MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Graph Signaling Denoising via Unrolling Networks

Chen, Siheng; Eldar, Yonina

TR2021-071 June 08, 2021

Abstract

We propose an interpretable graph neural network framework to denoise single or multiple noisy graph signals. The proposed graph unrolling networks expand algorithm unrolling to the graph domain and provide an interpretation of the architecture design from a signal processing perspective. We unroll an iterative denoising algorithm by mapping each iteration into a single network layer where the feed-forward process is equivalent to iteratively denoising graph signals. We train the graph unrolling networks through unsupervised learning, where the input noisy graph signals are used to supervise the networks. By leveraging the learning ability of neural networks, we adaptively capture appropriate priors from input noisy graph signals, instead of manually choosing signal priors. To validate the proposed methods, we conduct extensive experiments on both real-world datasets and simulated datasets, and demonstrate that our methods have smaller denoising errors than conventional denoising algorithms and state-of-the-art graph neural networks. For denoising a single smooth graph signal, the normalized mean square error of the proposed networks is around 40% and 60% lower than that of graph Laplacian denoising and graph wavelets, respectively

IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

GRAPH SIGNAL DENOISING VIA UNROLLING NETWORKS

Siheng Chen[1] *, Yonina C. Eldar*[2]

[1] Shanghai Jiao Tong University; [2] Weizmann Institute of Science

ABSTRACT

We propose an interpretable graph neural network framework to denoise single or multiple noisy graph signals. The proposed *graph unrolling networks* expand algorithm unrolling to the graph domain and provide an interpretation of the architecture design from a signal processing perspective. We unroll an iterative denoising algorithm by mapping each iteration into a single network layer where the feed-forward process is equivalent to iteratively denoising graph signals. We train the graph unrolling networks through unsupervised learning, where the input noisy graph signals are used to supervise the networks. By leveraging the learning ability of neural networks, we adaptively capture appropriate priors from input noisy graph signals, instead of manually choosing signal priors. To validate the proposed methods, we conduct extensive experiments on both real-world datasets and simulated datasets, and demonstrate that our methods have smaller denoising errors than conventional denoising algorithms and state-of-the-art graph neural networks. For denoising a single smooth graph signal, the normalized mean square error of the proposed networks is around 40% and 60% lower than that of graph Laplacian denoising and graph wavelets, respectively.

Index Terms— Graph signal denoising, algorithm unrolling, graph neural networks

1. INTRODUCTION

Data today is often generated from a diverse sources, including social, citation, biological, and physical infrastructure [1]. Unlike time-series signals or images, such signals possess complex and irregular structures, which can be modeled as graphs. Analyzing graph signals requires dealing with the underlying irregular relationships. Graph signal processing generalizes the classical signal processing toolbox to the graph domain and provides a series of techniques to process graph signals [1]. Graph neural networks provide a powerful framework to learn from graph signals with graphs as induced biases [2]. Permeating the benefits of deep learning to the graph domain, graph convolutional networks and variants have attained remarkable success in social network analysis [3] and computer vision [4].

In this work, we consider denoising graph signals [5]. In classical signal processing, signal denoising is one of the most ubiquitous tasks. To handle graph signals, there are two mainstream approaches: graph-regularization-based optimization and graph dictionary design. The optimization approach usually introduces a graph-regularization term that promotes certain properties of the graph signal and solves a regularized optimization problem to obtain a denoised solution [5, 6]. In comparison, the graph-dictionary approach aims to reconstruct graph signals through a predesigned graph dictionary, such as graph wavelets [7], and graph frames [8]. These dictionaries are essentially variants of graph filters. A reconstructed graph signal consists of a sparse combination of elementary graph signals in a graph dictionary.

A fundamental challenge for both denoising approaches is that we may not know an appropriate prior on the noiseless graph signals in practice. It is then hard to either choose an appropriate graph-regularization term or design an appropriate graph dictionary. Furthermore, some graph priors are too complicated to be precisely described in mathematical terms or may lead to computationally intensive algorithms.

To solve this issue, it is desirable to learn an appropriate prior from given graph signals; in other words, the denoising algorithm should have sufficient feasibility to learn from and adapt to arbitrary signal priors. In this work, we leverage the powerful learning ability of graph neural networks and combine them with interpretablity based on a signal processing perspective. Furthermore, most graph neural networks are developed for supervised-learning tasks, such node classification and graph classification [3]. Those tasks require a large number of ground-truth labels, which is expensive to obtain. Here we consider an unsupervised-learning setting, where the networks have to learn from a few noisy graph signals and the ground-truth noiseless graph signals are unknown. Through unsupervised learning, we demonstrate the generalization ability of the proposed graph neural networks.

Our goal is to develop a graph network denoising framework by combining advantages of both conventional graph signal denoising algorithms and graph neural networks. Here we expand algorithm unrolling to the graph domain [9]. We first propose a general iterative algorithm for graph signal denoising and then transform it to a graph neural network through algorithm unrolling, where each iteration is mapped to a network layer. Compared to conventional denoising al-

The work was done when Siheng Chen was working at Mitsubishi Electric Research Laboratories.

gorithms [5], the proposed graph unrolling network can learn a variety of priors from given graph signals by leveraging deep neural networks. Compared to many other graph neural networks [3], the proposed graph unrolling network is interpretable by following analytical iterative steps. To train graph unrolling networks, we use single or multiple noisy graph signals and minimize the difference between the original input and the network output; in other words, the input noisy measurements are used to supervise the neural network training.

To validate the empirical performance of the proposed method, we conduct a series of experiments on both simulated datasets and real-world datasets with mixture Gaussian and Laplace noises. We find that graph unrolling networks consistently achieve better denoising performances than conventional graph signal denoising algorithms and state-of-the-art graph neural networks on various types of graph signals and noise models. Even for denoising a single smooth graph signal, the proposed graph unrolling networks are around 40% and 60% better than graph Laplacian denoising [10] and graph wavelets [7], respectively. This demonstrates that the unrolling approach allows to obtain improved results over existing methods even using a single training point.

The main contributions of this work include:

• We propose interpretable graph unrolling networks by unrolling a general iterative algorithm for graph signal denoising in an unsupervised-learning setting; and

• We conduct experiments on both simulated and real-world data to validate that the proposed denoising methods outperform both conventional graph signal denoising methods and state-of-the-art graph neural networks on various types of graph signals and noise models.

2. DENOISING NETWORKS VIA UNROLLING

2.1. Problem Formulation

We first mathematically formulate the task of graph signal denoising. We consider a graph $G = (\mathcal{V}, \mathcal{E}, \mathrm{A})$, where $\mathcal{V} = \{v_n\}_{n=1}^N$ is the set of *vertices*, $\mathcal{E} = \{e_m\}_{m=1}^M$ is the set of undirected *edges*, and $\mathrm{A} \in \mathbb{R}^{N \times N}$ is the graph adjacency matrix, representing connections between vertices. The weight $\mathrm{A}_{i,j}$ of an edge from the ith to the jth vertex characterizes the relation, such as similarity or dependency, between the corresponding signal values. Using the graph representation G, a *graph signal* is defined as a map that assigns a signal coefficient $x_n \in \mathbb{R}$ to the vertex v_n. A graph signal can be written as a length-N vector defined by $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_N \end{bmatrix}^T$, where the nth vector element x_n is indexed by the vertex v_n.

Assume that we are given a length-N noisy measurement $\mathbf{t} = \mathbf{x} + \mathbf{e}$, where $\mathbf{x}$ is the noiseless graph signal and $\mathbf{e}$ is noise. The goal of graph signal denoising is to recover $\mathbf{x}$ from $\mathbf{t}$ by removing the noise.

Without any prior information on the noiseless graph signals, it is impossible to split noises from the measurements. Possible priors include sparsity, graph smoothness and graph piecewise-smoothness [11]. Here we consider a general graph signal model in which the graph signal is generated through graph filtering over vertices; that is, $\mathbf{x} = \mathbf{h} *_v \mathbf{s} = \sum_{\ell=1}^L h_\ell \, \mathrm{A}^\ell \, \mathbf{s}$, where $\mathbf{s} \in \mathbb{R}^N$ is a base graph signal, which may not have any graph-related properties, $*_v$ indicates a convolution on the graph vertex domain and $\mathbf{h} = \begin{bmatrix} h_1 & h_2 & \cdots & h_L \end{bmatrix}^T \in \mathbb{R}^L$ are the filter coefficients with L the filter length. Here we consider a typical design of a graph filter, which is a polynomial of the graph shift. The graph filtering process modifies a given base graph signal according to certain patterns of the graph and explicitly regularizes the output.

Based on this graph signal model, we can remove noises by solving the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{s} \in \mathbb{R}^N} \frac{1}{2} \|\mathbf{t} - \mathbf{x}\|_2^2 + u(\mathrm{P}\,\mathbf{x}) + r(\mathrm{Q}\,\mathbf{s}), \\ &\text{subject to } \mathbf{x} = \mathbf{h} *_v \mathbf{s}, \end{aligned} \tag{1}$$

where $u(\cdot), r(\cdot) \in \mathbb{R}$ are additional regularization terms on $\mathbf{s}$ and $\mathbf{x}$ respectively and P and Q are two matrices. The denoised graph signal is then given by $h *_v \mathbf{s}$. It is straightforward to show that graph Laplacian denoising, graph sparse coding and graph trend filtering are special cases of (1).

2.2. Algorithm Unrolling

Consider a general iterative algorithm to solve the graph signal denoising problem (1) based on the half-quadratic splitting algorithm. The basic idea is to perform variable-splitting and then alternating minimization on the penalty function [12]. Introduce two auxiliary variables $\mathbf{y} = \mathrm{P}\,\mathbf{x}$ and $\mathbf{z} = \mathrm{Q}\,\mathbf{s}$ and then reformulate (1) as

$$\begin{aligned} &\min_{\mathbf{s} \in \mathbb{R}^N} \frac{1}{2} \|\mathbf{t} - \mathbf{x}\|_2^2 + u(\mathbf{y}) + r(\mathbf{z}), \\ &\text{subject to } \mathbf{x} = \mathbf{h} *_v \mathbf{s}, \;\; \mathbf{y} = \mathrm{P}\,\mathbf{x}, \;\; \mathbf{z} = \mathrm{Q}\,\mathbf{s}. \end{aligned}$$

The penalty function is $J(\mathbf{x}, \mathbf{s}, \mathbf{y}, \mathbf{z}) = \|\mathbf{t} - \mathbf{x}\|_2^2/2 + u(\mathbf{y}) + r(\mathbf{z}) + \mu_1 \|\mathbf{x} - \mathbf{h} *_v \mathbf{s}\|_2^2/2 + \mu_2 \|\mathbf{y} - \mathrm{P}\,\mathbf{x}\|_2^2/2 + \mu_3 \|\mathbf{z} - \mathrm{Q}\,\mathbf{s}\|_2^2/2$, where μ_1, μ_2, μ_3 are appropriate step sizes. We alternately minimize J over $\mathbf{x}, \mathbf{s}, \mathbf{y}, \mathbf{z}$, leading to the following updates:

$$\mathbf{x} \leftarrow \widetilde{\mathrm{P}} \Big(\mu_1 \mathbf{h} *_v \mathbf{s} + \mathbf{t} + \mu_2 \, \mathrm{P}^T \mathbf{y} \Big), \tag{2a}$$

$$\mathbf{s} \leftarrow \widetilde{\mathrm{Q}} \Big(\mu_1 \mathbf{h} *_v^T \mathbf{x} + \mu_3 \, \mathrm{Q}^T \mathbf{z} \Big), \tag{2b}$$

$$\mathbf{y} \leftarrow \arg\min_{\mathbf{y}} \frac{\mu_2}{2} \|\mathbf{y} - \mathrm{P}\,\mathbf{x}\|_2^2 + u(\mathbf{y}), \tag{2c}$$

$$\mathbf{z} \leftarrow \arg\min_{\mathbf{z}} \frac{\mu_3}{2} \|\mathbf{z} - \mathrm{Q}\,\mathbf{s}\|_2^2 + r(\mathbf{z}), \tag{2d}$$

where

$$\widetilde{\mathrm{P}} = (\mathrm{I} + \mu_1 \, \mathrm{I} + \mu_2 \, \mathrm{P}^T \, \mathrm{P})^{-1}$$

and

$$\widetilde{\mathrm{Q}} = (\mu_1 \sum_{\ell'=1}^{L} h_{\ell'} \mathrm{A}^{\ell'} \sum_{\ell=1}^{L} h_{\ell} \mathrm{A}^{\ell} + \mu_3 \mathrm{Q}^T \mathrm{Q})^{-1}.$$

Intuitively, (2a) denoises by merging information from the original measurements $\mathbf{t}$, filtered signals $\mathbf{h} *_v \mathbf{s}$ and the auxiliary variable $\mathbf{y}$; (2b) generates a base graph signal through graph deconvolution; and (2c) and (2d) solve two proximal functions with regularization $u(\cdot)$ and $r(\cdot)$, respectively.

To unroll the iteration steps (2), we consider two major substitutions. First, we replace the fixed graph convolution by the graph convolution with trainable filter coefficients. Second, we replace the sub-optimization problems in (2c) and (2d) by a trainable neural network.

The bth unrolling layer for denoising a graph signal is,

$$\mathbf{x}^{(b+1)} \leftarrow \mathbb{A} *_a \mathbf{s}^{(b)} + \mathbb{B} *_a \mathbf{t}^{(b)} + \mathbb{C} *_a \left(\mathrm{P}^T \mathbf{y}^{(b)}\right), \quad (3a)$$

$$\mathbf{s}^{(b+1)} \leftarrow \mathbb{D} *_a \mathbf{x}^{(b+1)} + \mathbb{E} *_a \left(\mathrm{Q}^T \mathbf{z}^{(b)}\right), \quad (3b)$$

$$\mathbf{y}^{(b+1)} \leftarrow \mathrm{NN}_u \left(\mathrm{P}\, \mathbf{x}^{(b+1)}\right), \quad (3c)$$

$$\mathbf{z}^{(b+1)} \leftarrow \mathrm{NN}_r \left(\mathrm{Q}\, \mathbf{s}^{(b+1)}\right), \quad (3d)$$

where $\mathbb{A}*_a$, $\mathbb{B}*_a$, $\mathbb{C}*_a$, $\mathbb{D}*_a$, and $\mathbb{E}*_a$ are individual trainable graph convolutions with filter coefficients that are trainable parameters with subscript a indicating trainable, and $\mathrm{NN}_u(\cdot)$ and $\mathrm{NN}_r(\cdot)$ are two neural networks, which involve trainable parameters. Intuitively, (3a) and (3b) are neural-network implementations of (2a) and (2b), respectively, replacing fixed graph convolutions $\mathbf{h}*_v$ by trainable graph convolutions $\mathbb{A}*_a$; and (3c) and (3d) are neural-network implementations of the proximal functions (2c) and (2d), respectively, using neural networks to solve sub-optimization problems; see similar substitutions in [13, 14, 15]. Instead of following the exact mathematical relationship in (2), we allow trainable operators to adaptively learn from data, usually reducing a lot of computation. The implementations of (3c) and (3d) depend on specific regularization terms, $u(\cdot)$ and $r(\cdot)$. For some $u(\cdot), r(\cdot)$, we might end up with an analytical form for (3c) and (3d).

To build a complete network architecture, we initialize $\mathbf{z}^{(0)}, \mathbf{s}^{(0)}, \mathbf{y}^{(0)}$ to be all-zero matrices and sequentially stack B unrolling layers (3). This is hypothetically equivalent to running the iteration steps (2) for B times. Through optimizing trainable parameters in trainable graph convolutions and two sub-neural-networks, we obtain the denoised output $\widehat{\mathbf{x}} = \mathbf{x}^{(B)}$. Here all the trainable parameters come from two parts, including filter coefficients in each trainable graph convolution and the parameters in the neural networks (3c) and (3d). Through optimizing those parameters, we can capture complicated priors in the original graph signals in a data-driven manner. To train those parameters, we consider the loss function

$$\text{loss} = \|f(\mathbf{t}) - \mathbf{t}\|_2^2 = \|\widehat{\mathbf{x}} - \mathbf{t}\|_2^2, \quad (4)$$

where $\widehat{\mathbf{x}}$ is the output of the proposed network $f(\cdot)$, and $\mathbf{t}$ are the original measurements. We then use the stochastic gradient descent to minimize the loss and optimize this network [16]. The noisy measurement $\mathbf{t}$ is used as both input and supervision of the network. Note that it is straightforward to expand this setting to train with multiple graph signals.

Our algorithm unrolling here is rooted in the half-quadratic splitting algorithm. In practice, our optimization problem can be solved using various alternative iterative algorithms, which may lead to distinct network architectures. *No matter what iterative algorithm is used, the core strategy is to follow the iterative steps and use trainable graph convolution to substitute fixed, yet computationally expensive graph filtering.* We call a network architecture that follows this strategy a *graph unrolling network (GUN)*. Compared to conventional graph signal denoising algorithms [5, 10], the proposed GUN is able to learn a variety of complicated signal priors from given graph signals by leveraging the learning ability of deep neural networks. Compared to many generic graph neural networks [3], the proposed GUN is interpretable by following analytical iterative steps. We unroll an iterative algorithm for solving (1) into a graph neural network by mapping each iteration into a single network layer and stacking multiple layers together. Therefore, the proposed GUN can be naturally interpreted as a parameter optimized algorithm.

3. EXPERIMENTAL RESULTS

We now validate the superiority of the proposed method.

3.1. Dataset

U.S. temperature data. We consider 150 weather stations in the United States that record their local temperatures [17]. Each weather station has 365 days of recordings (one recording per day), for a total of 54,750 measurements. The graph representing these weather stations is obtained by measuring the geodesic distance between each pair of weather stations. The vertices are represented by an 8-nearest neighbor graph, in which vertices represent weather stations, and each stations is connected to the eight closest weather stations. Each graph signal is the daily temperature values recorded in each weather station.

NYC traffic data. We consider the taxi-pickup activity in Manhattan on January 1th, 2014. This is the Manhattan street network with $2,552$ intersections and $3,153$ road segments. We model each intersection as a vertex and each road segment as an edge. We model the taxi-pickup positions as signals supported on the Manhattan street network. We project each taxi-pickup to its nearest intersection, and then count the number of taxi-pickups at each intersection. Each graph signal is the hourly number of taxi-pickups recorded in each intersection.

Cora. We finally consider a citation network dataset, called Cora [3]. The datasets contain sparse bag-of-words feature vectors for each document and a list of citation links

Metric	Temperature NMSE		Traffic NMSE		cora Error Rate		cora F1 score	
Method	1	365	1	24	1	7	1	7
Baseline	0.34	0.377	0.392	0.377	0.095	0.099	0.829	0.695
GLD	0.045	0.024	0.248	0.255	0.055	0.032	0.609	0.793
GTF	0.079	0.036	0.202	0.176	0.06	0.039	0.484	0.532
GFT	0.065	0.053	0.257	0.231	0.079	0.053	0.459	0.489
SGWT	0.069	0.11	0.184	0.162	0.074	0.073	0.551	0.569
QMF	0.26	0.31	**0.18**	0.185	0.087	0.087	0.51	0.512
CSFB	0.07	0.061	0.344	0.36	0.129	0.143	0.43	0.437
MLP	0.142	0.027	0.31	0.169	0.095	0.072	0.829	0.695
GCN	0.041	0.033	0.293	0.279	0.042	0.025	0.903	0.901
GAT	0.044	0.031	0.267	0.264	0.041	0.032	0.909	0.873
GUN	**0.037**	**0.016**	0.324	0.178	**0.04**	**0.024**	**0.91**	**0.906**

Table 1: Denoising error of real-world data with mixture noises (Gaussian and Laplace).

between documents. We treat each citation link as an undirected edge and each document as a class label. The citation network has $2,708$ nodes and $5,429$ edges and 7 class labels. We consider 7 class labels as graph signals. We introduce Bernoulli noises and randomly flip 10% of the binary values.

3.2. Experimental setup

We consider three classes of competitive denoising algorithms: graph-regularized optimizations, graph filter banks and neural networks. For graph-regularized optimizations, we select graph Laplacian denoising (GLD) [10] and graph trend filtering (GTF) [6]. For graph filter banks, we consider graph Fourier transform (GFT) [10], spectral graph wavelet transform (SGWT) [7], graph quadrature-mirror-filters (QMF) [18] and critically sampled filter banks (CSFB) [19]. As competitive neural networks, we consider multilayer perception with three fully-connected layers [16], graph convolution networks (GCN) with three graph convolution layers [3], graph attention networks (GAT) with one graph attention layer [20] and graph autoencoder (GAE) with three graph convolution layers and one kron-reduction pooling layer [21].

Noise models. We consider two types of noises to validate the denoising algorithms: the mixture noise and Bernoulli noise. For the mixure noise, each element of $\mathbf{e}$ follows a mixture of Gaussian distribution and Laplace distribution; that is, $\mathbf{e}_i \sim \mathcal{N}(0, \sigma^2) + Laplace(0, b)$. By default, we set $\sigma = 0.2, b = 0.2$. For binary graph signals, we consider adding Bernoulli noise [22]; that is, we randomly select a subset of vertices and flip the associated binary values.

Evaluation metrics. To evaluate the denoising performance, the default metric is the normalized mean square error (NMSE); that is, $\text{NMSE} = \|\widehat{\mathbf{x}} - \mathbf{x}\|_2^2 / \|\mathbf{x}\|_2^2$, where $\mathbf{x} \in \mathbb{R}^N$ is a noiseless graph signal, and $\widehat{\mathbf{x}}$ is a denoised graph signal. A smaller value of NMSE indicates a better denoising performance. We also consider the normalized mean absolute error (NMAE); that is, $\text{NMAE} = \|\widehat{\mathbf{x}} - \mathbf{x}\|_1 / \|\mathbf{x}\|_1$. For binary graph signals, we evaluate by the error rate (ER), $\text{ER} = \sum_{i=1}^N \mathbf{1}(x_i \neq \widehat{x}_i)/N$, where x_i and $\widehat{x}_i$ are the ith element in $\mathbf{x}, \widehat{\mathbf{x}}$, respectively. A smaller value of ER indicates a better denoising performance. We also consider the F1 score, which is the harmonic mean of the precision and recall. A higher value of F1 indicates a better denoising performance.

3.3. Results

Table 1 shows the denoising performances across three datasets. Columns $2-3$ show the performances of the temperature dataset with two different numbers of graph signals: 1 and 365; Columns $4-5$ show the performances of the traffic dataset with two different numbers of graph signals: 1 and 24; and Columns $6-9$ show the performances of Cora dataset with two different numbers of graph signals: 1 and 7. We see that i) the proposed GUN significantly outperforms all the other competitive methods in terms of either NMSE or F1 score; ii) even denoising a single graph signal, the proposed GUN is much better than graph Laplacian denoising and graph wavelets; and iii) when more training data is available, the proposed two graph unrolling networks get better performance, reflecting the powerful learning ability to adapt to complicated data.

4. CONCLUSIONS

We propose graph unrolling networks, which is an interpretable neural network framework to denoise single or multiple noisy graph signals. The proposed graph unrolling networks expand algorithm unrolling to the graph domain. Through extensive experiments, we show that the proposed methods produce smaller denoising errors than both conventional denoising algorithms and state-of-the-art graph neural networks. Even for denoising a single graph signal, the normalized mean square error of the proposed networks is around 40% lower than that of graph Laplacian denoising, reflecting the advantages of learning from only a few training samples.

5. REFERENCES

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

[2] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Process. Mag.*, vol. 34, no. 4, pp. 18–42, 2017.

[3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[4] M. Li, S. Chen, Y. Zhao, Y. Zhang, Y. Wang, and Q. Tian, "Dynamic multiscale graph neural networks for 3d skeleton-based human motion prediction," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recogn.*, 2020.

[5] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Signal denoising on graphs via graph filtering," Dec. 2014, pp. 872–876.

[6] Y-X. Wang, J. Sharpnack, A. J. Smola, and R. J. Tibshirani, "Trend filtering on graphs," *J. Mach. Learn. Res.*, vol. 17, pp. 105:1–105:41, 2016.

[7] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmon. Anal.*, vol. 30, pp. 129–150, Mar. 2011.

[8] D. I. Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst, "Spectrum-adapted tight graph wavelet and vertex-frequency frames," *IEEE Trans. Signal Process.*, vol. 63, no. 16, pp. 4223–4235, 2015.

[9] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. abs/1912.10557, 2019.

[10] 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 Process. Mag.*, vol. 30, pp. 83–98, May 2013.

[11] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Representations of piecewise smooth signals on graphs," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*, 2016, pp. 6370–6374.

[12] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imaging Sciences*, vol. 1, no. 3, pp. 248–272, 2008.

[13] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*. 2010, pp. 399–406, Omnipress.

[14] O. Solomon, R. Cohen, Y. Zhang, Y. Yang, Q. He, J. Luo, R. J. G. van Sloun, and Y. C. Eldar, "Deep unfolded robust PCA with application to clutter suppression in ultrasound," *IEEE Transactions on Medical Imaging*, vol. 39, pp. 1051–1063, April 2020.

[15] Y. Li, M. Tofighi, J. Geng, V. Monga, and Y. C Eldar, "Efficient and interpretable deep blind image deblurring via algorithm unrolling," *IEEE Trans. Signal Process.*, vol. 6, pp. 666–681, Jan. 2020.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016, `http://www.deeplearningbook.org`.

[17] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[18] S. K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *IEEE Trans. Signal Process.*, vol. 60, pp. 2786–2799, June 2012.

[19] N. Tremblay and P. Borgnat, "Subgraph-based filterbanks for graph signals," *IEEE Trans. Signal Process.*, vol. 64, no. 15, pp. 3827–3840, 2016.

[20] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[21] T. Huu Do, D. Minh Nguyen, and N. Deligiannis, "Graph auto-encoder for graph signal denoising," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[22] S. Chen, Y. Yang, S. Zong, A. Singh, and J. Kovačević, "Detecting localized categorical attributes on graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 10, pp. 2725–2740, 2017.