Alternating Binary Classifier and Graph Learning from Partial Labels

Cheng Yang[†‡], Gene Cheung[‡], Vladimir Stankovic[§]
[†] Zhejiang University of Technology, Hangzhou, China
E-mail: cheng@nii.ac.jp Tel/Fax: +86-571-8529-0193
[‡] National Institute of Informatics, Tokyo, Japan
E-mail: {cheng, cheung}@nii.ac.jp Tel/Fax: +81-3-4212-2150
[§] University of Strathclyde, Glasgow, UK
E-mail: vladimir.stankovic@strath.ac.uk Tel: +44-141-548-2679

Abstract—**Semi-supervised binary classifier learning is a fundamental machine learning task where only partial binary labels are observed, and labels of the remaining data need to be interpolated. Leveraging on the advances of graph signal processing (GSP), recently binary classifier learning is posed as a signal restoration problem regularized using a graph smoothness prior, where the undirected graph consists of a set of vertices and a set of weighted edges connecting vertices with similar features. In this paper, we improve the performance of such a graph-based classifier by simultaneously optimizing the feature weights used in the construction of the similarity graph. Specifically, we start by interpolating missing labels by first formulating a boolean quadratic program with a graph signal smoothness objective, then relax it to a convex semi-definite program, solvable in polynomial time. Next, we optimize the feature weights used for construction of the similarity graph by reusing the smoothness objective but with a convex set constraint for the weight vector. The reposed convex but non-differentiable problem is solved via an iterative proximal gradient descent algorithm. The two steps are solved alternately until convergence. Experimental results show that our alternating classifier / graph learning algorithm outperforms existing graph-based methods and support vector machines with various kernels[1].**

I. Introduction

In *semi-supervised binary classifier learning* [1], missing labels of a dataset need to be interpolated given a small subset of observed binary labels. In practical scenarios, labeling is often an expensive and/or tedious task, whereas unlabelled data can often be easily and cheaply obtained, *e.g.*, social media. Thus, semi-supervised classifier learning is of practical importance as data volume grows.

Leveraging on the advance of graph signal processing (GSP) [2]–[5], recent works [6], [7] pose binary classifier learning as a signal restoration problem on graphs, where the undirected graph consists of a set of nodes (each associated with a feature vector) and a set of weighted graph edges connecting similar nodes in the high-dimensional feature space. A graph smoothness prior is typically adopted to regularize the ill-posed signal restoration problem [8]–[12]. While the above formulation results in the smoothest binary classifier signal with respect to the graph while being consistent to the observed partial labels, a strong assumption is implicitly made that the underlying graph is sufficiently informative in reflecting inter-node similarities based on feature distance for binary classification. Thus, one can potentially improve classifier performance if the graph construction *and* the classifier graph-signal restoration are optimized jointly, leading to an even smaller objective function value. This is the main idea of our paper.

Our alternating binary classifier and graph learning method consists of two iterative steps. We first formulate a *boolean quadratic program* (BQP) to predict missing labels where the binary classifier is assumed smooth with respect to a fixed similarity graph computed based on inter-node weighted feature distances. We then relax the NP-hard problem to a convex semi-definite program (SDP) [13], so that the optimization can be efficiently solved [14]. Next, keeping the classifier signal fixed, we optimize the set of feature weights using the same signal smoothness objective, subject to a convex set constraint for the weights. We reformulate the problem into an unconstrained one and solve it using a *proximal gradient* (PG) method [15]. The two steps are executed iteratively until convergence. Experimental results show that our alternating binary classifier and graph learning method outperforms existing graph-based methods and support vector machines (SVM) with various kernels.

The remainder of the paper is organized as follows. We review related graph learning work in Section II. We discuss our proposed method in Section III. We evaluate the proposed method on several datasets and compare it with existing graph-based methods and SVMs with various kernels in Section IV. Finally, we conclude in Section V.

II. Related Works

The idea of alternately restoring the binary classifier signal and learning the feature weights in the similarity graph was first studied in [6]. Specifically, [6] formulated the feature graph update problem as a quadratic program and solved it via Lagrangian relaxation, and the feature weights were learned using a Newton's descent method. However, the selection of the Lagrange multiplier grossly affects the performance and is difficult to set. In addition, Newton's descent method requires

[1]The work is partly funded by the European Unions Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 734331.

inverse Hessian computation of the feature weights in every iteration, and redundant features need to be removed if the Hessian matrix is singular. Unlike [6], our PG-based scheme avoids second-order Hessian computation, and thus is much more computationally practical.

The idea of updating the feature weights in the feature graph is similar to metric learning [16], including more recent approaches with geometric mean [17] and Bayesian inference [18], that aims to learn a Mahalanobis matrix which yields small distances for relatively similar nodes and large distances for relatively dissimilar nodes. In contrast, our PG-based feature weight update scheme not only promotes distance similarity and dissimilarity among observed graph nodes, but also preserves the smoothness of the classifier signal with respect to the graph.

III. Alternating Binary Classifier and Graph Learning

Given noise-free partial binary labels, we aim to alternately: 1) interpolate the remaining labels via SDP given a fixed similarity graph; and 2) compute the "best" feature weights in the similarity graph given a restored binary classifier signal. We alternately solve the two steps until convergence. We discuss the two steps in order next.

A. Binary Classifier Learning via SDP

We first construct an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{A}\}$ with positive edge weights, where $\mathcal{G}$ consists of a set of N vertices $\mathcal{V}$, a set of edges $\mathcal{E}$ connecting $\mathcal{V}$, and a set of *edge weights* in a weighted adjacency matrix $\mathbf{A}$. An edge weight $w_{i,j}$ in $\mathbf{A}$ between nodes i and j is computed as the Gaussian of the negative weighted feature distance square:

$$w_{i,j} = \exp\left\{ -\sum_{k=1}^{K} c_k \left(f_k(i) - f_k(j)\right)^2 \right\}, \tag{1}$$

where c_k is the weight for the k-th feature, and $f_k(i)$ is the k-th feature value for node i.

We define a *degree matrix* $\mathbf{D}$ whose i-th diagonal entry is $D_{i,i} = \Sigma_{j=1}^{N} w_{i,j}$. Subsequently, we define a *combinatorial graph Laplacian matrix* [2]–[5] $\mathcal{L} := \mathbf{D} - \mathbf{A}$. Given a label vector $\mathbf{x}$ with partial labels, we now formulate a two-way partition problem as follows [14]:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{x}^\top \mathcal{L} \mathbf{x} \\ \text{s.t.} \quad & x_i^2 = 1, \quad i \in \{1, \ldots, N\} \\ & x_i = \hat{x}_i, \quad i \in \mathcal{F}, \quad \mathcal{F} \subset \{1, \ldots, N\}, \end{aligned} \tag{2}$$

where $\hat{x}_i$ in subset $\mathcal{F}$ are the observed noise-free binary labels. Minimizing the objective $\mathbf{x}^\top \mathcal{L} \mathbf{x}$ means that signal $\mathbf{x}$ should be smooth with respect to the graph $\mathcal{G}$, as done in previous graph signal restoration works [8]–[12]. The problem (2) is also known as a *boolean quadratic program* (BQP) [13], which is NP-hard due to the constraint $x_i^2 = 1$, *i.e.*, $x_i \in \{-1, 1\}$.

Since $\mathbf{x}^\top \mathcal{L} \mathbf{x} = \text{tr}(\mathbf{x}^\top \mathcal{L} \mathbf{x}) = \text{tr}(\mathcal{L} \mathbf{x}\mathbf{x}^\top)$, we define $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$, where $\mathbf{X}$ is a rank-one symmetric positive semi-definite (PSD) matrix. We now define a symmetric matrix $\mathbf{M} = [\mathbf{X}\ \mathbf{x}; \mathbf{x}^\top\ 1]$. From classical linear algebra, we know that $\mathbf{M}$ is PSD if: i) sub-matrix $\mathbf{X}$ is PSD, and ii) the Schur complement $\mathbf{M}/\mathbf{X}$ of sub-matrix $\mathbf{X}$ of matrix $\mathbf{M}$ is PSD. $\mathbf{X}$ is indeed PSD if $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$. Schur complement $\mathbf{M}/\mathbf{X} = \mathbf{X} - \mathbf{x}\mathbf{x}^\top = 0$ if $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$, and hence requiring $\mathbf{X}$ and Schur complement $\mathbf{M}/\mathbf{X}$ to be PSD is a necessary but not sufficient condition for $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$. Thus the problem (2) can be written into a relaxed version as follows:

$$\begin{aligned} \min_{\mathbf{X}} \quad & \text{tr}(\mathcal{L}\mathbf{X}) \\ \text{s.t.} \quad & X_{ii} = 1, \quad i \in \{1, \ldots, N\} \\ & \begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix} \succeq 0, \quad x_i = \hat{x}_i, \quad i \in \mathcal{F} \\ & \text{rank}(\mathbf{X}) = 1, \end{aligned} \tag{3}$$

where $\text{tr}(\cdot)$ is the trace operator. The rank constraint $\text{rank}(\mathbf{X}) = 1$ still causes (3) to be non-convex. If we relax this constraint, then (3) without $\text{rank}(\mathbf{X}) = 1$ becomes a convex SDP problem, solvable in polynomial time [14]. We take sign operation given computed $\mathbf{x}^*$ to acquire the final predicted labels, *i.e.*, $\tilde{\mathbf{x}} = \text{sign}(\mathbf{x}^*)$.

B. Feature Graph Learning via PG

Given the label signal $\tilde{\mathbf{x}}$ computed in Section III-A, if we now consider the graph Laplacian $\mathcal{L}$ as a function of feature weights c_k's, we can compute the optimal c_k's using the same graph Laplacian regularizer [2]:

$$\begin{aligned} \min_{\mathbf{c}} \quad \tilde{\mathbf{x}}^\top \mathcal{L}(\mathbf{c}) \tilde{\mathbf{x}} \quad &= \quad \min_{\mathbf{c}} \quad \sum_{i,j} w_{i,j} \left(\tilde{x}_i - \tilde{x}_j\right)^2 \\ & \text{s.t.} \left\{ \begin{array}{l} c_k \in [0, C], \forall k \\ \mathbf{1}^\top \mathbf{c} \leq C \end{array} \right. \end{aligned} \tag{4}$$

where constraint $\mathbf{1}^\top \mathbf{c} \leq C$ is necessary to prevent the trivial solution when $c_k = \infty$. For notation simplicity, denote by $F_k(i,j) = (f_k(i) - f_k(j))^2$, and $d_{i,j} = (\tilde{x}_i - \tilde{x}_j)^2$. We next define a *convex set* $\mathcal{S} = \{\mathbf{c} \mid c_k \in [0, C], \forall k, \mathbf{1}^\top \mathbf{c} \leq C\}$. One can verify $\mathcal{S}$ is a convex set since $\forall \mathbf{v}_1, \mathbf{v}_2 \in \mathcal{S}$, their convex combination $\gamma \mathbf{v}_1 + (1-\gamma)\mathbf{v}_2 \in \mathcal{S}, \forall \gamma \in [0,1]$. Note that $\mathcal{S}$ is the intersection of half-space $\mathcal{U} = \{\mathbf{c} : \mathbf{1}^\top \mathbf{c} \leq C\}$ and a box $\mathcal{T} = \text{Box}[\mathbf{0}, \mathbf{C}]$ [19], where $\mathbf{C}$ is a K-dimension vector with entries all equal to C. We then define an *indicator function* $I_{\mathcal{S}}(\mathbf{c})$ as follows:

$$I_{\mathcal{S}}(\mathbf{c}) = \left\{ \begin{array}{ll} 0 & \text{if} \ \ \mathbf{c} \in \mathcal{S} \\ \infty & \text{o.w.} \end{array} \right. \tag{5}$$

With (5), (4) can be re-written as an unconstrained problem:

$$\min_{\mathbf{c}} \quad \underbrace{\sum_{i,j} \exp\left\{ -\sum_{k=1}^{K} c_k F_k(i,j) \right\} d_{i,j}}_{g(\mathbf{c})} + \mathbf{I}_{\mathcal{S}}(\mathbf{c}) \tag{6}$$

where the first term $g(\mathbf{c}) : \mathbf{R}^K \rightarrow \mathbf{R}$ is convex with respect to $\mathbf{c}$ (we prove the convexity of $g(\mathbf{c})$ in Appendix A) and differentiable, while the second term $I_{\mathcal{S}}(\mathbf{c}) : \mathbf{R}^K \rightarrow \mathbf{R} \cup \{+\infty\}$ is convex but non-differentiable. We thus employ proximal gradient descent [15] to solve (6).

In particular, we first compute the gradient of $g(\mathbf{c})$:

$$\nabla g(\mathbf{c}) = \begin{bmatrix} -\sum_{i,j} \exp\left\{-\sum_{k=1}^{K} c_k F_k(i,j)\right\} F_1(i,j) d_{i,j} \\ \vdots \\ -\sum_{i,j} \exp\left\{-\sum_{k=1}^{K} c_k F_k(i,j)\right\} F_K(i,j) d_{i,j} \end{bmatrix} \quad (7)$$

The proximal mapping $\mathbf{prox}_{I_\mathcal{S}}(\mathbf{u})$ for the indicator function $I_\mathcal{S}(\mathbf{c})$ is an Euclidean projection onto the convex set $\mathcal{S}$, *i.e.*,

$$\begin{aligned} \mathbf{prox}_{I_\mathcal{S}}(\mathbf{u}) &= \arg\min_{\mathbf{c}\in\mathcal{S}} \frac{1}{2}||\mathbf{u}-\mathbf{c}||_2^2 \\ = \ P_\mathcal{S}(\mathbf{u}) &= \begin{cases} P_\mathcal{T}(\mathbf{u}), & \text{if } \mathbf{1}^\top P_\mathcal{T}(\mathbf{u}) \le C \\ P_\mathcal{T}(\mathbf{u}-\alpha\cdot\mathbf{1}), & \text{o.w.} \end{cases} \end{aligned} \quad (8)$$

where $P_\mathcal{T}(\mathbf{u}) = (\min\{\max\{u_k, 0\}, C\})_{k=1}^K$, and α is any positive root of $\mathbf{1}^\top P_\mathcal{T}(\mathbf{u}-\alpha\cdot\mathbf{1}) = C$ [19].

See Fig. 1 for examples of the above projection of a given point onto the intersection $\mathcal{S}$ of $\mathcal{U}$ and $\mathcal{T}$ when $K=2$.

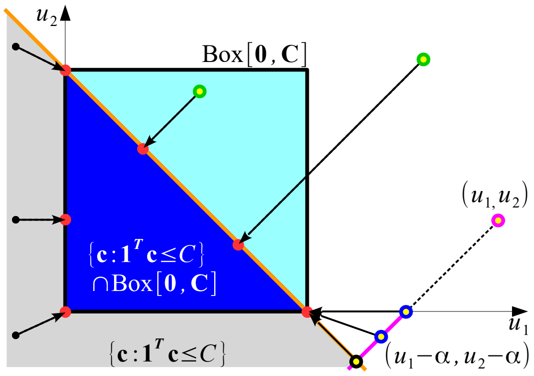
u_2
$\text{Box}[\mathbf{0},\mathbf{C}]$
(u_1,u_2)
$\{\mathbf{c}:\mathbf{1}^T\mathbf{c}\le C\}$
$\cap\text{Box}[\mathbf{0},\mathbf{C}]$
u_1
$\{\mathbf{c}:\mathbf{1}^T\mathbf{c}\le C\}$
$(u_1-\alpha, u_2-\alpha)$

Fig. 1. Illustration of $\mathbf{prox}_{I_\mathcal{S}}(\mathbf{u})$ using (8) when $K=2$, where the given points that suffice $\mathbf{1}^\top P_\mathcal{T}(\mathbf{u}) \le C$ are marked in black, and the ones that suffice $\mathbf{1}^\top P_\mathcal{T}(\mathbf{u}) > C$ are marked as green and magenta circles. For the given point, (u_1, u_2), marked as a magenta circle, the resulting point $(u_1-\alpha, u_2-\alpha)$, marked as blue and black circles, may lie anywhere on the magenta line. The projected points or points in $\mathcal{S}$ that are closest to the given points are marked in red.

The PG algorithm can thus be written as:

$$\mathbf{c}^{l+1} := P_\mathcal{S}(\mathbf{c}^l - t\nabla g(\mathbf{c}^l)). \quad (9)$$

One can use a fixed step size $t^l = t \in (0, 2/L]$ to solve (6) with a convergence rate $O(1/l)$, where L is a Lipschitz constant that requires computation of the Hessian of $g(\mathbf{c})$ [15], [20], [21]. In practice, to avoid a large computation cost, we select a step size small enough to satisfy the Lipschitz smoothness of the objective function.

IV. Experimentation

We evaluate our proposed scheme on a small *Human Quiet Stance* (`HQS`) dataset [22] with 5 classes, 50 features and 270 samples, where the features are extracted from the time series trajectories of twelve body parts. We also use the following small datasets in KEEL repository [23]: `appendicitis` (2 classes, 7 features, 106 samples), `wine` (3 classes, 13 features, 178 samples), `tae` (3 classes, 5 features, 151 samples), `led7digit` (10 classes, 7 features, 500 samples).

We solve the semi-definite relaxed version of (3) for binary classifier learning using CVX, a package for specifying and solving convex programs [24], [25], and heuristically set $C = 1000$ for the convex set $\mathcal{S}$ in (8) for feature graph learning. All experiments were performed in Matlab R2017a on an i5-7500, 8GB of RAM, Windows 10 PC. We compare our proposed method with SVMs with five different kernels: linear, quadratic, polynomial, Gaussian radial basis function (RBF) and multilayer perceptron (MLP). For optimal performance, we set polynomial kernel order 3, RBF kernel scaling factor 10, and two MLP kernel parameters $p_1 = 0.001, p_2 = -0.001$ in $\tanh(p_1\mathbf{w}\cdot\mathbf{z}+p_2)$. We also evaluate the following graph-based methods: 1) SDP-based classifier signal restoration without PG-based feature weight update, and 2) a quadratic-based classifier signal restoration formulation $\min_\mathbf{x} \mathbf{x}^\top\mathcal{L}\mathbf{x}$ s.t. $x_i = \hat{x}_i, i \in \mathcal{F}$. We adopt one against one multi-class classification strategy [26] on datasets with more than two classes for all evaluated methods. Note that neural-network-based methods, e.g., [27], are not evaluated since they generally require large training dataset.

Tables I-V present the classification error rates based on inverse q-fold cross-validation (CV) and q-fold CV, which show that our SDP binary classifier scheme outperforms all competing methods except for `HQS` dataset, where potential noise in the time series body part trajectories results in noisy extracted features and limits the classification performance of all evaluated methods. In practice, one can use our SDP binary classifier scheme (Graph-SDP) for efficient computation compared to the quadratic formulation (Graph-quadratic) given the fact that the former approximates the latter, as can be seen from Tables I, III, IV and V.

TABLE I
Classification Error Rates on `HQS`.

inverse q-fold CV	$q=9$	$q=8$	$q=7$	$q=6$
SVM-linear	1.80%	1.31%	1.12%	1.18%
SVM-quadratic	3.43%	3.04%	2.76%	2.44%
SVM-polynomial	3.01%	2.65%	2.53%	2.27%
SVM-RBF	**1.22%**	1.28%	**0.96%**	1.12%
SVM-MLP	1.78%	1.55%	1.36%	1.35%
Graph-quadratic	3.53%	3.32%	3.42%	3.41%
Graph-SDP	3.53%	3.37%	3.64%	3.50%
Proposed	1.41%	**1.15%**	1.27%	**0.94%**

TABLE II
Classification Error Rates on `appendicitis`.

inverse q-fold CV	$q=9$	$q=7$	$q=5$	$q=3$
SVM-linear	22.81%	21.31%	21.19%	21.86%
SVM-quadratic	25.85%	25.76%	25.20%	20.83%
SVM-polynomial	22.96%	23.69%	25.41%	27.00%
SVM-RBF	23.50%	24.28%	21.76%	16.27%
SVM-MLP	25.13%	27.27%	25.49%	24.29%
Graph-quadratic	19.81%	19.81%	19.81%	19.81%
Graph-SDP	19.81%	19.81%	19.81%	19.81%
Proposed	**17.09%**	**15.70%**	**14.78%**	**14.90%**

V. Conclusion

Leveraging on the advance of the graph signal processing, the semi-supervised binary classifier learning problem can be

TABLE III
CLASSIFICATION ERROR RATES ON `wine`.

inverse q-fold CV	$q=9$	$q=8$	$q=7$	$q=6$
SVM-linear	4.71%	4.32%	4.51%	3.42%
SVM-quadratic	15.12%	13.87%	12.06%	10.45%
SVM-polynomial	16.43%	16.21%	13.78%	12.69%
SVM-RBF	3.87%	3.37%	3.38%	3.00%
SVM-MLP	4.18%	3.58%	3.72%	3.51%
Graph-quadratic	33.12%	33.38%	31.87%	29.20%
Graph-SDP	43.62%	39.46%	42.52%	37.52%
Proposed	**2.96%**	**2.94%**	**2.22%**	**2.25%**

TABLE IV
CLASSIFICATION ERROR RATES ON `tae`.

	inverse q-fold CV		q-fold CV	
	$q=5$	$q=3$	$q=3$	$q=8$
SVM-linear	41.98%	40.59%	41.34%	47.02%
SVM-quadratic	42.51%	40.59%	41.34%	44.37%
SVM-polynomial	43.45%	41.42%	45.14%	48.01%
SVM-RBF	44.80%	45.50%	43.87%	42.19%
SVM-MLP	44.94%	45.70%	44.15%	41.66%
Graph-quadratic	56.39%	52.37%	44.98%	42.45%
Graph-SDP	62.57%	60.24%	65.40%	55.43%
Proposed	**41.80%**	**39.82%**	**34.71%**	**31.85%**

posed as a graph-based signal restoration problem regularized by a widely-used graph smoothness prior. In this paper, we proposed a novel graph-based semi-supervised learning method for classification that alternatively restores the binary classifier signal and learn the feature weights in the similarity graph. We do this by first predicting the remaining labels given an initial graph via a convex SDP formulation that is a relaxation of the NP-hard BQP, and then compute the optimal feature weights in the similarity graph given the restored binary classifier signal via an unconstained formulation solved using PG. Experimental results confirm that our alternating binary classifier and graph learning method outperforms existing graph-based methods and SVM's with various kernels.

APPENDIX A
CONVEXITY OF $g(\mathbf{c})$

Proof: If we denote $h_{i,j}(\mathbf{c}) = -\sum_{k=1}^{K} c_k F_k(i,j)$, then $g(\mathbf{c}) = \sum_{i,j} w_{i,j}(h_{i,j}(\mathbf{c}))d_{i,j} = \sum_{i,j} \exp\{h_{i,j}(\mathbf{c})\}\, d_{i,j}$. Since $h_{i,j}(\beta\mathbf{a} + (1-\beta)\mathbf{b}) = \beta h_{i,j}(\mathbf{a}) + (1-\beta)h_{i,j}(\mathbf{b})$, $\forall \mathbf{a}, \mathbf{b} \in \mathcal{S}, \forall \beta \in [0,1]$, $h_{i,j}(\mathbf{c})$ is both convex and concave.

Since the function $\exp\{h_{i,j}(\mathbf{c})\}$ is a composition of 1) $h_{i,j}(\mathbf{c}) : \mathbf{R}^K \to \mathbf{R}$ and 2) $w_{i,j}(h_{i,j}(\mathbf{c})) : \mathbf{R} \to \mathbf{R}$, where $w_{i,j}(h_{i,j}(\mathbf{c}))$ is twice differentiable and $w_{i,j}^{''}(h_{i,j}(\mathbf{c})) \geq 0$, besides, such operation, namely 'composition with scalar functions', preserves convexity [14], thus $w_{i,j}(h_{i,j}(\mathbf{c}))$ is convex.

Next, since $w_{i,j}(h_{i,j}(\mathbf{c}))$'s are convex functions and $d_{i,j}$'s are nonnegative scalars, and $g(\mathbf{c})$ is a nonnegative weighted sum, where the operation 'nonnegative weighted sums' preserves convexity [14], thus $g(\mathbf{c})$ is convex. ■

TABLE V
CLASSIFICATION ERROR RATES ON `led7digit`.

inverse q-fold CV	$q=35$	$q=25$	$q=15$	$q=9$	$q=7$
SVM-linear	**0.49%**	**0.16%**	0.15%	0.19%	0.18%
SVM-quadratic	1.28%	1.52%	0.90%	1.09%	1.69%
SVM-polynomial	1.24%	1.64%	0.68%	0.57%	0.53%
SVM-RBF	0.55%	0.21%	0.17%	0.19%	0.21%
SVM-MLP	0.68%	0.20%	0.16%	0.19%	0.19%
Graph-quadratic	12.20%	7.42%	2.48%	0.31%	0.21%
Graph-SDP	16.04%	12.24%	6.23%	0.81%	0.36%
Proposed	9.05%	3.18%	**0.12%**	**0.00%**	**0.00%**

REFERENCES

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

[2] D. I. Shuman et al., "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE SPM*, vol. 30, no. 3, pp. 83–98, May 2013.

[3] G. Cheung, E. Magli, Y. Tanaka, and M. K. Ng, "Graph spectral image processing," *Proc. IEEE*, vol. 106, no. 5, pp. 907–930, May 2018.

[4] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, May 2018.

[5] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE TSP*, vol. 64, no. 14, pp. 3775–3789, July 2016.

[6] Y. Mao, G. Cheung, C.-W. Lin, and Y. Ji, "Joint learning of similarity graph and image classifier from partial labels," in *APSIPA ASC*, Jeju, South Korea, Dec. 2016.

[7] G. Cheung et al., "Robust semi-supervised graph classifier learning with negative edge weights," *IEEE TSIPN*, 2018, early access.

[8] J. Pang and G. Cheung, "Graph laplacian regularization for image denoising: Analysis in the continuous domain," *IEEE TIP*, vol. 26, no. 4, pp. 1770–1785, Apr. 2017.

[9] W. Hu, G. Cheung, and M. Kazui, "Graph-based dequantization of block-compressed piecewise smooth images," *IEEE SPL*, vol. 23, no. 2, pp. 242–246, Feb. 2016.

[10] Y. Mao et al., "On constructing z-dimensional dibr-synthesized images," *IEEE TMM*, vol. 18, no. 8, pp. 1453–1468, Aug. 2016.

[11] P. Wan, G. Cheung, D. Florencio, C. Zhang, and O. C. Au, "Image bit-depth enhancement via maximum a posteriori estimation of AC signal," *IEEE TIP*, vol. 25, no. 6, pp. 2896–2909, Jun. 2016.

[12] X. Dong, "Multi-view signal processing and learning on graphs," Ph.D. dissertation, EPFL, Lausanne, Switzerland, Sept. 2014.

[13] Z.-Q. Luo et al., "Semidefinite relaxation of quadratic optimization problems," *IEEE SPM*, vol. 27, no. 3, pp. 20–34, Apr. 2010.

[14] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[15] N. Parikh and S. Boyd, "Proximal algorithms," in *Foundations and Trends in Optimization*, vol. 1, no.3, 2013, pp. 123–231.

[16] B. Kulis, "Metric learning: A survey," in *Foundations and Trends in Machine Learning*, vol. 5, no.4, 2013, pp. 287–364.

[17] P. H. Zadeh et al., "Geometric mean metric learning," in *ICML*, New York, NY, 2016.

[18] D. Wang and X. Tan, "Robust distance metric learning via bayesian inference," *IEEE TIP*, vol. 27, no. 3, pp. 1542–1553, Mar. 2018.

[19] A. Beck, *First-Order Methods in Optimization*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2017.

[20] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.

[21] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pp. 185–212, 2011.

[22] T. Yagi et al., "Comparison of righting reactions between elderly with and without stroke using solid angles," in *Asian Confederation of Physical Therapy*, Kuala Lumpur, Malaysia, Oct. 2016.

[23] J. Alcal-Fdez et al., "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, pp. 255–287, 2011.

[24] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[25] ——, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer, 2008, pp. 95–110, http://stanford.edu/~boyd/graph_dcp.html.

[26] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[27] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," in *NIPS*, Montreal, Canada, Dec. 2014.