tau)$ used for the three datasets respectively are: $(1, 0.1, 1, [0.0018, 0.057])$,$(1, 1, 10, [0.138, 0.185])$, $(1, 1, 2, [0.011, 0.033])$. Compared to the graph classifier with all positive edge weights, our results show that adding negative edge weights can effectively improve the classification accuracy by 0.13–11.71%. By allowing a certain amount of ambiguous samples to remain unlabeled (less than 10% rejection rate in our experiments), our proposed generalized graph-signal smoothness prior can further improve the classification accuracy.

7. CONCLUSION

In this paper we learn a graph-based image classifier by viewing the classifier as a graph-signal in a high-dimensional feature space. Unlike previous graph-based classifiers, we consider in addition edges with negative weights that express dissimilarity between sample pairs. By interpreting a graph as an electrical circuit, we first derive a condition from which we can optimally insert negative edges. We then show that a generalized smoothness prior can promote ambiguity in the classifier signal, so that estimated labels with low confidence can be rejected. Experimental results show that our proposal outperforms SVM variants and previous graph-based classifiers using positive-edge graphs noticeably.

8. REFERENCES

[1] A. Guillory and J. Bilmes, "Label selection on graphs," in *Twenty-Third Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, December 2009.

[2] S. Chen, A. Sandryhaila, J. Moura, and J. Kovacevic, "Signal recovery on graphs: Variation minimization," in *IEEE Transactions on Signal Processing*, September 2015, vol. 63, no.17, pp. 4609–4624.

[3] A. Gadde, A. Anis, and A. Ortega, "Active semi-supervised learning using sampling theory for graph signals," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, August 2014.

[4] F. Chung, *Spectral Graph Theory*, American Mathematical Society, 1996.

[5] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," in *IEEE Signal Processing Magazine*, May 2013, vol. 30, no.3, pp. 83–98.

[6] D. Zelazo and M. Burger, "On the definiteness of the weighted Laplacian and its connection to effective resistance," in *53rd IEEE Conference on Decision and Control*, Los Angeles, CA, December 2014.

[7] Y. Cheng, S. Z. Khong, and T. T. Georgiou, "On the definiteness of graph laplacians with negative weights: Geometrical and passivity-based approaches," in *2016 American Control Conference*, Boston, MA, July 2016.

[8] F. Dörfler and F. Bullo, "Kron reduction of graphs with applications to electrical networks," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, January 2013, vol. 60, no.1, pp. 150–163.

[9] K. Bredies and M. Holler, "A TGV-based framework for variational image decompression, zooming and reconstruction. Part I: Analytics," in *SIAM Jour*, 2015, vol. 8, no.4, pp. 2814–2850.

[10] I. Daubechies, R. Devore, M. Fornasier, and S. Gunturk, "Iteratively re-weighted least squares minimization for sparse recovery," in *Communications on Pure and Applied Mathematics*, January 2010, vol. 63, no.1, pp. 1–38.

[11] Y. Freund, "A more robust boosting algorithm," May 2009, https://arxiv.org/abs/0905.2138.

[12] M. Speriosu, N. Sudan, S. Upadhyay, and J. Baldridge, "Twitter polarity classification with label propagation over lexical links and the follower graph," in *Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, July 2011.

[13] Y. Wang and A. Pal, "Detecting emotions in social media: A constrained optimization approach," in *Twenty-Fourh International Joint Conference on Artificial Intelligence*, Buenos Aires, Argentina, July 2015.

[14] J. Pang, G. Cheung, W. Hu, and O. C. Au, "Redefining self-similarity in natural images for denoising using graph signal gradient," in *APSIPA ASC*, Siem Reap, Cambodia, December 2014.

[15] J. Pang, G. Cheung, A. Ortega, and O. C. Au, "Optimal graph Laplacian regularization for natural image denoising," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Brisbane, Australia, April 2015.

[16] J. Pang and G. Cheung, "Graph laplacian regularization for inverse imaging: Analysis in the continuous domain," in *IEEE Transactions on Image Processing*, December 2016, vol. 26, no.4, pp. 1770–1785.

[17] Y. Mao, G. Cheung, and Y. Ji, "On constructing z-dimensional DIBR-synthesized images," in *IEEE Transactions on Multimedia*, August 2016, vol. 18, no.8, pp. 1453–1468.

[18] W. Hu, G. Cheung, and M. Kazui, "Graph-based dequantization of block-compressed piecewise smooth images," in *IEEE Signal Processing Letters*, February 2016, vol. 23, no.2, pp. 242–246.

[19] X. Liu, G. Cheung, X. Wu, and D. Zhao, "Random walk graph laplacian based smoothness prior for soft decoding of JPEG images," in *IEEE Transactions on Image Processing*, February 2017, vol. 26, no.2, pp. 509–524.

[20] A. Sandryhaila and J. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," in *IEEE Signal Processing Magazine*, August 2014, vol. 31, no.5, pp. 80–90.

[21] T. Biyikoglu, J. Leydold, and P. F. Stadler, "Nodal domain theorems and bipartite subgraphs," in *Electronic Journal of Linear Algebra*, November 2005, vol. 13, pp. 344–351.

[22] J. Shi and J. Malik, "Normalized cuts and image segmentation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, August 2000, vol. 22, no.8, pp. 888–905.

[23] A. Brew, D. Greene, and P. Cunningham, "The interaction between supervised learning and crowdsourcing," in *Computational Social Science and the Wisdom of Crowds Workshop at NIPS*, Whistler, Canada, December 2010.

[24] L. Spacek, "Face recognition data, university of essex, uk," http://cswww.essex.ac.uk/mv/allfaces/faces94.html, February 2007.

[25] "Cbcl face database," http://www.ai.mit.edu/projects/cbcl, 2000.

[26] J. Alcala-Fdez et al., "Keel: A software tool to assess evolutionary algorithms to data mining problems," in *Soft Computing*, February 2009, vol. 13, no.3, pp. 307–318.