Online Discriminative Graph Learning from Multi-Class Smooth Signals*,**

Seyed Saman Saboksayr*a,∗, Gonzalo Mateosa, Mujdat Cetinab, b

aDept. of Electrical and Computer Eng., Univ. of Rochester, Rochester, USA
bGoergen Institute for Data Science, Univ. of Rochester, Rochester, USA

Abstract

Graph signal processing (GSP) is a key tool for satisfying the growing demand for information processing over networks. However, the success of GSP in downstream learning and inference tasks such as supervised classification is heavily dependent on the prior identification of the relational structures. Graphs are natural descriptors of the relationships between entities of complex environments. The underlying graph is not readily detectable in many cases and one has to infer the topology from the observed signals which admit certain regularity over the sought representation. Assuming that signals are smooth over the latent class-specific graph is the notion we build upon. Firstly, we address the problem of graph signal classification by proposing a novel framework for discriminative graph learning. To learn discriminative graphs, we invoke the assumption that signals belonging to each class are smooth with respect to the corresponding graph while maintaining non-smoothness with respect to the graphs corresponding to other classes. Discriminative features are extracted via graph Fourier transform (GFT) of the learned representations and used in downstream learning tasks. Secondly, we extend our work to tackle increasingly dynamic environments and real-time topology inference. To this end, we adopt recent advances of GSP and time-varying convex optimization. We develop a proximal gradient (PG) method which can be adapted to situations where the data are acquired on-the-fly. Beyond discrimination, this is the first work that addresses the problem of dynamic graph learning from smooth signals where the sought network alters slowly. The introduced online framework is guaranteed to track the optimal time-varying batch solution under mild technical conditions. The validation of the proposed frameworks is comprehensively investigated using both synthetic and real data. The proposed classification pipeline

*This work was supported in part by the National Science Foundation under awards CCF-1750428, CCF-1934962 and ECCS-1809356.
**Part of the results in this paper were submitted to the 2021 ICASSP Conference [1, 2].
∗Corresponding author
Email addresses: ssaboks@ur.rochester.edu (Seyed Saman Saboksayr),
gmateosb@ece.rochester.edu (Gonzalo Mateos), mujdat.cetin@rochester.edu (Mujdat Cetin)

Preprint submitted to Signal Processing January 5, 2021
outperforms the-state-of-the-art methods when applied to the problem of emo-
tion classification based on electroencephalogram (EEG) data. We also perform
network-based analysis of epileptic seizures using electrocorticography (ECoG)
records. Moreover, by applying our method to financial data, our approach in-
fers the relationships between the stock-price behavior of leading US companies
and the recent events including the COVID-19 pandemic.

**Keywords:** Graph learning, network topology inference, graph signal
processing, online optimization, discriminative learning, smooth signals.

1. Introduction

We are witnessing an uptick of recent interest in signal and information pro-
cesing tasks involving network data, which go all the way from clustering similar
actors in a social network to classifying brain connectomes in computational
neuroscience studies [3]. Complex relational structures arising with network
datasets are naturally mapped out using graph representations. These graphs
are essential for understanding how different system elements interact with each
other, especially in dynamic environments. Moreover, they often provide useful
context (e.g., in the form of a prior or regularizer) to effectively extract action-
able information from graph signals, i.e., nodal attributes such as sensor mea-
surements, infected case counts across different localities, or discipline-related
labels assigned to scientific papers [4]. Some graphs may be readily obtained
based on prior surveys and experiments (e.g., road networks and molecule struc-
tures), or they could be (at least partially) directly observable as with social
and citation networks. In other cases, however, there is a need to utilize net-
work topology inference algorithms to estimate the graph structure from graph
signal observations; see e.g., [5], [3, Ch. 7]. This paper tackles a classification
problem involving network data, where class-specific graphs are learned from
labeled signals to obtain discriminative representations.

1.1. Related work

Popular graph construction schemes rely on ad hoc thresholding of user-
defined edgewise similarity measures or kernels. Albeit intuitive, these informal
approaches may fail to capture the actual intrinsic relationships between enti-
ties and do not enjoy any sense of optimality. Recognizing this shortcoming,
a different paradigm is to cast the graph-learning problem as one of selecting
the best representative from a family of candidate graphs by bringing to bear
elements of statistical modeling and inference [3, Ch. 7]. This path usually
leads to inverse problems that minimize criteria stemming from different mod-
els binding the (statistical) signal properties to the graph topology. Noteworthy
approaches from the Graph Signal Processing (GSP) literature usually assume
the signals are stationary with respect to the sought sparse network [5, 7], or
that they are smooth in the sense that a desirable graph is one over which
the nodal attributes exhibit limited variations; see e.g., [8, 9] and Section 2.3.
Graph learning via signal smoothness maximization is closely related to Gaussian graphical model selection when the precision matrix is constrained to have a combinatorial Laplacian structure; see also the recent review papers [5, 10].

The graph Fourier transform (GFT) is defined in terms of the network’s eigenvectors [4], so through this lens, topology inference is tightly related to the problem of learning efficient graph signal representations for denoising and compression, just to name several relevant downstream tasks. Acknowledging this perspective, [11] puts forth a graph learning approach facilitating accurate reconstruction of sampled bandlimited (i.e., smooth) graph signals. Laplacian-regularized dictionary learning was considered in [12], where the graph can be learned to effect sparse signal representations by better capturing the irregular data geometry. Naturally, some works have dealt with supervised classification of network data. Given a known graph over which the signals in all classes reside, the approach of [13] is to learn class-specific parametric graph sub-dictionaries. The discriminative graphical lasso algorithm in [14] constructs class-specific graphs from labeled signals, by minimizing the conventional Gaussian maximum likelihood objective augmented with a term that boosts the discrimination ability of the learnt representations. Here we advance this line of work in several directions as discussed in Section 1.2.

1.2. Proposed approach and contributions

We tackle a multi-class network topology inference problem for graph signal classification; see Section 3.1 for a formal statement of the problem. The signals in each class are assumed to be smooth (or bandlimited) with respect to unknown class-specific graphs. Given training signals the goal is to recover the class-specific graphs subject to signal smoothness constraints (Section 3.2), so that the obtained GFT bases can be subsequently used to classify unseen (and unlabeled) graph signals effectively as discussed in Section 3.3. This problem bears some similarities with subspace clustering [15]. Indeed, smoothness with respect to class-specific graphs can be interpreted as data living in multiple low-dimensional subspaces spanned by the respective low-frequency GFT components. However, here the problem is supervised and we contend there is value in learning the graph topologies beyond the induced discriminative transforms (i.e., subspaces) to reveal important structure in the data for each class. To this end, we develop a new framework for discriminative graph learning in which we encourage smoothness of each class signals on their corresponding inferred network representation as well as non-smoothness over all other classes’ topologies. Our convex criterion for graph learning builds on [9], which we augment with a judicious penalty term to effect class discriminability [14]. Remark 1 summarizes the distinctive features of our approach relative to [9, 14]. From an algorithmic standpoint, we develop lightweight proximal gradient iterations (Section 3.2) with well documented rates of convergence, acceleration potential, and that can be readily adapted to process streaming signals in an online fashion – our second main contribution.

As network data sources are increasingly generating real-time streams, training of models (here discriminative graph learning) must often be performed on-
line, typically with limited memory footprint and without waiting for a batch of signals to become available. To accommodate this pragmatic scenario, in Section 4.1 we develop online proximal gradient iterations to track the (possibly time-varying) network structure per class. Since the dynamic topology inference cost is strongly convex, we establish that the online graph estimates closely hover within a ball centered at the optimal time-varying batch solution (Section 4.2), and characterize the convergence radius even in a dynamic setting [10, 17, 18]. This algorithmic contribution is of independent interest beyond the classification theme of this paper. Dropping from the objective the regularization term that encourages discriminability, yields for the very first time an online graph learning algorithm under smoothness priors. Existing algorithms to identify dynamic network topologies from smooth signals operate in a batch fashion, and so their computational cost and memory requirements grow with time (i.e., the dataset size) [13, 20, 21].

The effectiveness and convergence of the proposed algorithms is corroborated through a comprehensive performance evaluation carried out in Section 5. We include numerical test cases with synthetic and real data in both batch and online settings, spanning several tasks across timely application domains such as a emotion classification from electroencephalogram (EEG) data, online graph learning from financial time-series, and network-based analysis of epileptic seizures from electrocorticography (ECoG) records. Concluding remarks are given in Section 6, while some technical details are deferred to the appendices.

This journal paper offers a more thorough treatment of online discriminative graph learning relative to its short conference precursors [1, 2]. This is achieved by means of expanded technical details, discussions and insights, as well as through more comprehensive performance evaluation studies with synthetic and real data experiments. In particular, focus in [1] is on emotion classification from EEG signals (the subject of Section 5.3), while [2] deals with online topology identification without a classification task in mind; see also Remark 2.

Notational conventions. The entries of a matrix \( X \) and a (column) vector \( x \) are denoted by \( X_{ij} \) and \( x_i \), respectively. Sets are represented by calligraphic capital letters. The notation \( \top \) and \( \dagger \) stand for transpose and pseudo-inverse, respectively; \( \mathbf{0} \) and \( \mathbf{1} \) refer to the all-zero and all-one vectors; while \( \mathbf{I}_N \) denotes the \( N \times N \) identity matrix. For a vector \( x \), \( \text{diag}(x) \) is a diagonal matrix whose \( i \)th diagonal entry is \( x_i \). The operators \( \circ \), \( \text{tr}(\cdot) \), and \( \text{vec}(\cdot) \) stand for Hadamard (element-wise) product, matrix trace, and vectorization, respectively. Lastly, \( \|X\|_p \) denotes the \( \ell_p \) norm of \( \text{vec}(X) \) and \( \|\cdot\|_F \) refers to the Frobenius norm.

2. Graph Signal Processing Background

We start by briefly reviewing the necessary GSP concepts and tools that will be used recurrently in the ensuing sections.

2.1. Graphs, signals and shift operators

We define the graph signal \( x = [x_1, \ldots, x_N]^\top \in \mathbb{R}^N \) over a weighted, undirected graph \( \mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W}) \), where \( \mathcal{V} = \{1, \ldots, N\} \) represents the node set of
cardinality $N$ and $\mathcal{E} \subseteq V \times V$ the set of edges. Following this notation, $x_i \in \mathbb{R}$ and $W \in \mathbb{R}^{N \times N}$ denote the signal value at node $i \in V$ and the adjacency matrix of edge weights, respectively. The symmetric and non-negative coefficients $W_{ij} = W_{ji} \in \mathbb{R}_{+}$ indicate the strength of the connection (or similarity) between node $i$ and node $j$. In the absence of connection [i.e., $(i,j) \notin \mathcal{E}$] one has $W_{ij} = 0$. Moreover, we assume that $\mathcal{G}$ does not include any self-loops which implies $W_{ii} = 0, \forall i \in V$. We will henceforth assume nodal degrees $d := W \mathbf{1}$ are uniformly lower bounded away from zero, i.e., $d \succeq d_{\min} \mathbf{1}$ (entry-wise inequality) for some prescribed $d_{\min} > 0$. Else if degrees become arbitrarily small, it is prudent to apply a threshold and remove the loosely connected nodes from $\mathcal{G}$. Extensions to directed graphs could be important [22], but are beyond the scope of this paper. Complex-valued signals can be accommodated as well [4].

The adjacency matrix $W$ encodes the topology of $\mathcal{G}$. More generally it is possible to define the so-called graph-shift operator $S$, which captures the sparsity pattern of $\mathcal{G}$ without any specific assumption on the value of its non-zero entries. The shift $S \in \mathbb{R}^{N \times N}$ is a symmetric matrix whose entry $S_{ij}$ can be non-zero only if $i = j$ or if $(i,j) \in \mathcal{E}$. Beyond the adjacency matrix $W$, results in spectral graph theory often motivate choosing the combinatorial graph Laplacian $L := \text{diag}(d) - W$ as well as their various degree-normalized counterparts. Other application-dependent alternatives have been proposed as well; see [4] and the references therein. In particular, $L$ plays a central role in defining a useful and intuitive graph Fourier transform (GFT) as described next.

2.2. Graph Fourier transform and signal smoothness

In order to introduce the network’s spectral basis and define the GFT, we decompose the (symmetric and positive semi-definite) combinatorial graph Laplacian as $L = \mathbf{V} \Lambda \mathbf{V}^\top$, where $\Lambda := \text{diag}(\lambda_1, \ldots, \lambda_N)$ denotes the diagonal matrix of non-negative eigenvalues and $\mathbf{V} := [v_1, \ldots, v_N]$ the orthonormal matrix of eigenvectors. The GFT of $x$ with respect to $L$ is the signal

$$\tilde{x} := \mathbf{V}^\top x. \quad (1)$$

The inverse (i)GFT of $\tilde{x} := [\tilde{x}_1, \ldots, \tilde{x}_N]^\top$ is given by $x = \mathbf{V} \tilde{x} = \sum_{k=1}^N \tilde{x}_k v_k$, which is a proper inverse due to the orthonormality of $\mathbf{V}$. The GFT encodes a notion of signal variability over $\mathcal{G}$ (akin to frequency in Fourier analysis of temporal signals) by synthesizing $x$ as a sum of orthogonal frequency components $v_k$ (see the iGFT definition above). The GFT coefficient $\tilde{x}_k$ is the contribution of $v_k$ to the graph signal $x$.

To elaborate on the notion of frequency for graph signals, consider the total variation (or Dirichlet energy) of $x$ with respect to the combinatorial graph Laplacian $L$ defined as

$$TV(x) := x^\top L x = \sum_{i \neq j} W_{ij} (x_i - x_j)^2 = \sum_{k=1}^N \lambda_k \tilde{x}_k^2. \quad (2)$$
The quadratic form (2) acts as a smoothness measure, because it effectively quantifies how much the graph signal \( x \) changes with respect to \( G \)’s topology. If we evaluate the total variation of eigenvector \( v_k \) (itself a graph signal), one immediately obtains \( \text{TV}(v_k) = \lambda_k \). Accordingly, the Laplacian eigenvalues \( 0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N \) can be viewed as graph frequencies indicating how the eigenvectors (i.e., frequency components) vary with respect to \( G \) [4].

Smoothness is a cardinal property of many real-world network processes; see e.g. [3]. The last equality in (2) suggests that smooth (or bandlimited) signals admit a sparse representation in the graph spectral domain. Intuitively, they tend to be spanned by few Laplacian eigenvectors associated with small eigenvalues. Exploiting this structure by means of priors or TV-based regularizers is at the heart of several graph-based statistical learning tasks including nearest-neighbor prediction (also known as graph smoothing), denoising, semi-supervised learning, and spectral clustering [4, 3]. More germane to our graph-learning problem is to use smoothness as the criterion to construct graphs on which network data admit certain regularity.

2.3. Learning graphs from observations of smooth signals

Consider the following network topology identification problem. Given a set \( \mathcal{X} := \{x_p\}_{p=1}^P \) of possibly noisy graph signal observations, the goal is to learn an undirected graph \( G(V, E, W) \) with \( |V| = N \) nodes such that the observations in \( \mathcal{X} \) are smooth on \( G \). In this section we review the solution proposed in [9], that we build on in the rest of the paper.

Given \( \mathcal{X} \) one can form the data matrix \( X = [x_1, \ldots, x_P] \in \mathbb{R}^{N \times P} \), and let \( \bar{x}_i^\top \in \mathbb{R}^{1 \times P} \) denote its \( i \)-th row collecting those \( P \) measurements at vertex \( i \). The key idea in [9] is to establish a link between smoothness and sparsity, namely

\[
\sum_{p=1}^P \text{TV}(x_p) = \text{tr}(X^\top LX) = \frac{1}{2} \|W \circ Z\|_1, \tag{3}
\]

where the Euclidean-distance matrix \( Z \in \mathbb{R}^{N \times N} \) has entries \( Z_{ij} := \|\bar{x}_i - \bar{x}_j\|^2 \), \( i, j \in V \). The intuition is that when the given distances in \( Z \) come from a smooth manifold, the corresponding graph has a sparse edge set, with preference given to edges \((i, j)\) associated with smaller distances \( Z_{ij} \).

Given these considerations, a general purpose framework for learning graphs under a smoothness prior is advocated in [9], which entails solving

\[
\min_W \|W \circ Z\|_1 + f(W) \tag{4}
\]

s. t. \( \text{diag}(W) = 0, \ W_{ij} = W_{ji} \geq 0, \ i \neq j \).

The convex objective function \( f(W) \) augments the smoothness criterion \( \|W \circ Z\|_1 \), and several choices have been proposed to e.g., recover common graph constructions based on the Gaussian kernel [23], accommodate time-varying graphs [24], or to scale other related graph learning algorithms [8]. Identity (3)
offers a favorable way of formulating the inverse problem \(1\), because the space of adjacency matrices can be described via simpler (meaning entry-wise decoupled) constraints relative to its Laplacian counterpart. As a result, the convex optimization problem \(1\) can be solved efficiently with complexity \(O(N^2)\) per iteration, by leveraging provably-convergent primal-dual solvers amenable to parallelization \(24\); see also Section 3.2 for a different optimization approach based on proximal gradient algorithms.

The framework in \(9\) has been shown to attain state-of-the-art performance and is not only attractive due to its computational efficiency but also due to its generality. Henceforth, we leverage and expand these ideas to tackle two important graph signal and information processing problems.

3. Discriminative Graph Learning

In an effort to address classification problems involving network data (Section 3.1), we bring to bear GSP insights to propose a new framework for learning discriminative graph-based representation of the signals. To this end, a proximal gradient (PG) algorithm is developed to solve the separable graph-learning problems per class (Section 3.2). After this training phase, the GFTs of the optimum graphs can be used to extract discriminative features of the test signals. In the test phase, we exploit the extracted GFT-based features in order to classify the test signals (Section 3.3).

3.1. Optimization Problem Formulation

Consider a dataset \(\mathcal{X} = \bigcup_{c=1}^{C} \mathcal{X}_c\) comprising labeled graph signals \(\mathcal{X}_c := \{x^{(c)}_p\}_{p=1}^{P_c}\) from \(C\) different classes. The signals in each class possess a very distinctive structure, namely they are assumed to be smooth (or bandlimited) with respect to unknown class-specific graphs \(G_c = (V, E_c, W_c)\), \(c = 1, \ldots, C\). This notion is analogous to a multiple linear sub-space model in which graph signals of each class are assumed to be spanned by a few vectors (namely, the basis of the corresponding low-dimensional subspace) \(15\). In fact, the closing discussion in Section 2.2 implies one can equivalently restate the class-conditional signal model assumptions as follows: there is a network \(G_c\) such that graph signals in \(\mathcal{X}_c\) are spanned by a few Laplacian eigenvectors (associated with small eigenvalues), for \(c = 1, \ldots, C\). Similar to \(14\), given \(\mathcal{X}\) the goal is to learn the class-specific adjacency matrices \(W_c\) under signal smoothness priors, so that the obtained GFT bases can be subsequently used to classify unseen (and unlabeled) graph signals effectively; see Section 3.3.

Our approach blends elements and ideas from the graph learning framework in \(9\) [cf. \(4\)] along with the discriminative graphical lasso estimator \(14\). Indeed, the algorithm in \(9\) optimizes network topology recovery under smoothness assumptions, but is otherwise agnostic to the performance of a potential downstream (say, classification) task the learnt graph may be integral to. Inspired by \(14\), we seek graph representations that capture the underlying network topology (i.e., the class structure), but at the same time are discriminative.
to boost classification performance. To this end, we propose to learn a graph representation $G_c$ per class by solving the following convex optimization problems [cf. (4)]

$$\min_{W_c \in W_m} \|W_c \odot Z_c\|_1 - \alpha 1^T \log (W_c 1) + \beta \|W_c\|_F^2 - \gamma \sum_{k \neq c} \|W_c \odot Z_k\|_1, \quad (5)$$

where $W_c$ is the adjacency matrix of $G_c$, constrained to the set $W_m = \{W \in \mathbb{R}^{N \times N} : W = W^T, \text{diag}(W) = 0\}$. Moreover, $Z_c$ is the distance matrix constructed from class $c$ signals $X_c := [x^{(c)}_1, \ldots, x^{(c)}_{P_c}] \in \mathbb{R}^{N \times P_c}$, while $\alpha$, $\beta$, and $\gamma$ are positive regularization parameters.

Taking a closer look at the objective function, minimizing $\|W_c \odot Z_c\|_1$ encourages a graph $W_c$ over which the signals in $X_c$ are smooth. At the same time, the last term enforces non-smoothness of the signals in the other $C - 1$ classes. This composite criterion will thus induce a GFT with better discrimination ability than the baseline $\gamma = 0$ case. In fact, the energy of class $c$ signals will be predominantly concentrated in lower frequencies, while the spectral content of the other classes is pushed towards high-pass regions of the spectrum; see Section 5.1 for corroborating simulations. Under Gaussian assumptions this can be interpreted as a Fisher discrimination criterion used in Linear Discriminant Analysis (LDA) [23, Ch. 4.3], which entails the other way around; see [14].

The logarithmic barrier on the nodal degree sequence $W_c 1$ precludes the trivial all-zero solution. Moreover, it ensures the estimated graph is devoid of isolated vertices. The Frobenius-norm regularization on the adjacency matrix $W_c$ controls the graphs’ edge sparsity pattern by penalizing larger edge weights (the sparsest graph is obtained for $\beta = 0$). Overall, this combination forces degrees to be positive but does not prevent most individual edge weights from becoming zero [9].

Remark 1 (Comparison with [9] and [14]). The topology inference problem (5) is subsumed by the general framework in [9], for a particular choice of the regularization function $f(W_i)$ in [4]. The distinctive goal here is that of discriminative transform learning, to effectively tackle a classification problem involving network data. Different from [9], in the next section we solve (5) using efficient PG iterations that can afford Nesterov-type acceleration and are readily adapted to process streaming data in an online fashion during training (Section 4). A discriminative graph learning approach, sharing similar objectives to ours was formulated in [14], but under the lens of probabilistic graphical model selection. Therein, graph signals are viewed as random vectors adhering to a Gaussian Markov random field distribution, where the unknown class-specific precision matrices typically play the role of graph Laplacians. However, the discriminative graphical lasso estimator in [14] is not guaranteed to return a valid graph Laplacian for each class, since the search is performed over the whole positive semi-definite cone. Incorporating Laplacian constraints may challenge the block coordinate-descent algorithm in [14]. Accordingly, one misses on the
GSP insights offered here in terms of signal smoothness and bandlimitedness in the graph spectral domain. Numerical tests in Section 5 show that the proposed approach outperforms the algorithms in [9] and [14] when it comes to classification, while it recovers interpretable graphs offering insights into the structure of the data classes.

3.2. Proximal-gradient algorithm

Given the optimization problem formulation in (5), in this section we develop a PG algorithm to recover the network topology of the graphs $G_c$ per class $c = 1, \ldots, C$; see [25] for a tutorial on proximal methods and their applications. PG methods have been popularized for $\ell_1$-norm regularized linear regression problems, and their desirable features of computational simplicity as well as suitability for online operation are starting to permeate naturally to the topology identification context of this paper [26].

To make (5) amenable to this optimization method, recall first that the adjacency matrix $W_c \in \mathbb{W}$ is symmetric with diagonal elements equal to zero. Therefore, the independent decision variables are effectively the upper-triangular elements $[W_c]_{ij}, j > i$, which we collect in the vector $w_c \in \mathbb{R}^{(N-1)/2}$. Second, it will prove convenient to enforce the non-negativity constraints by means of a penalty function augmenting the original objective. Just like [9], we add an indicator function $I\{w_c \succeq 0\} = 0$ if $w_c \succeq 0$, and $I\{w_c \succeq 0\} = \infty$ otherwise. Given all these considerations we reformulate the objective in (5) as the function $g(w_c)$ of a vector variable, and write the equivalent composite, non-smooth optimization problem:

$$
\min_{w_c} F(w_c) := \underbrace{-\alpha \mathbf{1}^T \log (S w_c) + \beta \|w_c\|^2}_{h(w_c)} + I\{w_c \succeq 0\} + 2 w_c^T z_c - \gamma \sum_{k \neq c} 2 w_k^T z_k,
$$

where $z_c$ is a vector containing the upper-triangular entries of $Z_c$, and $S \in \{0, 1\}^{N \times (N-1)/2}$ is such that $d_c = W_c \mathbf{1} = Sw_c$.

To arrive at the PG iterations, first note that the gradient of $g$ in (6) has the simple form

$$
\nabla g(w_c) = 4\beta w_c - \alpha S^T \left( \frac{1}{Sw_c} \right),
$$

where $1/(Sw_c)$ stands for element-wise division. Moreover, the gradient $\nabla g$ is a Lipschitz-continuous function with constant $\eta = \left( 4\beta + \frac{2\alpha (N-1)}{d_{\min}} \right)$; see Appendix A for a proof. With $\lambda > 0$ a fixed positive scalar, introduce the proximal
Algorithm 1: PG for graph learning, class $c$

Input parameters $\alpha, \beta, \gamma, \eta$, data $\{z_c^C\}_{c=1}^C$, initial $w_{c,0}$.

Set $k = 0$, $\mu \leq 2/\eta$ and $\bar{\mu} = 2\mu (z_c - \gamma \sum_{k \neq c} z_k)$.

while not converged do

Compute $\nabla g(w_{c,k}) = 4\beta w_{c,k} - \alpha S^\top \left( \frac{1}{5} w_{c,k} \right)$.

Update $w_{c,k+1} = [w_{c,k} - \mu \nabla g(w_{c,k}) - \bar{\mu}]_+$.

Increment $k \leftarrow k + 1$.

end

operator of a closed proper convex function $f : \mathbb{R}^N \to \mathbb{R} \cup \{+\infty\}$ evaluated at point $\nu \in \mathbb{R}^N$ as

$$\text{prox}_{\lambda f}(\nu) := \arg\min_x \left\{ f(x) + \frac{1}{2\lambda} \|x - \nu\|^2 \right\}. \tag{8}$$

With this definition, the PG updates with fixed step size $\mu < \frac{2}{\eta}$ to solve each of the class-specific graph learning problems (6) are given by (henceforth $k = 0, 1, 2, \ldots$ denote iterations)

$$w_{c,k+1} := \text{prox}_{\mu h} (w_{c,k} - \mu \nabla g(w_{c,k})). \tag{9}$$

It follows that graph estimate refinements are generated via the composition of a gradient-descent step and a proximal operator. Efficient evaluating of the latter is key to the success of PG methods. The proximal operator of $h$ in (6) is given by

$$\text{prox}_{\mu h}(w_c) = \begin{bmatrix} w_c - \frac{2\mu}{\gamma} \left(z_c - \gamma \sum_{k \neq c} z_k \right) \end{bmatrix}^+, \tag{10}$$

where $[\nu]^+ := \max(0, \nu)$ denotes the projection operator onto the non-negative orthant (the max operator is applied entry-wise). The non-negative soft-thresholding operator in (10) sets to zero all edge weights in $w_c$ that fall below the data-dependent thresholds in vector $\bar{\mu} := 2\mu (z_c - \gamma \sum_{k \neq c} z_k)$. The resulting iterations are tabulated under Algorithm 1 which will also serve as the basis for the online algorithm in Section 4.

The computational complexity is dominated by the gradient evaluation in (7), incurring a cost of $O(N^2)$ per iteration $k$ due to scaling and additions of vectors of length $N(N - 1)/2$. For sparse graphs $G_c$ the iterates $w_{c,k}$ tend to become (and remain) quite sparse at early stages of the algorithm by virtue of the soft-thresholding operations (a sparse initialization $w_{c,0}$ is useful to this end). Hence, it is possible to reduce the complexity further if Algorithm 1 is implemented carefully using sparse vector operations. All in all, Algorithm 1 scales well to large graphs with thousands of nodes and it is competitive with
the state of the art primal-dual solver in [9]. In terms of convergence, as \( k \to \infty \) the sequence of iterates \( \{x_k\} \) provably approaches a minimizer of the composite cost \( F \) in \( \{6\} \); see e.g., [25] for the technical details. Moreover, the worst-case convergence rate of PG algorithms is well documented (namely \( O(1/\varepsilon) \) iteration complexity to return a \( \varepsilon \)-optimal solution measured in terms of \( F \) values), and can be boosted to \( O(1/\sqrt{\varepsilon}) \) via Nesterov-type acceleration techniques that are readily applicable to Algorithm [1].

3.3. Classification via low-pass graph filtering

During the training phase of the classification task, the goal is to learn \( C \) class-specific graphs \( G_c \) from labeled graph signals \( X_c := \{x_{c,p}\}_{p=1}^P \). This can be accomplished by running \( C \) parallel instances of Algorithm 1. Let \( \hat{W}_c \) denote the estimated adjacency matrix of the graph representing class \( C \), and likewise let \( \hat{L}_c = \text{diag}(\hat{W}_c \mathbf{1}) - \hat{W}_c \) be the combinatorial graph Laplacian. Finally, let \( \hat{V}_c \) denote the orthonormal GFT basis of Laplacian eigenvectors for class \( C \); see Section 2.2.

In the operational or test phase, we are presented with an unseen and unlabelled graph signal \( x \) which we wish to classify into one of the \( C \) classes. To that end, we will process \( x \) with a filter-bank comprising \( C \) graph filters. The \( c \)-th branch yields the graph-frequency domain output

\[
\hat{x}_{F,c} = \hat{H}_c \hat{V}_c^\top x = \text{diag}(\hat{h}) \hat{x}_c = \hat{h} \circ \hat{x}_c, \tag{11}
\]

where \( \hat{x}_c \) are the GFT coefficients of \( x \) with respect to graph \( G_c \), and \( \hat{h} = [h_1, \ldots, h_N]^\top \) is the frequency response of an ideal low-pass filter with bandwidth \( w \in \{1, 2, \ldots, N\} \), i.e., \( \hat{h}_i := \mathbf{1}\{i \leq w\} \). The indicator function \( \mathbf{1}\{i \leq w\} = 1 \) if \( i \leq w \), and \( \mathbf{1}\{i \leq w\} = 0 \) otherwise. Typically, one chooses the tunable parameter \( w \) to be \( N/2 \) or smaller in order to implement a low-pass filter. Notice that while the frequency response \( \hat{h} \) is the same for all \( C \) branches, the graph filters \( H_c = V_c \text{diag}(\hat{h}) V_c^\top \) differ because the learnt graphs (hence the GFT transforms) vary across classes. From the definition of \( \hat{h} \), it immediately follows that \( \hat{x}_{F,c} \) is nothing else than the projection of \( x \) onto the eigenvectors of \( \hat{L}_c \) corresponding to the smallest \( w \) eigenvalues.

If \( x \) belongs to class \( c^\star \), say, then this graph signal should be smoothest with respect to \( G_{c^\star} \). Equivalently, for fixed (appropriately low) bandwidth \( w \) we expect the signal power to be largest when projected onto the GFT basis constructed from \( \hat{L}_c \). Accordingly, the adopted classification rule is simply

\[
\hat{c} = \arg\max_c \{\|\hat{x}_{F,c}\|^2\}. \tag{12}
\]

A classification error occurs whenever \( \hat{c} \neq c^\star \).

4. Online Discriminative Graph Learning

As network data sources are increasingly generating real-time streams, training of models (here discriminative graph learning) must often be performed on-
the-fly, typically (i) not affording to store or revisit past measurements; and (ii) without waiting for a whole batch of signals to become available. To accommodate this envisioned scenario, we switch gears to online estimation of $W_c$ (or even tracking $W_{c,t}$ in a non-stationary setting when the class-conditional distributions exhibit variations) from streaming data $\{x_1^{(c)}, \ldots, x_t^{(c)}, x_{t+1}^{(c)}, \ldots\}$.

A viable approach is to solve at each time instant $t = 1, 2, \ldots$, the composite, time-varying optimization problem [cf. (6)]

$$w_{c,t}^* := \arg\min_{w_c} F_{c,t}(w_c) := -\alpha 1^T \log (Sw_c) + 2\beta \|w_c\|^2$$

$$+ \begin{cases} 1 \{w_c \succeq 0\} & + 2w_c^T z_{c,1:t} - \gamma \sum_{k \neq c} 2w_c^T z_{k,1:t}, \quad (13) \end{cases}$$

In writing $z_{c,1:t} \in \mathbb{R}^N(N-1)/2$ we make explicit that the Euclidean-distance matrix is computed using all class-$c$ signals acquired by time $t$. In this infinite-memory case there is a need for normalization, for instance by using instead the time average $\bar{z}_{c,t} := \frac{z_{c,1:t}}{t}$. Otherwise the signal-dependent terms in (13) would explode due to data accumulation in $z_{c,1:t} = z_{c,1:t-1} + z_{c,t}$ as $t \to \infty$. This and other limited-memory averaging alternatives that are better suited for tracking variations in the graphs will be discussed in Section 4.1. Either way, as data come in, the edge-wise $\ell_1$-norm weights in $\|W_{c,t} \circ Z_{c,1:t}\|_1 = 2w_c^T z_{c,1:t}$ will fluctuate explaining the time dependence of $F_{c,t}(w_c)$ through its non-differentiable component $h_{c,t}(w_c)$. Notice that $g(w_c)$ is time-invariant for this particular instance of the topology inference problem, but this need not be the case in general [26].

### 4.1. Online algorithm construction

A na"ive sequential estimation approach consists of solving (13) repeatedly using the batch PG algorithm in Section 3.2. However (pseudo) real-time operation in delay-sensitive applications may not tolerate running multiple inner PG iterations per time interval, so that convergence to $w_{c,t}^*$ is attained for each $t$. For time-varying graphs it may not be even prudent to obtain $w_{c,t}^*$ with high precision (hence incurring high delay and unnecessary computational cost), since at time $t + 1$ a new datum arrives and the solution $w_{c,t+1}^*$ may be well off the prior estimate. These reasons motivate devising an efficient online and recursive algorithm to solve the time-varying optimization problem (13).

To this end, we build on recent advances in time-varying convex optimization, in particular the online PG methods in [16,18]. Our algorithm construction approach entails two steps per time instant $t = 1, 2, \ldots$. First, we recursively update the upper-triangular entries $z_{c,1:t} = z_{c,1:t-1} + z_{c,t}$ of the Euclidean-distance
Algorithm 2: Online discriminative graph learning, class $c$

**Input** parameters $\alpha, \beta, \gamma, \theta$, stream $z_{c,1}, z_{c,2}, \ldots$, initial $w_{c,1}, \bar{z}_{c,0}$.

for $t = 1, 2, \ldots,$ do

- Update $\bar{z}_{c,t} = (1 - \theta)\bar{z}_{c,t-1} + \theta z_{c,t}$.
- Update $\mu_t = \left(4\beta + \frac{2\alpha(N-1)}{\min(Sw_{c,t})}\right)^{-1}$.
- Compute $\nabla g(w_{c,t}) = 4\beta w_{c,t} - \alpha S^\top \left(\frac{1}{Sw_{c,t}}\right)$.
- Update $w_{c,t+1} = w_{c,t} - \mu_t \nabla g(w_{c,t}) - \mu_t$.

end

matrix via an exponential moving average (EMA), namely

$$\bar{z}_{c,t} = (1 - \theta)\bar{z}_{c,t-1} + \theta z_{c,t}. \quad (14)$$

The constant $\theta \in (0, 1)$ is a discount factor, which downweights past data to facilitate tracking dynamic graphs in non-stationary environments. The closer $\theta$ becomes to 1, the faster EMA discounts past observations. Another viable alternative is to use a fix-length sliding window [18], whereby the Euclidean distance matrix $z_{c,t-L+1:t}$ is computed using the most recent $L \geq 1$ graph signals from class $c$. For the infinite-memory case (suitable for time-invariant class-specific data distributions), one can rely on recursive updates of the sample mean $\bar{z}_{c,t} = \frac{1}{t-1}z_{c,t-1} + \frac{1}{t}z_{c,t}$. In any case, recursive updates of the vectorized Euclidean-distance matrix can be carried out in $O(N^2)$ complexity (and memory) per instant $t$. We initialize $\bar{z}_{c,0}$ from a few graph signals acquired before the online algorithm starts.

In the second step, we run a single iteration

$$w_{c,t+1} = \text{prox}_{\mu_t h_{c,t}}(w_{c,t} - \mu_t \nabla g(w_{c,t})), \quad (15)$$

of the batch graph learning algorithm developed in Section 3 to update $w_{t+1}$, with $\mu_t < \frac{2}{\eta} = \left(2\beta + \frac{\alpha(N-1)}{d_{\min}}\right)^{-1}$. An adaptive step-size rule $\mu_t$ whereby $d_{\min}$ is replaced with $\min(Sw_{c,t-1})$ is an alternative that works well in practice. In a nutshell, (15) amounts to letting iterations $k = 1, 2, \ldots$ in Algorithm 1 match the time instants $t$ of signal acquisition. Especially when dealing with slowly-varying graphs or in delay-tolerant applications, running a few PG iterations during $(t-1, t)$ would likely improve recovery performance; see also the discussion following Theorem 1. The gradient calculation is unchanged from the batch case [see (7)], and the non-negative soft-thresholding operator is now defined in terms of a time-varying threshold $\bar{\mu}_t := 2\mu_t \left(z_{c,t} - \gamma \sum_{k \neq c} \bar{z}_{k,t}\right)$. This is a direct manifestation of the fact that data enters the cost in (13) as time-varying $\ell_1$-norm weights. Accordingly, the computational complexity remains $O(N^2)$.
per instant $t$ – just as in the batch setting of Section 3. The resulting online iterations are tabulated under Algorithm 2.

Unlike recent approaches that estimate dynamic graph topologies from the observation of smooth signals \cite{19, 20, 21}, Algorithm 2’s memory footprint and computational cost per data sample $x_t^{(c)}$ do not grow with $t$. Specifically, the algorithms in \cite{19, 20, 21} operate in batch fashion and the number of optimization variables $(W_1, \ldots, W_T$ or their Laplacian counterparts) increase by a factor of $T$ when time-varying graphs are learnt over a horizon $t = 1, \ldots, T$. Moreover, these algorithms can only start once all data have been acquired and stored, rendering them unsuitable for online operation using streaming signals. Finally, the formulations in \cite{19, 20, 21} rely on explicit regularization terms such as $\sum_{t=2}^{T} \|W_t - W_{t-1}\|^2$ in order to constrain the temporal variation of the estimated graphs; see also \cite{27} for a recent approach whereby signals are assumed to be stationary over the dynamic networks. The proposed online PG iterations offer an implicit proximal regularization \cite[Sec. 2.1]{28}, so there is no need to enforce temporal smoothness explicitly in the criterion. Before moving on to issues of convergence, a remark is in order.

**Remark 2** (Online graph learning from smooth signals). By setting $\gamma = 0$ in \cite{13} and Algorithm 2 (to the best of our knowledge) we obtain the very first online algorithm to learn graphs from the observation of streaming smooth signals. This is a novel contribution of independent interest beyond the (discriminative) classification task-oriented focus of this paper. As we were preparing the final version of this manuscript, we became aware of recent related work on online graph learning via prediction-correction algorithms \cite{29}. While the framework proposed therein is undoubtedly general, concrete iterations are developed for the problem of time-varying GMRF model selection. Similar to the discriminative graphical lasso algorithm in \cite{14}, lacking Laplacian constraints on the sought precision matrices, the connections with smoothness priors are tenuous. Moreover, Algorithm 2 is a first-order method while prediction-correction iterations also require (second-order) Hessian information. This is likely to result in increased computational load. Finally, \cite{29} does not offer convergence guarantees. Altogether, \cite{29, 26} along with the ideas presented here underscore the potential of recent advances in time-varying convex optimization towards tackling dynamic network topology inference problems in various relevant settings.

### 4.2. Convergence analysis

Here we establish that Algorithm 2 can closely track the sequence of batch minimizers $w^*_t$ [recall \cite{13}] for large enough $t$; see also the corroborating simulations in Sections 5.2 and 5.5. Our results build on the performance guarantees derived in \cite{13} for online subspace clustering algorithms. To simplify notation, we henceforth drop the subindex $c$ from all relevant quantities. The results hold as stated for the estimated graphs in all classes $c = 1, \ldots, C$.

Instrumental to stating bounds for the tracking error $\|w_t - w^*_t\|_F$ is to note that $g(w)$ in \cite{13} is $4\beta$-strongly convex (see Appendix B) and its gradient is $\eta$-Lipschitz continuous. It thus follows that $w^*_t$ is the unique minimizer of \cite{13}. 

14
Next, let us define \( v_t := \|w_{t+1}^* - w_t^*\|_F \) to quantify the temporal variability of the optimal solution of (13). Since \( g(w) \) is strongly convex, we have the following (non-asymptotic) performance guarantee for Algorithm 2.

**Theorem 1.** For all \( t \geq 2 \), the sequence of iterates \( w_t \) generated by Algorithm 2 satisfies:

\[
\|w_t - w_t^*\|_F \leq \hat{L}_{t-1} \left( \|w_1 - w_1^*\|_F + \sum_{\tau=1}^{t-1} \frac{v_\tau}{L_\tau} \right),
\]

where \( L_t := \max \{|1 - 4 \mu_t \beta|, |1 - \mu_t \eta_t|\} \), \( \hat{L}_t := \prod_{\tau=1}^{t} L_\tau \). Moreover, for the sequence of objective values we can write \( F_t(w_t) - F_t(w_t^*) \leq \frac{\eta_t}{2} \|w_t - w_t^*\|_F \); see [30, Theorem 10.29].

Theorem 1 is adapted from [18, Theorem 1], and the proof can be obtained via straightforward modifications to the arguments therein. If one could afford taking \( i_t \) PG iterations (instead of a single one as in Algorithm 2) per time step \( t \), the performance gains can be readily evaluated by substituting \( L_t = \prod_{\tau=1}^{t} L_\tau \) in Theorem 1 to use the bound (16).

To gain further insights on the tracking bound, let us define \( \hat{L}_t := \max_{\tau=1,\ldots,t} L_\tau \), \( \hat{v}_t := \max_{\tau=1,\ldots,t} v_\tau \). With the aid of these upper bounds, the sum of the geometric series in the right-hand side of (16) can be simplified to

\[
\|w_t - w_t^*\|_F \leq \left( \hat{L}_{t-1} \right)^t \|w_1 - w_1^*\|_F + \frac{\hat{v}_t}{1 - \hat{L}_{t-1}}.
\]

Accordingly, \( \hat{L}_t = (\eta_t - 4\beta)/\eta_t < 1 \) since in Algorithm 2 we set \( \mu_t = \eta_t^{-1} \). Therefore, \( (\hat{L}_{t-1})^t \to 0 \) and Algorithm 2 hovers on the vicinity of the optimal solution with a misadjustment \( \hat{v}_t/(1 - \hat{L}_{t-1}) \). It follows that the tracking error increases with \( \hat{v}_t \) (rapidly-varying class-conditional graphs are more challenging to track) and also if the problem is badly conditioned (i.e., \( \beta \to 0 \) in which case \( \hat{L}_t \to 1 \)).

5. Numerical Experiments

Here we test both the discriminative graph learning and the online graph learning approach on different synthetic and real-world signals. A comprehensive performance evaluation is carried out for the proposed frameworks. In the case of discriminative graph learning we: (i) determine the classification accuracy; (ii) assess the similarity of the learned graph and the ground truth using F-measure of the detected edges (defined as the harmonic mean between edge precision and recall) [31]; (iii) illustrate the distribution of the GFT coefficients; (iv) compare with state-of-the-art methods; and (v) check if our findings are aligned with the literature. Also, we investigate the performance of the online discriminative graph learning approach by evaluating how it follows its offline
counterpart. For online graph learning we: (i) investigate how the online algorithm follows the optimum objective value; and (ii) study the similarity of the learned graph and the ground truth.

For the synthetic data experiments, we generate i.i.d. smooth signals with respect to the underlying graphs. The signals are drawn from a Gaussian distribution

\[
\mathbf{X} \sim \mathcal{N}(0, \mathbf{L}^\dagger + \sigma_e^2 \mathbf{I}_N),
\]

where \(\sigma_e\) represents the noise level; see e.g., [8]. Throughout, we perform a grid search to determine the best regularization parameters \(\alpha, \beta, \gamma\) and report the results. Also, after learning the underlying graph, we remove the weak connections by thresholding.

5.1. Learning discriminative graphs from synthetic data

In this section, we illustrate the proficiency of our proposed framework in synthetic settings. In order to assess the performance of our method we compare it with other state-of-the-art methods e.g., the framework proposed in [9] (Kalofolias), and the method proposed in [14] (DISC). Moreover, to study if the learned graphs have practical impacts, we checked the classification performance using the raw signals as features and Support Vector Machine (SVM) [32] as the classifier. The evaluation is conducted in two different scenarios as follows.

5.1.1. Scenario One

Consider two different types of random graphs namely: (i) Erdős-Rényi (ER) [33], and (ii) Barabási-Albert (BA) [34]. In this experiment, we generate a 60-nodes ER graph with edge-formation probability \(p = 0.1\), and a 60-nodes BA graph by adding a new node to the graph each time, connecting to 3 existing nodes in the graph. This will give us graphs with approximately 0.1 edge sparsity. Then, 100 independent random signals are generated using (18) half of which are smooth over the ER graph and the other half over the BA graph. The signals have different levels of noise \(\sigma_e \in [0.05, 3]\) and we use 80 percent of the data for training and 20 percent for test. This procedure has been repeated 50 times with 50 different ER and BA graphs.

The summary of this evaluation can be found in Fig. 1. Fig. 1(a,b,c) present the average of classification accuracy and F-measure over 50 trials. Also to consider the statistics of our findings we plot the 95 percent confidence interval as a shaded area. One can see in Fig. 1(a) that our discriminative graph learning approach outperforms state-of-the-art methods in classification accuracy. By increasing the noise level we can see the drop in the performance of all methods. While, our proposed method manages to outperform the other state-of-the-art methods in all noise levels. Note that the proposed method in [9] favors accurate signal representation and does not include any information about classes (discriminative parameter). Therefore, it makes sense if Kalofolias algorithm cannot do well in noisy situations because of the aforementioned limitation. With that being said, Fig. 1(b,c) shows that our proposed method
Despite including discrimination into account, can still accurately recover the ground truth graph as well as celebrated methods for graph learning like Kalofolias method [9]. The DISC method [14] focuses more on discrimination thus it gets outperformed by Kalofolias and our method in capturing the ground truth graph, see Fig. 1(b,c). Nevertheless, DISC method [14] is able to perform as good as Kalofolias method [9] in terms of classification accuracy since it takes into account the discrimination objective. These results suggest that our method is more robust in noisy scenarios and it can track the underlying graph while maintaining discriminative power.

As mentioned in Section 3.3 we use the cumulative relative energy of the projected signals onto the one-third low-frequency GFT basis. Accordingly, we plot the GFT coefficients of both graphs and the cumulative relative energy in Fig. 1. As shown in Fig. 1(d,e), the mean of magnitude of the first one-third of the GFT coefficients with respect to the corresponding graphs are larger than the other. This leads to high accuracy in the classification task. However, Fig. 1(f) shows that the mean of the cumulative relative energy with respect to the corresponding graph is higher not only for the first one-third of the GFT basis but also is higher for all the GFT basis.
5.1.2. Scenario Two

In this scenario, unlike the previous one, we generate 100 random graphs by replacing a proportion of edges with new ones drawing from the same family of random graphs. This way we try to duplicate a real-world data-set wherein graphs have some common similar patterns but also have differences. The 100 time sample smooth signals over them are created using (18) where $\sigma_e = 1$. The ER graphs have edge-formation probability $p = 0.1$, and BA graphs are generated via adding a new node to the graph each time, connecting to 5 existing nodes in the graph. Graphs have 100 vertices and we repeat this procedure 50 times and with different proportions of edge replacement.

Since we generate 100 different random graphs, the evaluation of the F-measure is not feasible. Therefore, in Fig. 2 we show the classification accuracy averaged over 50 trials and the shaded area is the 95 percent confidence interval. As one can see in Fig. 2 our framework outperforms state-of-the-art methods in all edge replacement proportions.

We analyze the online discriminative graph learning approach in a similar fashion. To this end, we create 100 random graphs with 100 nodes by saving 25% of edges and replace other edges from the same family of random graphs. The ER and BA graphs have the same property as above. We assume that the training signals are streaming online and we have access to whole test data. We run the online algorithm one iteration per new time sample. The result of the classification accuracy averaged over 50 trials is shown in Fig. 3. Fig. 3 indicates that after enough acquired time samples (iterations), the online algorithm’s result will converge to its offline counterpart.
5.2. Online graph learning from synthetic data

To assess the performance of the proposed online graph learning algorithm (without discriminative term; see Remark 2), we test it on simulated streaming data. We generate a piecewise-constant sequence of two random ER graphs (edge formation probability $p = 0.1$) with $N = 30$ nodes. The initial graph switches in the middle of the experiment (specifically after $t = 10000$ time samples). The final graph is obtained from the initial one by redrawing 40 percent of its edges. Then, we synthetically generate i.i.d. smooth signals with respect to the time-varying underlying graph. The signals are drawn from \[ \sigma_n = 0.05 \] by setting $\sigma_n = 0.05$. Results are averaged over 10 independent Monte Carlo trials. We compute the time-varying solution $F_t$ in (13) by running the proposed batch PG algorithm until convergence after each 500 time samples and consider that to be the optimal value. Note that for this experiment, we set $\theta$ in Algorithm 2 to 0.003.

Fig. 4(a) shows that after around 500 time samples (iterations) the objective value of the online Algorithm 2 reaches the optimal value of its time-varying batch counterpart. The observed oscillation depends on the step size $\mu_t$ in Algorithm 2. As expected, increasing the step size leads to faster convergence with larger oscillations. Conversely, reducing the step size leads to smoother and slower convergence. The other noteworthy observation is that the objective value markedly increases when the graph changes (after 10000 time samples), but the online algorithm can effectively track the dynamic graph after a sufficient number of samples have been acquired. A similar trend can be observed for the F-measure of the detected edges; see Fig. 4(b). For visual comparison we illustrate the snapshots of the estimated underlying graph at time samples 10000 and 20000, and compare it with the ground truth; see Fig. 4(c).
5.3. EEG emotion recognition

In this section, we study the application of the discriminative graph learning algorithm on a real-world problem namely emotion recognition using EEG signals. To this end, we apply our method to a widely used and publicly available EEG data-set called DEAP [35].

The DEAP data-set contains EEG and peripheral physiological signals of 32 participants. The data were recorded while subjects were watching one-minute long music videos. Each participant has 40 trials (music videos) and rates each video in terms of the levels of valence, arousal, like/dislike, dominance, and familiarity [35]. In this study, we focus on the valence and arousal classification. Ratings are decimal numbers between 1 and 9 and in order to make this a classification task we divide the ratings into two classes of low when the ratings are smaller than 5, and high when ratings are larger than or equal to 5. We exploit the pre-processed version of the data set which contains 32 EEG channels with 128 Hz sampling rate and the 3 second pre-trial baseline is discarded.

We perform the classification task in leave-one-trial-out scheme where for each subject we use 39 trials as the training set and test on the one remaining trial. We repeat this 40 times for each subject and report the mean of accuracy. Consequently, this is a subject dependent procedure which means that we perform training and classification for each subject separately. Classification follows the same procedure described in Section 3.3 and we project the test signals on the first $1/4$ of the GFT basis. The discriminative graph is learned for the training set via Algorithm 1 and then normalized to have unit Frobenius norm. The average classification accuracy over all the trials and all the participants is reported in Table 1. The results of the other state-of-the-art methods are also reported for the comparison. Results of the proposed method (DISC) in [14] are omitted since the provided code did not converge for some of the DEAP dataset trials.

As it is shown in Table 1, the proposed discriminative graph learning approach outperforms state-of-the-art methods in the classification of emotions. Moreover, we can see 6 and 5 percent improvement in the classification of va-
Table 1: The average of EEG emotion classification accuracy over all subjects

| Study               | Valence | Arousal |
|---------------------|---------|---------|
| **Proposed method** | **92.73** | **93.44** |
| Kalofolias [9]      | 86.56   | 88.91   |
| Chao and Liu [36]   | 77.02   | 76.13   |
| Koelstra et al. [35]| 57.60   | 62.00   |
| Chung and Yoon [37] | 66.60   | 66.40   |
| Rozgić et al. [38]  | 76.90   | 69.10   |
| Chen et al. [39]    | 76.17   | 73.59   |
| Tripathi et al. [40]| 81.40   | 73.36   |

Valence and arousal, respectively with respect to Kalofolias method which is the foundation of our model and that is because of the added discriminative term. Additionally, there exist some other studies that aim at the classification of emotion. However, their classification schemes are different than what we considered in our experimental analysis e.g. [41, 42] divide the valence-arousal space into four sub-spaces and perform a 4-class emotion recognition.

Now that we established the superior performance of our proposed framework, it is of interest to investigate whether there is any useful information in the underlying learned graphs. To this end, we first discard the trials that have ratings in (4.5, 5.5) where the participants themselves are not confident enough in rating the trials. Secondly, we study the connections that are significantly different between classes. Finally, we illustrate the decomposed signals and the eigenvectors with respect to the low, mid, and high frequency in order to investigate whether our findings are aligned with the literature.

Accordingly, we learn two graphs corresponding to low and high emotions per person, using the parameters that gave us the best results in the classification. In Fig. 5 we show the mean of the connections over all subjects and the significantly different connections between low and high emotions. Interestingly, it seems that the edge weights are related to the intensity of the emotions, i.e., valence and arousal. As shown in Fig. 5 the graphs corresponding to high valence/arousal (Fig. 5(b,e)) have stronger connection than the learned graphs for low valence/arousal (Fig. 5(a,d)). Moreover, the statistical way to check whether the learned representations are discriminative is through analyzing the significantly different connections. Here, we apply a non-parametric test of the null hypothesis named Wilcoxon rank-sum test [43]. Fig. 5(c,f) show connections that are significantly different between low and high valence with $p \leq 0.002$ and between low and high arousal with $p \leq 0.03$, respectively. Our findings indicate that the significantly different connections across different classes incorporate approximately all the channels that were originally mentioned in [35, Table 4].

Frequency analysis is the common theme in the EEG signal processing literature. Therefore, we conduct a similar analysis with the help of GFT and the underlying learned graphs. Fig. 6(a,b,c) demonstrate the average of the magnitudes of the eigenvector sets associated with low (first 1/4 components), mid,
Figure 5: The average of learned graphs for (a) low, and (b) high valence over all subjects. (c) Significantly different connections between low and high valence with \( p \leq 0.002 \). The mean of learned graphs for (d) low, and (e) high arousal over all subjects. (f) Significantly different connections between low and high arousal with \( p \leq 0.03 \). These results suggest the learned graphs for low and high emotions show significant difference with each other. Color shades in (c) and (f) encode connections involving different brain regions.

and high (last 1/4 components) frequencies, respectively. The arousal counterparts are shown in Fig. 6(d,e,f). The asymmetrical pattern of the frontal EEG activity can be seen in Fig. 6(a,b) which is consistent with the findings in [44] about valence. Also, as it is noted in Fig 5(c) most of the connections are related to the frontal lobe (red-shaded colors). That being said, the authors in [45] discovered that for classifying positive from negative emotions, the features are generally in the right occipital lobe and parietal lobe for the alpha band, the central lobe for the beta band, and the left frontal and right temporal lobe for the gamma band. Since we do the classification via low-frequency components, the employed features are mainly in the low-frequency representation Fig. 6(a,d). Evidently there are different patterns in the left frontal, right temporal, central, and parietal lobe between high and low valence/arousal; see Fig. 6(a,d). Moreover, in high frequencies, we observe different patterns in the right parietal and left occipital.

To continue the frequency analysis we decompose the EEG signals into the low, mid, and high components with respect to the underlying graph. The mean of the absolute value of the decomposed signals are plotted on the human scalp and can be found in Fig. 7. Different patterns can be recognized in low and high valence/arousal. More specifically, different patterns in the frontal lobe are captured in low frequency for both valence and arousal. For mid-frequency, we detect a distinct pattern between high and low valence in the left frontal and right temporal.
5.4. Seizure detection

Here, we investigate the applicability of the discriminative graph learning method on electrocorticogram (ECoG) data for detecting epileptic seizures. Epilepsy is known to be one of the most common brain disorders in the world and is a chronic neurological condition [3, Ch. 4.4]. To this end, we use one of the publicly available data sets for epileptic seizure detection. This data set contains 8 instances of seizures that are acquired from a human patient with epilepsy. The ECoG data are recorded via 76 electrodes at the sampling rate of 400 Hz. Sixty-four electrodes are in the shape of an 8 × 8 grid and implanted at the cortical level. In order to record the voltage activity from mesial temporal structures, the other 12 electrodes (two strips of 6 electrodes) are located deeper in the brain [46, 3]. Each seizure is annotated by an epileptologist to identify two key segments relating to the seizure i.e., pre-ictal (before the epileptic attack) and ictal (during the epileptic attack) periods.

The task at hand is to classify pre-ictal and ictal periods. To this end,

\footnote{The data can be found at \url{http://math.bu.edu/people/kolaczyk/datasets.html}}
we use Algorithm 1 to capture graph representations and do the classification in a leave-one-trial-out scheme. We achieve 100 percent classification accuracy averaged over all trials. The corresponding learned representations are in Fig. 8. Fig. 8(a) shows the learned graph for the pre-ictal period while Fig. 8(b) shows the representation for the ictal period. Our findings in Fig. 8 indicate a reduction in the overall level of brain connectivity when the patient is experiencing an epileptic seizure. Consistent with the findings in [46], we observe the edge drops are more vivid in the bottom corner of the grid and in the two strips. According to [46], the edge drops in the brain network might lead to disclosing the brain regions that facilitate the seizure.

5.5. Stock price data analysis

In this test case, we perform a study involving real financial data. We consider the daily stock prices for ten large US companies, namely: Microsoft (MSFT), Apple (AAPL), Amazon (AMZN), Facebook (FB), Alphabet class A shares (GOOGL), Alphabet class C shares (GOOG), Johnson & Johnson (JNJ), Berkshire Hathaway (BRK-B), Visa (V), and Procter & Gamble (PG). We collect their daily stock prices from Yahoo! Finances over the time period from May 1st, 2019 to August 1st, 2020. The aforementioned time period consists of 317 days worth of data, and it overlaps with the very recent COVID-19 pandemic that led to significant market instabilities. Under normal circumstances, we would expect limited variations in the network describing the pairwise relationships between the chosen stock prices, since these large companies are well-established in the market [47]. However, events like COVID-19 can cause abrupt changes in such a network. We run Algorithm 2 to estimate daily graphs in order to monitor the sudden changes in the stock market. To this end, we consider both (i) logarithmic stock prices and (ii) the relative temporal variation of the stock prices $|\text{price}_{today} - \text{price}_{yesterday}|/|\text{price}_{yesterday}|$ as the input.
signal. Following studies like [47, 19], we quantify the variation of the network via (i) relative temporal deviation \( \| W_t - W_{t-1} \|_F / \| W_{t-1} \|_F \), and (ii) algebraic connectivity, i.e. the second smallest eigenvalue of the combinatorial graph Laplacian \( L_t \). Since we have limited amount of data, the discount factor \( \theta \) and the employed step size in Algorithm 2 are 0.8 and \( \mu = 2/\eta \), respectively.

First, we use logarithmic stock prices as the input of our graph learning problem. In Fig. 9(a) we plot the S&P500 log-price as an indicator of the market’s condition. The relative temporal variation of the learned graphs is illustrated in Fig. 9(b). The graphs are obtained by setting the regularization parameters as \( \alpha = 0.316 \), \( \beta = 0.05 \), and \( \gamma = 0 \). Fig. 9(a) shows that the COVID-19 impact on the markets started in February 2020 and it got to its worse situation during March 2020. The following sudden changes are apparent by inspection of the spikes in Fig. 9(b): (i) September 2019, (ii) November 2019, (iii) December 2019, (iv) January 2020, (v) March 2020, and (vi) April 2020. These abrupt changes are consistent with major events occurring during these time periods. In September 2019 there was a congressional hearing regarding the impeachment inquiry of President Trump. The changes in November 2019 can be explained by the optimism surrounding the U.S.-China trade negotiations. The U.S. House of Representatives’ voted to impeach President Trump and that caused another sudden change in December 2019. The big spike at the end of January 2020 is probably the result of the World Health Organization (WHO) declaring a global health emergency due to the COVID-19 pandemic. In the middle of March 2020, President Trump declared a national emergency due to COVID-19. The epidemic appears to continue (adversely) affecting the market well into April 2020, when the new COVID-19 cases per day reached record values of 30000.

Then we exploit the relative temporal variation of the stock prices. The relative temporal variation of S&P500 is shown in Fig. 10(a). We capture the underlying graph by setting the regularization parameters \( \alpha = 3.162 \), \( \beta = 0.177 \), and \( \gamma = 0 \). Then we extract the algebraic connectivity from the captured
Figure 10: (a) The relative temporal variation of S&P500 price per day. (b) The algebraic connectivity of the estimated graph over the days when the relative temporal variation of stock prices is the graph signal. The algebraic connectivity notes that at the time of crisis the relation graph tends to get connected, while it is disconnected when the market is stable.

graphs. The algebraic connectivity over the time interval of interest is shown in Fig. 10(b). The algebraic connectivity indicates that the graph is usually disconnected unless a crisis happens which can also be seen in Fig. 11. This is because we use the the relative temporal variation of stock prices as the input graph signals. The stock prices usually drop at the time of a crisis and this makes the gradient to be negative for all nodes. Therefore, we will see the underlying graph is connected as the relative variations exhibit smoothness over the network. However, in the stable times the gradient will be different for each company. Thus the graph we learn tends to be disconnected. The peaks in Fig. 10(b) show phenomena that had a great influence on the stock market. We already mentioned these incidents above. However, the algebraic connectivity reveals sudden changes in June 2020, which has not been detected by the relative temporal variation. In early June 2020, the market observed a rise in stock prices due to reopening trade. However, it seems that the reopening effect faded when the number of new COVID-19 cases increased. Therefore, the market experienced a drop in stock prices. Moreover, to show these sudden changes in the relation network, we plot the captured graphs in Fig. 11. Some of these graphs are in stable times and some in the time of crisis.

6. Conclusion

In this work, we proposed a novel framework for online task-driven graph learning by leveraging recent advances at the crossroads of GSP and time-varying convex optimization. A particular task at hand is supervised classification of signals assumed to be smooth over latent class-conditional graphs. To recover said graph representations of the data during the training phase, we develop proximal gradient algorithms for discriminative network topology inference given (possibly streaming) smooth signals per class. The intuition behind the optimality criterion guiding the search of the class-conditional graphs
is simple: we look for a class $c$ graph over which (i) class $c$ signals are smooth (to capture the data structure); and (ii) all other signals are non-smooth (for discriminability). The GFTs of the optimum graphs can then be used to extract discriminative features of the test signals we wish to classify. We validate the proposed classification pipeline on both synthetic and real data, for instance in an emotion recognition task from EEG data where we outperform state-of-the-art methods developed for the DEAP data-set. In this context, the recovered class-conditional networks can inform significant connections between brain regions to discern among emotional states.

Moreover, results in this paper contribute to the algorithmic foundations of online network topology inference from streaming signals (in a classification setting and beyond). We develop for the first time an online algorithm to track the possibly slowly-varying topology of a dynamic graph, given streaming signals that are smooth over the sought network. The algorithmic framework comes with tracking performance guarantees, is computationally efficient and has a fixed memory storage requirement irrespective of the temporal horizon’s span. Numerical tests with synthetic data show the online topology estimates closely track the performance of the batch estimator benchmark, even when the network is subject to abrupt changes in connectivity. We also applied our algorithm to recent stock-prices from leading US companies, which resulted in time-varying graphs that reveal interesting relationships among the firms as well as the financial market’s response to the COVID-19 pandemic.

Acknowledgment

The authors would like to thank Mark Kramer and Eric Kolaczyk (Boston University) for providing useful clarifications on the epilepsy dataset studied in Section 5.4.

Appendix A. Lipschitz constant of $\nabla g$

Recall the expression for $\nabla g$ in (7). For arbitrary points $x, y \in \mathbb{R}_+^{N(N-1)/2}$ representing graph topologies we have
\[ \| \nabla g(x) - \nabla g(y) \| = \left\| 4\beta (x - y) - \alpha S^T \left( \frac{1}{Sx} - \frac{1}{Sy} \right) \right\| \]
\[ \leq 4\beta \| x - y \| + \alpha \| S \| \left\| \frac{1}{Sx} - \frac{1}{Sy} \right\| \]
\[ \leq 4\beta \| x - y \| + \alpha \frac{\| S \|^2}{d_{\min}^2} \| Sx - Sy \| \]
\[ = 4\beta \| x - y \| + \alpha \frac{\| S \|^2}{d_{\min}^2} \| x - y \| \]
\[ = \left( 4\beta + \frac{2\alpha(N-1)}{d_{\min}^2} \right) \| x - y \|, \]

establishing the Lipschitz continuity of the gradient. In deriving the last equality, we used that \( \| S \| = \sqrt{2(N-1)} \) – a simple result we state next for completeness.

**Lemma 1.** With reference to (6), let \( S \in \{0, 1\}^{N \times N(N-1)/2} \) be such that \( W_c 1 = S w_c \). Then the spectral norm of \( S \) is \( \| S \| = \sqrt{2(N-1)} \), where \( N \) is the number of nodes in \( G_c \).

**Proof.** Since \( S \) maps \( w_c \) to nodal degrees \( d_c \), then \( S \) has exactly \( N-1 \) ones in each row. Hence the diagonal entries of \( SS^T \) are all \( N-1 \) and the off-diagonal entries are equal to 1. Moreover, \( \| S \| \) is equal to the square root of the spectral radius of \( SS^T = (N-2)I + 11^T \). The eigenvalues are solutions to the characteristic equation

\[
\det (SS^T - \lambda I) = \det ((N-2)I + 11^T - \lambda I) = \det\left((N-2-\lambda)I + 11^T\right) = \det(R) + 11^T \text{adj}(R)1 = \prod_{i=1}^{N} R_{ii} \sum_{j=1}^{N} \prod_{i \neq j} R_{ij} = (N-2-\lambda)^N + N(N-2-\lambda)^{N-1} = (2N-2-\lambda)(N-2-\lambda)^{N-1} = 0.
\]

In arriving at the third equality we used the Sherman-Morrison formula, where \( \text{adj}(R) \) denotes the adjugate matrix of \( R \). All in all, the roots are \( \lambda_1 = 2(N-1) > \lambda_2 = \cdots = \lambda_N = N-2 \) and the result follows. \( \square \)

**Appendix B. Strong convexity of \( g \)**

To establish that \( g \) in (13) is strongly convex with constant \( m = 4\beta > 0 \), it suffices to note that

28
\[ g(w) - \frac{m}{2} \|w\|^2 = -\alpha \mathbf{1}^T \log(Sw) \]

is a convex function.

References

[1] S. S. Saboksayr, G. Mateos, and M. Cetin, “EEG-based emotion classification using graph signal processing,” in IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), Toronto, ON, 2021 (submitted).

[2] ——, “Online graph learning under smoothness priors,” in IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), Toronto, ON, 2021 (submitted).

[3] E. D. Kolaczyk, Statistical Analysis of Network Data: Methods and Models. New York, NY: Springer-Verlag, 2009.

[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, 2018.

[5] G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro, “Connecting the dots: Identifying network structure via graph signal processing,” IEEE Signal Process. Mag., vol. 36, no. 3, pp. 16–43, 2019.

[6] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat, “Characterization and inference of graph diffusion processes from observations of stationary signals,” vol. PP, no. 99, Sep. 2017.

[7] S. Segarra, A. Marques, G. Mateos, and A. Ribeiro, “Network topology inference from spectral templates,” vol. 3, no. 3, pp. 467–483, Aug. 2017.

[8] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, “Learning Laplacian matrix in smooth graph signal representations,” IEEE Trans. Signal Process., vol. 64, no. 23, pp. 6160–6173, 2016.

[9] V. Kalofolias, “How to learn a graph from smooth signals,” in Artif. Intel. and Stat. (AISTATS), 2016, pp. 920–929.

[10] G. B. Giannakis, Y. Shen, and G. V. Karanikolas, “Topology identification and learning over graphs: Accounting for nonlinearities and dynamics,” Proc. IEEE, vol. 106, no. 5, pp. 787–807, 2018.

[11] S. Sardellitti, S. Barbarossa, and P. Di Lorenzo, “Graph topology inference based on sparsifying transform learning,” IEEE Trans. Signal Process., vol. 67, no. 7, pp. 1712–1727, 2019.

[12] Y. Yankelevsky and M. Elad, “Dual graph regularized dictionary learning,” vol. 2, no. 4, pp. 611–624, 2016.

[13] D. Thanou, D. I. Shuman, and P. Frossard, “Learning parametric dictionaries for signals on graphs,” IEEE Trans. Signal Process., vol. 62, no. 15, pp. 3849–3862, 2014.
[14] J. Kao, D. Tian, H. Mansour, A. Ortega, and A. Vetro, “Disc-GLasso: Discriminative graph learning with sparsity regularization,” in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2017, pp. 2956–2960.

[15] R. Vidal, “Subspace clustering,” *IEEE Signal Process. Mag.*, vol. 28, no. 2, pp. 52–68, 2011.

[16] E. Dall’Anese, A. Simonetto, S. Becker, and L. Madden, “Optimization and learning with information streams: Time-varying algorithms and applications,” *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 71–83, 2020.

[17] A. Simonetto, E. Dall’Anese, S. Paternain, G. Leus, and G. B. Giannakis, “Time-varying convex optimization: Time-structured algorithms and applications,” *Proc. IEEE*, pp. 1–17, 2020.

[18] L. Madden, S. Becker, and E. Dall’Anese, “Online sparse subspace clustering,” in *IEEE Data Sci. Wrksp. (DSW)*, 2019, pp. 248–252.

[19] J. V. d. M. Cardoso and D. P. Palomar, “Learning undirected graphs in financial markets,” *arXiv preprint arXiv:2005.09958 [stat.ML]*, 2020.

[20] V. Kalofolias, A. Loukas, D. Thanou, and P. Frossard, “Learning time varying graphs,” in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2017, pp. 2826–2830.

[21] K. Yamada, Y. Tanaka, and A. Ortega, “Time-varying graph learning with constraints on graph temporal variation,” *arXiv preprint arXiv:2001.03346 [eess.SP]*, 2020.

[22] A. G. Marques, S. Segarra, and G. Mateos, “Signal processing on directed graphs,” *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 99–116, 2020.

[23] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed. New York: Springer, 2009.

[24] N. Komodakis and J. Pesquet, “Playing with duality: An overview of recent primal–dual approaches for solving large-scale optimization problems,” *IEEE Signal Process. Mag.*, vol. 32, no. 6, pp. 31–54, 2015.

[25] N. Parikh and S. Boyd, “Proximal algorithms,” *Foundations and Trends in Optimization*, vol. 1, no. 3, p. 127–239, 2014.

[26] R. Shafipour and G. Mateos, “Online topology inference from streaming stationary graph signals with partial connectivity information,” *Algorithms*, vol. 13, no. 9, pp. 1–19, Sep. 2020.

[27] M. Navarro, Y. Wang, A. G. Marques, C. Uhler, and S. Segarra, “Joint inference of multiple graphs from matrix polynomials,” *arXiv preprint arXiv:2010.08120 [stat.ML]*, 2020.

[28] A. Beck and M. Teboulle, “A fast iterative shrinkage-thresholding algorithm for linear inverse problems,” *SIAM J. Imag. Sci.*, vol. 2, pp. 183–202, Jan. 2009.
A. Natali, M. Coutino, E. Isufi, and G. Leus, “Online time-varying topology identification via prediction-correction algorithms,” in IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), Toronto, ON, 2021 (submitted; see also arXiv:2010.11634 [eess.SP]).

A. Beck, First-order Methods in Optimization. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2018.

C. Manning, P. Raghavan, and H. Schütze, “Introduction to information retrieval,” Nat. Lang. Eng., vol. 16, no. 1, pp. 100–103, 2010.

C. Cortes and V. Vapnik, “Support-vector networks,” Machine learning, vol. 20, no. 3, pp. 273–297, 1995.

P. Erdős and A. Rényi, “On the evolution of random graphs,” Publ. Math. Inst. Hung. Acad. Sci., vol. 5, no. 1, pp. 17–60, 1960.

A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” Science, vol. 286, no. 5439, pp. 509–512, 1999.

S. Koelstra, C. Muhl, M. Soleymani, J.-S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, and I. Patras, “DEAP: A database for emotion analysis using physiological signals,” IEEE Trans. Affect. Comput., vol. 3, no. 1, pp. 18–31, 2011.

H. Chao and Y. Liu, “Emotion recognition from multi-channel EEG signals by exploiting the deep belief-conditional random field framework,” IEEE Access, vol. 8, pp. 33002–33012, 2020.

S. Y. Chung and H. J. Yoon, “Affective classification using Bayesian classifier and supervised learning,” in Intl. Conf. Control, Automation and Systems (ICCAS), 2012, pp. 1768–1771.

V. Rozgić, S. N. Vitaladevuni, and R. Prasad, “Robust EEG emotion classification using segment level decision fusion,” in IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), 2013, pp. 1286–1290.

M. Chen, J. Han, L. Guo, J. Wang, and I. Patras, “Identifying valence and arousal levels via connectivity between EEG channels,” in Intl. Conf. Affect. Comput. and Intel. Interaction (ACII), 2015, pp. 63–69.

S. Tripathi, S. Acharya, R. D. Sharma, S. Mittal, and S. Bhattacharya, “Using deep and convolutional neural networks for accurate emotion classification on DEAP dataset,” in Proc. of the Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 4746–4752.

W. Zheng, J. Zhu, and B. Lu, “Identifying stable patterns over time for emotion recognition from EEG,” IEEE Trans. Affect. Comput., vol. 10, no. 3, pp. 417–429, 2019.

P. Li, H. Liu, Y. Si, C. Li, F. Li, X. Zhu, X. Huang, Y. Zeng, D. Yao, Y. Zhang, and P. Xu, “EEG based emotion recognition by combining functional connectivity network and local activations,” IEEE Trans. Biomed. Eng., vol. 66, no. 10, pp. 2869–2881, 2019.
[43] M. P. Fay and M. A. Proschan, “Wilcoxon-Mann-Whitney or t-test? on assumptions for hypothesis tests and multiple interpretations of decision rules,” Statistics surveys, vol. 4, p. 1, 2010.

[44] L. A. Schmidt and L. J. Trainor, “Frontal brain electrical activity (EEG) distinguishes valence and intensity of musical emotions,” Cognition & Emotion, vol. 15, no. 4, pp. 487–500, 2001.

[45] D. Nie, X. Wang, L. Shi, and B. Lu, “EEG-based emotion recognition during watching movies,” in Intl. IEEE/EMBS Conf. Neural Eng., 2011, pp. 667–670.

[46] M. A. Kramer, E. D. Kolaczyk, and H. E. Kirsch, “Emergent network topology at seizure onset in humans,” Epilepsy Res., vol. 79, no. 2, pp. 173–186, 2008.

[47] D. Hallac, Y. Park, S. Boyd, and J. Leskovec, “Network inference via the time-varying Graphical Lasso,” in Proc. of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 205–213.