GRAPH FILTERING WITH MULTIPLE SHIFT MATRICES

Jie Fan, *Cihan Tepedelenlioglu*, *Andreas Spanias*
SenSIP Center, School of ECEE, Arizona State University

***Abstract*—We propose a novel graph filtering method for semi-supervised classification that adopts multiple graph shift matrices to obtain more flexibility in dealing with misleading features. The resulting optimization problem is solved with a computationally efficient alternating minimization approach. In simulation experiments, we implement both conventional and our proposed graph filters as semi-supervised classifiers on real and synthetic datasets to demonstrate advantages of our algorithms in terms of classification performance.**

***Index Terms*—Graph signal processing, graph filter, semi-supervised classification, multiple graph shift matrices**

I. Introduction

In recent years, graph-structured datasets and their processing are experiencing a surge in various areas including wireless sensor networks, social networks, image processing and deep learning [1]–[4]. Toward this goal, traditional DSP theory has been generalized to graph signal processing ($\mathrm{DSP_G}$) [5]. The $\mathrm{DSP_G}$ framework enables applications of fundamental techniques, such as filtering and frequency analysis, to graph signals. $\mathrm{DSP_G}$ has been widely adopted in classification [6]–[8], approximation [9], inpainting [10] and denoising [11] of graph signals.

In this paper, we focus on applying $\mathrm{DSP_G}$ to semi-supervised classification [12]. Graph-based semi-supervised learning has been widely studied in the last twenty years. A graph mincut algorithm is proposed in [13] for binary classification. An algorithm based on Markov random walks utilize a manifold structure to classify data [14]. Label propagation approaches [15]–[17] find the most likely state configuration and can address classification problems with more than two categories. Similar to the Markov random walk algorithm in [14], label propagation approaches utilize a manifold structure. Inspired by label propagation algorithms, $\mathrm{DSP_G}$ utilizes graph filter to solve the semi-supervised classification which was pioneered in [5], [6].

Unlike label propagation algorithms directly updating the labels in their manifold processes, $\mathrm{DSP_G}$ methods update filter parameters to obtain a well-trained graph filter as the classifier, which allows less parameters in optimization. In $\mathrm{DSP_G}$, the graph filter exploits information among both labeled and unlabeled nodes. After filtering, the initially unlabeled nodes are assigned with predicted labels and then classification is completed. However, as label propagation algorithms rely heavily on the initial generated graph, the performance highly depends on its graph shift matrix, which is created from data features. Because different features have different importance, when some features cannot clearly show the similarities among all nodes or may even mislead the classifier, the features should not be treated equally. Therefore, some label propagation algorithms that do not use $\mathrm{DSP_G}$ integrate multiple graphs to improve their performance [17]–[20]. However, these methods are still based on the conventional label propagation techniques. For the first time in the literature, we adopt multiple graph shift matrices in graph filter design corresponding to different features. The proposed design does not require any complicated pre-processing to determine the importance of each feature. Instead, such importance can be determined by optimizing the weights of all graph shift matrices. Through an alternating minimization process, we can obtain the filter taps and graph shift weights for our well-trained filter.

The rest of the paper is organized as follows. In Section II, we briefly introduce the mathematical background of $\mathrm{DSP_G}$ and give a specific problem statement. In Section III, we develop our filter design approach. To compare our algorithm with the conventional graph filter, we adopt both real and synthetic datasets and design MATLAB simulation experiments to observe their performance. The simulation results is shown in Section IV. We draw conclusions in Section V.

II. Problem Statement

We consider semi-supervised graph data classification by designing a graph filter. Let $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^T$ be a real $N \times D$ data matrix, which collects data on N nodes with D kinds of features. Here, $\mathbf{x}_i$ represents a $D \times 1$ column vector that collects D features of the ith node. As a classification problem, we want to use initially labeled nodes $\{1, \cdots, N_1\}$ for $N_1 < N$ to classify the remaining $N - N_1$ nodes into K categories. The labeled nodes and the remaining unlabeled nodes together constitute an $N \times K$ initial label matrix $\mathbf{S}$. For example, if the ith node is labeled and belongs to the jth category, then $\mathbf{S}_{i,j} = 1$ while remaining elements in the jth row are zeros. When the ith node is unlabeled, all elements in the ith row of $\mathbf{S}$ are zeros. In other words, $\mathbf{S}_{i,j} = 0$ whenever $i > N_1$.

A graph shift operator [6] is given by an $N \times N$ matrix $\mathbf{A}$ which is a generalization of the time-shift z^{-1} in traditional DSP. Similar to FIR filters in DSP, an Lth order shift-invariant graph filter is defined as

$$\mathbf{H} = h_1\mathbf{A} + h_2\mathbf{A}^2 + \cdots + h_L\mathbf{A}^L, \tag{1}$$

where h_i are scalar graph filter coefficients. To solve the classification problem described earlier, a graph filter can be

used and the $N \times K$ classified label matrix $\mathbf{S}^{\mathtt{cla}}$ can be obtained by,

$$\mathbf{S}^{\mathtt{cla}} = \mathcal{Q}(\mathbf{S}^{\mathtt{fil}}) = \mathcal{Q}(\mathbf{HS}), \tag{2}$$

where $\mathbf{S}^{\mathtt{fil}}$ is the filtered label matrix and $\mathcal{Q}(\cdot)$ is a quantization operator which selects the most likely class. For data classification, although the graph filter method is based on label propagation, two major distinctions generally differentiate the $\mathrm{DSP_G}$ approach from label propagation approaches [5], [6]. First, $\mathrm{DSP_G}$ approach only requires $\mathbf{A}$ to be nonnegative and indicate similarities among nodes while label propagation algorithms needs it to be a stochastic matrix. Second, the graph filter is constructed by a matrix polynomial instead of propagating node labels in a Markov chain. In [5], [6], the $N \times N$ graph shift matrix is chosen as

$$A_{i,j} = \frac{\mathtt{exp}\left(-\frac{\rho(\mathbf{x}_i,\mathbf{x}_j)}{\sigma}\right)}{\sum_{i=1}^{N}\mathtt{exp}\left(-\frac{\rho(\mathbf{x}_i,\mathbf{x}_j)}{\sigma}\right)}, \tag{3}$$

where ρ represents Euclidean distance, σ is a scaling coefficient and $\mathbf{x}_i, \mathbf{x}_j$ are ith and jth rows of $\mathbf{X}$. The choice of $\mathbf{A}$ will highly influence the performance of the graph filter.

The numerator in equation (3), which considers all features together, captures the similarity between every pair of nodes. In many practical applications, the information provided by different features are not the same. To avoid demanding prior knowledge and complicated preprocessing to evaluate the importance of all features, we address a novel graph filtering method with multiple graph shift matrices. Similar to the conventional graph filter, we only require the graph shift matrices to be nonnegative instead of being stochastic. We keep the matrix polynomial structure.

Under the same assumptions of conventional graph filter design, we can generate a series of graph matrices $\mathbf{A}(d)$ where $d \in \{1, 2, \cdots, D\}$ corresponds to different features instead of considering all features in one graph in (3). For any graph shift matrix $\mathbf{A}(d)$, the element in the ith row and jth column is

$$A(d)_{i,j} = \frac{\mathtt{exp}\left(-\frac{(X_{i,d}-X_{j,d})^2}{\sigma}\right)}{\sum_{i=1}^{N}\mathtt{exp}\left(-\frac{(X_{i,d}-X_{j,d})^2}{\sigma}\right)}, \tag{4}$$

where $X_{i,d}$ and $X_{j,d}$ are elements of $\mathbf{X}$ correspond to the indices of rows and columns. A weighted multiple-graph-shift filter is proposed as

$$\mathbf{H} = \sum_{d=1}^{D}\sum_{l=1}^{L} w_d h_l \mathbf{A}(d)^l, \tag{5}$$

where the real vectors $\mathbf{h} = [h_1, h_2, \cdots, h_L]^T$ contains graph filter taps and $\mathbf{w} = [w_1, w_2, \cdots, w_D]^T$ contains weight parameters corresponding to graph shift matrices. Since we introduce multiple graph shift matrices corresponding to different features, we gain more flexibility in filter designing by adjusting $\mathbf{w}$. This has benefits, especially when some groups of features are not strongly positively related to the expected classification results.

Similar to the optimization problem introduced in [6], a loss function is created to optimize the L filter taps and D graph weights,

$$\begin{aligned}\arg\min_{\mathbf{h},\mathbf{w}} \quad \mathcal{L} &= \|\mathbf{RHS} - \mathbf{S}\|_F + \alpha\|\mathbf{h}\|_2 + \beta\|\mathbf{w}\|_2 \\ &= \left\|\mathbf{R}\left(\sum_{d=1}^{D}\sum_{l=1}^{L} w_d h_l \mathbf{A}(d)^l\right)\mathbf{S} - \mathbf{S}\right\|_F \\ &\quad + \alpha\|\mathbf{h}\|_2 + \beta\|\mathbf{w}\|_2\end{aligned} \tag{6}$$

where $\|\cdot\|_F$ represents Frobenius norm and $\mathbf{R} = \mathtt{diag}(\mathbf{r})$ is an $N \times N$ diagonal matrix. The first N_1 elements in the $N \times 1$ vector $\mathbf{r}$ equal to 1 and the remaining $N - N_1$ elements equal to 0. Since the unlabeled nodes are initially assigned with zero elements in $\mathbf{S}$ but we expect each row in $\mathbf{S}^{\mathtt{fil}}$ has at least one nonnegative value, naively optimizing the corresponding elements in $\mathbf{S}^{\mathtt{fil}}$ to zeros will mislead the classification. Therefore, we only utilize the labeled part in $\mathbf{S}$ as a reference. We introduce $\mathbf{R}$ to select the corresponding nodes from the filtered label matrix. The first term in (6) is trying to enhance the influences among similar nodes and weakens the influences among dissimilar nodes by retaining the initial label values.The second and third terms in (6) are regularizations for $\mathbf{h}$ and $\mathbf{w}$ to avoid overfitting, while α and β are coefficients to balance the weights of all terms. Considering the similarities among all labeled and unlabeled nodes influence each other through matrix polynomials, the classification problem is still semi-supervised even after the selection process $\mathbf{R}$.

After all optimized filter parameters are obtained, the graph filter is well-trained. Then by quantizing the filtered label $\mathbf{S}^{\mathtt{fil}}$ as shown in (2), we can obtain the classified labels for all nodes. Here, we adopt an operator $\mathcal{Q}(\cdot)$ quantizes the largest element in each row to be 1 and remaining elements in the row to be 0.

III. Algorithm Description

The loss function $\mathcal{L}$ in (6) has a bilinear term and is nonconvex. We implement an alternating minimization method to optimize this nonconvex problem, which is simple to implement. Although it is usually hard to guarantee a global optimum, alternating minimization can always convergence except for few special circumstances, which makes it a very practical and competitive method for some nonconvex problems [21].

Vectors $\mathbf{h}$ and $\mathbf{w}$ are initialized by

$$\begin{gathered}h_l = \frac{1}{L}, w_d = \frac{1}{D} \\ \forall l \in \{1, 2, \cdots, L\}, \forall d \in \{1, 2, \cdots, D\}.\end{gathered} \tag{7}$$

In the $(m+1)^{\mathtt{th}}$ iteration, we first assume $\mathbf{w}$ is fixed, which means we adopt value $\mathbf{w}^{(m)}$ from the mth iteration. Then we optimize $\mathbf{h}$ by transforming $\mathcal{L}$ into a convex loss function $\mathcal{L}_h$,

$$\begin{aligned}\arg\min_{\mathbf{h}} \quad \mathcal{L}_h^{(m+1)} &= \|\mathbf{RHS} - \mathbf{S}\|_F + \alpha\|\mathbf{h}\|_2 \\ &= \left\|\mathbf{R}'\mathbf{Q}_h(\mathbf{w}^{(m)})\mathbf{h} - \mathbf{S}'\right\|_2 \\ &\quad + \alpha\|\mathbf{h}\|_2\end{aligned} \tag{8}$$

where $\mathbf{Q}_h(\mathbf{w}^{(m)})$ is an $NK \times L$ matrix that is used to optimize $\mathbf{h}$ contains the information of all graph shift matrices $\mathbf{A}(1), \cdots, \mathbf{A}(D)$ and relies on the current weights $\mathbf{w}^{(m)}$. It should be noted that $\mathbf{R}' = \mathtt{diag}(\mathbf{r}')$ is an $NK \times NK$ diagonal matrix and the $NK \times 1$ vector $\mathbf{r}' = \left[\mathbf{r}^T, \mathbf{r}^T, \cdots, \mathbf{r}^T\right]^T$. To demonstrate the generating process of $\mathbf{Q}_h(\mathbf{w}^{(m)})$, we first introduce a set of $N \times L$ matrices $\{\mathbf{B}_1, \cdots, \mathbf{B}_K\}$, which gives us $\mathbf{Q}_h(\mathbf{w}^{(m)}) = \left[\mathbf{B}_1^T, \cdots, \mathbf{B}_K^T\right]^T$. Then we assume

$$\begin{aligned} \mathbf{B}_k &= [\mathbf{c}_1(k), \cdots, \mathbf{c}_L(k)] \\ &\forall k \in \{1, 2, \cdots, K\}, \end{aligned} \tag{9}$$

where $\{\mathbf{c}_1(k), \cdots, \mathbf{c}_L(k)\}$ is a set of $N \times 1$ vectors that can be obtained by

$$\begin{aligned} \mathbf{c}_l(k) &= \sum_{d=1}^{D} w_d^{(m)} \mathbf{A}(d)^l \mathbf{s}_k \\ &\forall l \in \{1, 2, \cdots, L\}. \end{aligned} \tag{10}$$

Here $\mathbf{s}_k$ is an $N \times 1$ label vector that satisfies $\mathbf{S} = [\mathbf{s}_1, \cdots, \mathbf{s}_k, \cdots, \mathbf{s}_K]$ and the $N_2 \times 1$ transformed label vector $\mathbf{S}'$ is formed as

$$\mathbf{S}' = \left[\mathbf{s}_1^T, \mathbf{s}_2^T, \cdots, \mathbf{s}_K^T\right]^T. \tag{11}$$

Similarly, once we got the optimized value $\mathbf{h}^{(m+1)}$, we assume $\mathbf{h}$ to be fixed and start optimizing for $\mathbf{w}^{(m+1)}$ by

$$\begin{aligned} \arg\min_{\mathbf{w}} \quad \mathcal{L}_w^{(m+1)} &= \left\| \mathbf{R}' \mathbf{Q}_w(\mathbf{h}^{(m+1)}) \mathbf{w} - \mathbf{S}' \right\|_2 \\ &+ \beta \left\| \mathbf{w} \right\|_2 \end{aligned} \tag{12}$$

where $\mathbf{Q}_w(\mathbf{h}^{(m+1)})$ is an $NK \times D$ matrix that is used to optimize $\mathbf{w}$ using the current $\mathbf{h}$ vector. It can be represented by $\mathbf{Q}_w(\mathbf{h}^{(m+1)}) = \left[\mathbf{E}_1^T, \cdots, \mathbf{E}_K^T\right]^T$. Here $\{\mathbf{E}_1, \cdots, \mathbf{E}_K\}$ is a set of $N \times D$ matrices and it satisfies that

$$\begin{aligned} \mathbf{E}_k &= [\mathbf{f}_1(k), \cdots, \mathbf{f}_D(k)] \\ &\forall k \in \{1, 2, \cdots, K\}, \end{aligned} \tag{13}$$

where $\{\mathbf{f}_1(k), \cdots, \mathbf{f}_D(k)\}$ is a set of $N \times 1$ vectors that can be obtained by

$$\begin{aligned} \mathbf{f}_d(k) &= \sum_{l=1}^{L} h_l^{(m+1)} \mathbf{A}(d)^l \mathbf{s}_k \\ &\forall d \in \{1, 2, \cdots, D\}. \end{aligned} \tag{14}$$

We alternate these two convex minimization processes $\mathcal{L}_h$ and $\mathcal{L}_w$. When $\left|\mathcal{L}^{(m+1)} - \mathcal{L}^{(m)}\right| < \upsilon$, where υ is a selected small value, we recognize the loss function $\mathcal{L}$ converged. The pseudo-code of this algorithm is shown as $\mathbf{Algorithm\,1}$.

IV. Simulation Results

In this section, we exhibit the simulation experiment results based on both synthetic and real datasets. We implement the conventional shift-invariant graph filter and our proposed graph filter with multiple weighted graph shifts on data classification problems through Monte Carlo tests.

Algorithm 1 Graph Filter with Multiple Graph Shift Matrices

Input: feature matrix $\mathbf{X}$, initial label matrix $\mathbf{S}$, number of filter taps L

Output: classification result $\mathbf{S}^{\mathtt{cla}}$

1: Generate all graph shift matrices $\mathbf{A}(d), d \in \{1, 2, \cdots, D\}$ through $\mathbf{S}$.
2: **while** $\left|\mathcal{L}^{(m-1)} - \mathcal{L}^{(m-2)}\right| > \upsilon$ **do**
3: $\quad \mathbf{h}^{(m)} \leftarrow \arg\min_{\mathbf{h}} \left\| \mathbf{R}' \mathbf{Q}_h(\mathbf{w}^{(m-1)}) \mathbf{h} - \mathbf{S}' \right\|_2 + \alpha \left\| \mathbf{h} \right\|_2$
4: $\quad \mathbf{w}^{(m)} \leftarrow \arg\min_{\mathbf{w}} \left\| \mathbf{R}' \mathbf{Q}_w(\mathbf{h}^{(m)}) \mathbf{w} - \mathbf{S}' \right\|_2 + \beta \left\| \mathbf{w} \right\|_2$
5: $\quad \mathbf{H}^{(m)} = \sum_{d=1}^{D} \sum_{l=1}^{L} w_d^{(m)} h_l^{(m)} \mathbf{A}(d)^l$
6: $\quad \mathcal{L}^{(m)} = \left\| \mathbf{R} \mathbf{H}^{(m)} \mathbf{S} - \mathbf{S} \right\|_F$
7: **end while**
8: Obtain classification result $\mathbf{S}^{\mathtt{cla}} = \mathcal{Q}(\mathbf{S}^{\mathtt{fil}}) = \mathcal{Q}(\mathbf{HS})$

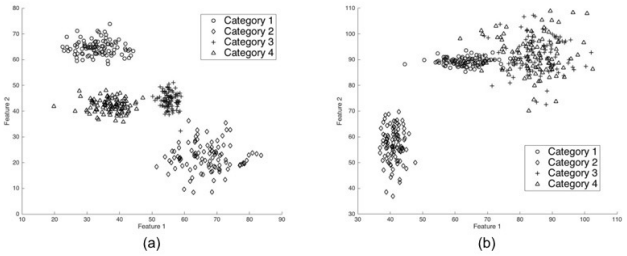

Fig. 1. (a) A sample of normal features with four categories and (b) a sample of overlapping features with four categories.

A. Synthetic Dataset

In many conventional machine learning studies, researchers test their classifier with various synthetic dataset to evaluate their performance. It has been discussed in [22] that some characters such as overlapping, noise and inherent complexity will bring more difficulties in classification for many data mining algorithms. To study the filter performance with different feature qualities, we experiment both graph filters by considering five kinds of features: normal, overlapping, spiral, half ellipse and noise. As shown in Figure 1 (a), we illustrate four categories containing normal features in a two-dimensional domain. The features belonging to each category on each dimension is generated by a Gaussian distribution with randomly selected means and variances. For overlapping features shown in Figure 1 (b), there are still four categories with Gaussian generated features but two of them are strongly overlapping. The spiral and half ellipse exhibits in Figure 2 (a) and (b) also have four categories with randomly selected parameters. The noise features, which is not illustrated in figure, are randomly generated for all categories through one uniform distribution with randomly selected parameters.

In our MATLAB simulation experiments, we corroborate normal features with the other four features respectively. By doing this, we create a synthetic dataset that have different qualities of features. Since the conventional graph filter relies heavily on that graph shift matrix should correctly represent the similarities among all nodes, the different qualities of

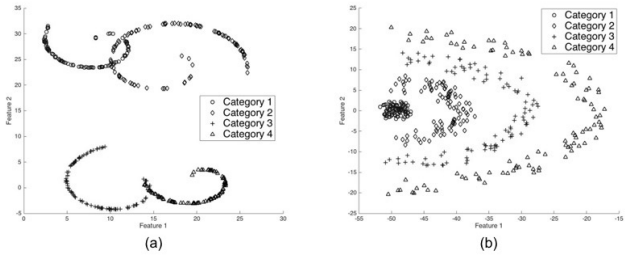

Fig. 2. (a) A sample of spiral features with four categories and (b) a sample of half ellipse features with four categories.

features will be a meaningful challenge to building an effective graph filter. Specifically, in each Monte Carlo simulation, we first generate and standardize a graph dataset with randomly selected feature generation parameters who contains half normal features and half another kind of features from the remaining four. The synthetic dataset is fully labeled and uniformly classified into four categories with $N = 400$ and $D = 20$. Then we randomly select 10% nodes from each categories as known nodes and treat the remaining 90% as unknown nodes. With different qualities of features and different number of labeled nodes, we are able to analyze how feature quality influence the performance of the graph filters.

Within rounds of Monte Carlo simulations, we test the performance of both conventional and proposed graph filters with different combinations of features. The average error rates of two graph filters are recorded in the Table I. We compare their performances by calculating the ratio between error rate of conventional graph filter and error rate of proposed graph filter. It shows that with the decreasing of SNR, our proposed graph filter provides smaller error rate.

TABLE I
ERROR RATE OF GRAPH-BASED CLASSIFICATION WITH SYNTHETIC DATA

Feature Combination	Average Classification Error Rate	
	Conventional	Proposed
Normal & Overlapping	0.0000	0.0000
Normal & Spiral	0.0056	0.0045
Normal & Half Ellipse	0.1796	0.0016
Normal & Noise	0.7962	0.0009

From the first two groups of error rates we can see that the graph-based classifier works well on both linearly separable and inseparable features. However, as mentioned in Section I, the graph shift matrix highly influences the performance of conventional graph filter. As shown by the simulation result of "Normal & Half Ellipse" features, the average error rate of conventional graph filter goes up since the corresponding graph shift matrices no longer represents the similarity information correctly. When we select the "Normal & Noise" feature combination, which is more chaotic, the conventional graph filter fails in the classification.

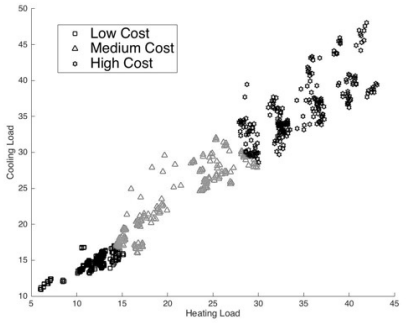

Fig. 3. The building energy load dataset with three categories: low cost, medium cost, high cost.

B. Real Dataset

The real dataset we adopt is called energy efficiency dataset [23]. This dataset collects 8 shape-related features for 768 buildings, for example surface area, with two real value responses, hearing load and cooling load. As shown in Figure 3, we uniformly cluster the data into 3 categories: low load, median load and high load corresponding to these two responses.

Similar to the simulation experiments on the synthetic dataset, for the energy efficiency dataset, we still compute the average error rate as the filter performance criterion. In each Monte Carlo test, we randomly select a number of nodes as labeled corresponds to a certain labeling ratio and record the performance for two kinds of filters. Through Monte Carlo tests, the simulation result is shown in Table II. As we can see that with comparatively low labeling ratio, our proposed algorithm can provide significant advantage in classification error rate.

TABLE II
ERROR RATE OF GRAPH-BASED CLASSIFICATION WITH REAL DATA

Labeling Ratio	Average Classification Error Rate	
	Conventional	Proposed
0.05	0.4688	0.2513
0.1	0.3066	0.2005
0.15	0.2663	0.1901
0.2	0.1953	0.1947

V. CONCLUSIONS

In this paper, we discussed the problem of using graph filter to do data classification. We propose a novel graph filter designing method, which considers multiple graph shift matrices. Comparing to the conventional graph filter that adopts only one graph shift matrix, our method is more flexible and robust when we have features with various qualities and can be efficiently solved by alternating minimization. We use simulations on both synthetic data and real data to show that the proposed approach can provide lower error rate than the conventional one when the feature qualities are poor.

REFERENCES

[1] S. Zhang, C. Tepedelenlioğlu, M. K. Banavar, and A. Spanias, "Distributed node counting in wireless sensor networks in the presence of communication noise," *IEEE Sensors Journal*, vol. 17, no. 4, pp. 1175–1186, 2017.

[2] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems*, 2016, pp. 3844–3852.

[3] 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," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[4] H. Song, J. J. Thiagarajan, P. Sattigeri, and A. Spanias, "Optimizing kernel machines using deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 11, pp. 5528–5540, 2018.

[5] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE transactions on signal processing*, vol. 61, no. 7, pp. 1644–1656, 2013.

[6] ——, "Discrete signal processing on graphs: Graph filters." in *ICASSP*, 2013, pp. 6163–6166.

[7] S. Chen, A. Sandryhaila, J. M. Moura, and J. Kovacevic, "Adaptive graph filtering: Multiresolution classification on graphs," in *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. IEEE, 2013, pp. 427–430.

[8] S. Chen, F. Cerda, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovačević, "Semi-supervised multiresolution classification using adaptive graph filtering with application to indirect bridge structural health monitoring," *IEEE Transactions on Signal Processing*, vol. 62, no. 11, pp. 2879–2893, 2014.

[9] D. Thanou, D. I. Shuman, and P. Frossard, "Parametric dictionary learning for graph signals," in *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. IEEE, 2013, pp. 487–490.

[10] S. Chen, A. Sandryhaila, G. Lederman, Z. Wang, J. M. Moura, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovacevic, "Signal inpainting on graphs via total variation minimization," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 8267–8271.

[11] S. Chen, A. Sandryhaila, J. M. Moura, and J. Kovacevic, "Signal denoising on graphs via graph filtering," in *Signal and Information Processing (GlobalSIP), 2014 IEEE Global Conference on*. IEEE, 2014, pp. 872–876.

[12] U. S. Shanthamallu, J. J. Thiagarajan, and A. Spanias, "Improving robustness of attention models on graphs," *arXiv preprint arXiv:1811.00181*, 2018.

[13] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," 2001.

[14] M. Szummer and T. Jaakkola, "Partially labeled classification with Markov random walks," in *Advances in neural information processing systems*, 2002, pp. 945–952.

[15] X. Zhu, J. Lafferty, and Z. Ghahramani, "Semi-supervised learning: From gaussian fields to gaussian processes," 2003.

[16] Y. Bengio, O. Delalleau, and N. Le Roux, "11 label propagation and quadratic criterion," 2006.

[17] M. Karasuyama and H. Mamitsuka, "Multiple graph label propagation by sparse integration," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 12, pp. 1999–2012, 2013.

[18] K. Tsuda, H. Shin, and B. Schölkopf, "Fast protein classification with multiple networks," *Bioinformatics*, vol. 21, no. suppl_2, pp. ii59–ii65, 2005.

[19] M. Wang, X.-S. Hua, R. Hong, J. Tang, G.-J. Qi, and Y. Song, "Unified video annotation via multigraph learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 5, pp. 733–746, 2009.

[20] T. Kato, H. Kashima, and M. Sugiyama, "Robust label propagation on multiple networks," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 35–44, 2009.

[21] M. Udell, C. Horn, R. Zadeh, S. Boyd *et al.*, "Generalized low rank models," *Foundations and Trends® in Machine Learning*, vol. 9, no. 1, pp. 1–118, 2016.

[22] P. D. Scott and E. Wilkins, "Evaluating data mining procedures: techniques for generating artificial data sets," *Information and software technology*, vol. 41, no. 9, pp. 579–587, 1999.

[23] A. Tsanas and A. Xifara, "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools," *Energy and Buildings*, vol. 49, pp. 560–567, 2012.