

# Quantifying the Alignment of Graph and Features in Deep Learning

Yifan Qian<sup>1</sup>, Paul Expert<sup>1</sup>, Tom Rieu, Pietro Panzarasa<sup>1</sup>, and Mauricio Barahona<sup>1</sup>

**Abstract**—We show that the classification performance of graph convolutional networks (GCNs) is related to the alignment between features, graph, and ground truth, which we quantify using a subspace alignment measure (SAM) corresponding to the Frobenius norm of the matrix of pairwise chordal distances between three subspaces associated with features, graph, and ground truth. The proposed measure is based on the principal angles between subspaces and has both spectral and geometrical interpretations. We showcase the relationship between the SAM and the classification performance through the study of limiting cases of GCNs and systematic randomizations of both features and graph structure applied to a constructive example and several examples of citation networks of different origins. The analysis also reveals the relative importance of the graph and features for classification purposes.

**Index Terms**—Data alignment, deep learning, graph convolutional networks (GCNs), graph subspaces, principal angles.

## I. INTRODUCTION

DEEP learning encompasses a broad class of machine learning methods that use multiple layers of nonlinear processing units in order to learn multilevel representations for detection or classification tasks [1]–[5]. The main realizations of deep multilayer architectures are the so-called deep neural networks (DNNs), which correspond to artificial neural networks (ANNs) with multiple layers between input and output

layers. DNNs have been shown to perform successfully in processing a variety of signals with an underlying Euclidean or grid-like structure, such as speech, images, and videos. Signals with an underlying Euclidean structure usually come in the form of multiple arrays [1] and are known for their statistical properties, such as locality, stationarity, and hierarchical compositionality from local statistics [6], [7]. For instance, an image can be seen as a function on Euclidean space (the 2-D plane) sampled from a grid. In this setting, the locality is a consequence of local connections, stationarity results from shift-invariance, and compositionality stems from the intrinsic multiresolution structure of many images [4]. It has been suggested that such statistical properties can be exploited by convolutional architectures via DNNs, namely, (deep) convolutional neural networks (CNNs) [8]–[10], that are based on four main ideas: local connections, shared weights, pooling, and multiple layers [1]. The role of the convolutional layer in a typical CNN architecture is to detect local features from the previous layer that are shared across the image domain, thus largely reducing the parameters compared with traditional fully connected feed-forward ANNs.

Although deep learning models, and in particular CNNs, have achieved highly improved performance on data characterized by an underlying Euclidean structure, many real-world data sets do not have a natural and direct connection with the Euclidean space. Recently, there has been interest in extending deep learning techniques to non-Euclidean domains, such as graphs and manifolds [4]. An archetypal example is social networks, which can be represented as graphs with users as nodes and edges representing social ties between them. In biology, gene regulatory networks represent relationships between genes encoding proteins that can up- or down-regulate the expression of other genes. In this article, we illustrate our results through examples stemming from another kind of relational data with no discernible Euclidean structure, yet with a clear graph formulation, namely, citation networks, where nodes represent documents and an edge is established if one document cites the other [11].

To address the challenge of extending deep learning techniques to graph-structured data, a new class of deep learning algorithms, broadly named graph neural networks (GNNs), has been recently proposed [4], [12], [13]. In this setting, each node of the graph represents a sample, which is described by a feature vector, and we are additionally provided with relational information between the samples that can be formalized as a graph. GNNs are well suited to

Manuscript received February 14, 2020; revised June 15, 2020 and September 29, 2020; accepted November 25, 2020. The work of Yifan Qian was supported by the China Scholarship Council Program under Grant 201706020176. The work of Paul Expert was supported in part by the National Institute for Health Research (NIHR) Imperial Biomedical Research Centre (BRC) under Grant NIHR-BRC-P68711 and in part by the Engineering and Physical Sciences Research Council (EPSRC) Centre for Mathematics of Precision Healthcare under Grant EP/N014529/1. The work of Mauricio Barahona was supported by the EPSRC Centre for Mathematics of Precision Healthcare under Grant EP/N014529/1. (Corresponding authors: Mauricio Barahona; Pietro Panzarasa.)

Yifan Qian and Pietro Panzarasa are with the School of Business and Management, Queen Mary University of London, London E1 4NS, U.K. (e-mail: y.qian@qmul.ac.uk; p.panzarasa@qmul.ac.uk).

Paul Expert is with the School of Public Health, Imperial College London, London SW7 2AZ, U.K. (e-mail: paul.expert08@imperial.ac.uk).

Tom Rieu was with the Department of Mathematics, Imperial College London, London SW7 2AZ, U.K. He is now with Facebook, London W1T 1FB, U.K. (e-mail: rieu@fb.com).

Mauricio Barahona is with the Department of Mathematics, Imperial College London, London SW7 2AZ, U.K. (e-mail: m.barahona@imperial.ac.uk).

Data is available on-line at <https://github.com/haczqyf/gcn-data-alignment/tree/master/alignment/data>.

This article has supplementary material provided by the authors and color versions of one or more figures available at <https://doi.org/10.1109/TNNLS.2020.3043196>.

Digital Object Identifier 10.1109/TNNLS.2020.3043196

node (i.e., sample) classification tasks. For a recent survey of this fast-growing field, see [14].

Generalizing convolutions to non-Euclidean domains is not straightforward [15]. Recently, graph convolutional networks (GCNs) have been proposed [16] as a subclass of GNNs with convolutional properties. The GCN architecture combines the full relational information from the graph together with the node features to accomplish the classification task, using the ground-truth class assignment of a small subset of nodes during the training phase. GCNs have shown improved performance for semisupervised classification of documents (described by their text) into topic areas, outperforming methods that rely exclusively on text information without the use of any citation information, e.g., multilayer perceptron (MLP) [16].

However, we would not expect such an improvement to be universal. In some cases, the additional information provided by the graph (i.e., the edges) might not be consistent with the similarities between the features of the nodes. In particular, in the case of citation graphs, it is not always the case that documents cite other documents that are similar in content. As we will show in the following with some illustrative data sets, in those cases, the conflicting information provided by the graph means that a graph-less MLP approach outperforms GCN. Here, we explore the relative importance of the graph with respect to the features for classification purposes and propose a geometric measure based on subspace alignment to explain the relative performance of GCN against different limiting cases.

Our hypothesis is that a degree of alignment among the three layers of information available (i.e., the features, the graph, and the ground truth) is needed for GCN to perform well, and any degradation in the information content leads to an increased misalignment of the layers and worsened performance. We will first use randomization schemes to show that the systematic degradation of the information contained in the graph and the features leads to a progressive worsening of GCN performance. Second, we propose a simple spectral alignment measure and show that this measure correlates with the classification performance in a number of data sets: 1) a constructive example built to illustrate our work; 2) CORA, a well-known citation network benchmark; 3) AMiner, a newly constructed citation network data set; and 4) two subsets of Wikipedia: Wikipedia I, where GCN outperforms MLP, and Wikipedia II, where MLP outperforms GCN.

## II. RELATED WORK

### A. Neural Networks on Graphs

The first attempt to generalize neural networks on graphs can be traced back to Gori *et al.* [17] who proposed a scheme combining recurrent neural networks (RNNs) and random walk models. Their method requires the repeated application of contraction maps as propagation functions until the node representations reach a stable fixed point. This method, however, did not attract much attention when it was proposed. With the current surge of interest in deep learning, this work has been reappraised in a new and modern form: [18] introduced modern techniques for RNN training based on the original

GNN framework, whereas [19] proposed a convolution-like propagation rule on graphs and methods for graph-level classification. Nonspectral methods have also been successfully proposed. For example, [20] shows how diffusion-based representations can be learned from graph-structured data and used as the basis for node classification by introducing a diffusion-convolution operation. Niepert *et al.* [21] convert graphs locally into sequences fed into a conventional 1-D CNN, which needs the definition of a node ordering in a preprocessing step.

The first formulation of CNNs on graphs (GCNNs) was proposed by Bruna *et al.* [22]. These researchers applied the definition of convolutions to the spectral domain of the graph Laplacian. While being theoretically salient, this method is unfortunately impractical due to its computational complexity. This drawback was addressed by subsequent studies [15]. In particular, [15] leveraged fast localized convolutions with Chebyshev polynomials. In [16], a GCN architecture was proposed via a first-order approximation of localized spectral filters on graphs. In that work, Kipf and Welling considered the task of semisupervised transductive node classification where labels are only available for a small number of nodes. Starting with a feature matrix  $X$  and a network adjacency matrix  $A$ , they encoded the graph structure directly using a neural network model  $f(X, A)$  and trained on a supervised target loss function  $\mathcal{L}$  computed over the subset of nodes with known labels. Their proposed GCN was shown to achieve improved accuracy in classification tasks on several benchmark citation networks and a knowledge graph data set. In our study, we examine how the properties of features and the graph interact in the model proposed by Kipf and Welling for semisupervised transductive node classification in citation networks. The architecture and propagation rules of this method are detailed in Section II-C.

### B. Spectral Graph Convolutions

We now present briefly the key insights introduced by Bruna *et al.* [22] to extend CNNs to the non-Euclidean domain. For an extensive recent review, the reader should refer to [4].

We study GCNs in the context of a classification task for  $N$  samples. Each sample is described by a  $C^0$ -dimensional feature vector, which is conveniently arranged into the feature matrix  $X \in R^{N \times C^0}$ . Each sample is also associated with the node of a given graph  $\mathcal{G}$  with  $N$  nodes, with edges representing additional relational (symmetric) information. This undirected graph is described by the adjacency matrix  $A \in R^{N \times N}$ . The ground-truth assignment of each node to one of  $F$  classes is encoded into a 0–1 membership matrix  $Y \in R^{N \times F}$ .

The main hurdle is the definition of a convolution operation on a graph between a filter  $g_w$  and the node features  $X$ . This can be achieved by expressing  $g_w$  onto a basis encoding information about the graph, e.g., the adjacency matrix  $A$  or the Laplacian  $L = D - A$ , where  $D = \text{diag}(A\mathbf{1})$ . This real symmetric matrix has an eigendecomposition  $L = U\Lambda U^T$ , where  $U$  is the matrix of column eigenvectors with associated eigenvalues collected in the diagonal matrix  $\Lambda$ . The filters can then be expressed in the eigenbasis  $U$  of  $L$

$$g_w = U g_w(\Lambda) U^T \quad (1)$$

with the convolution between filter and signal given by

$$g_w \star X = U g_w(\Lambda) U^T X. \quad (2)$$

The signal is, thus, projected onto the space of the graph, filtered in the frequency domain, and projected back onto the nodes.

### C. Graph Convolutional Networks

A GCN is a semisupervised method, in which a small subset of the node ground-truth labels is used in the training phase to infer the class of unlabeled nodes. This type of learning paradigm, where only a small amount of labeled data is available, therefore, lies between supervised and unsupervised learning.

Furthermore, the model architecture and, thus, the learning depend explicitly on the structure of the network. Hence, the addition of any new data point (i.e., a new node in the network) will require retraining of the model. GCNs are, therefore, an example of a transductive learning paradigm, where the classifier cannot be generalized to data that have not already seen. Node classification using a GCN can be seen as a label propagation task: given a set of seed nodes with known labels, the task is to predict which label will be assigned to the unlabeled nodes given a certain topology and attributes.

1) *Layerwise Propagation Rule and Multilayer Architecture*: Our study uses the multilayer GCN proposed in [16]. Given the matrix  $X$  with sample features and the (undirected) adjacency matrix  $A$  of the graph  $\mathcal{G}$  encoding relational information between the samples, the propagation rule between layers  $\ell$  and  $\ell + 1$  (of size  $C^\ell$  and  $C^{\ell+1}$ , respectively) is given by

$$H^{\ell+1} = \sigma^\ell(\widehat{A}H^\ell W^\ell) \quad (3)$$

where  $H^\ell \in R^{N \times C^\ell}$  and  $H^{\ell+1} \in R^{N \times C^{\ell+1}}$  are matrices of activation in the  $\ell$ th and  $(\ell + 1)$ th layers, respectively;  $\sigma^\ell(\cdot)$  is the threshold activation function for layer  $\ell$ ; the weights connecting layers  $\ell$  and  $\ell + 1$  are stored in the matrix  $W^\ell \in R^{C^\ell \times C^{\ell+1}}$ . Note that the input layer contains the feature matrix  $H^0 \equiv X$ .

The graph is encoded in  $\widehat{A} = \widetilde{D}^{-1/2} \widetilde{A} \widetilde{D}^{-1/2}$ , where  $\widetilde{A} = A + I_N$  is the adjacency matrix of a graph with added self-loops,  $I_N$  is the identity matrix, and  $\widetilde{D} = \text{diag}(\widetilde{A}\mathbf{1})$  is a diagonal matrix containing the degrees of  $\widetilde{A}$ . In the remainder of this work (and to ensure comparability with the results in [16]), we use  $\widehat{A}$  as the descriptor of the graph  $\mathcal{G}$ .

Following [16], we implement a two-layer GCN with propagation rule (3) and different activation functions for each layer, i.e., a rectified linear unit (ReLU) for the first layer and a softmax unit for the output layer

$$\sigma^0 : \text{ReLU}(x_i) = \max(x_i, 0) \quad (4)$$

$$\sigma^1 : \text{softmax}(x)_i = \frac{\exp(x_i)}{\sum_i \exp(x_i)} \quad (5)$$

where  $x$  is a vector. The model then takes the simple form

$$Z = f(X, A) = \text{softmax}(\widehat{A} \text{ReLU}(\widehat{A}XW^0) W^1) \quad (6)$$

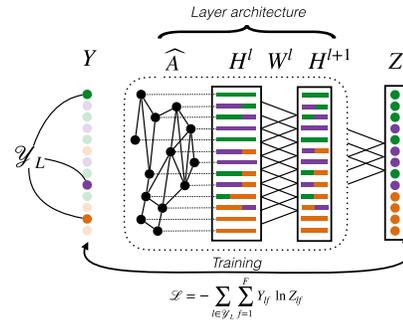


Fig. 1. Schematic of GCN used. The graph  $\widehat{A}$  is applied to the input of each layer  $\ell$  before it is funneled into the input of layer  $\ell + 1$ . The process is repeated until the output has dimension  $N \times F$  and produces a predicted class assignment. During the training phase, the predicted assignments are compared against a subset of values  $\mathcal{Y}_L$  of the ground truth.

where the softmax function is applied rowwise and the ReLU is applied elementwise. Note that there is only one hidden layer with  $C^1$  units. Hence,  $W^0 \in R^{C^0 \times C^1}$  maps the input with  $C^0$  features to the hidden layer, and  $W^1 \in R^{C^1 \times C^2}$  maps these hidden units to the output layer with  $C^2 = F$  units, corresponding to the number of classes of the ground truth. In this semisupervised multiclass classification, the cross-entropy error over all labeled instances is evaluated as follows:

$$\mathcal{L} = - \sum_{l \in \mathcal{Y}_L} \sum_{f=1}^F Y_{lf} \ln Z_{lf} \quad (7)$$

where  $\mathcal{Y}_L$  is the set of nodes that have labels. The weights of the neural network ( $W^0$  and  $W^1$ ) are trained using gradient descent to minimize the loss  $\mathcal{L}$ . A visual summary of the GCN architecture is shown in Fig. 1. The reader is referred to [16] for details and in-depth analysis. Although GCN was introduced as a simplified form of spectral-based GNNs, it shows a natural connection with spatial-based GNNs, in which graph convolutions are defined by information propagation. Hence, our study of the alignment of graph and features is related more widely to graph-feature correlations, such as spatial autocorrelations measured with, e.g., Moran's and Geary's indices [23], [24] that capture how the features of nodes influence each other via network structure.

## III. METHODS

### A. Randomization Strategies

To test the hypothesis that a degree of alignment across information layers is crucial for a good classification performance of GCN, we gradually randomize the node features, the node connectivity, or both. For the randomization to give a meaningful notion of alignment, at least one ingredient needs to be kept constant. Since we focus on the alignment of graphs and features, we keep the ground-truth constant.

1) *Randomization of the Graph*: The edges of the graph are randomized by rewiring a percentage  $p_{\widehat{A}}$  of edge stubs (i.e., "half-edges") under the constraint that the degree distribution remains unchanged. This randomization strategy is described in Algorithm 1, which is based on the configuration

**Algorithm 1: Randomization of the Graph**

**Input:** A graph  $G(V, E)$ , where  $V$  is the set of nodes and  $E$  is the set of edges, and a randomization percentage  $0 \leq p_{\hat{A}} \leq 100$ .

**Output:** A randomized graph  $G_{p_{\hat{A}}}(V, E')$

1. Choose a random subset of edges  $E_r$  from  $E$  with  $|E_r| = \lfloor |E| \times p_{\hat{A}}/100 \rfloor$ , and denote the unrandomized edges in  $E$  as  $E_u$ .
2. Obtain the degree sequence of nodes from  $E_r$ , and build a stub list  $l_s$  based on the degree sequence.
3. Obtain a randomized stub list  $l'_s$  by shuffling  $l_s$ , and randomized edges  $E'_r$  by connecting the stubs in the corresponding positions of the two stub lists  $l_s$  and  $l'_s$ .
4. Compute  $E_u \cup E'_r$ , remove multiedges and self-loops, and obtain the final edge set  $E'$ .
5. Generate randomized graph  $G_{p_{\hat{A}}}(V, E')$  from node set  $V$  and edge set  $E'$ .

model [25]. Once a randomized realization of the graph is produced, the corresponding  $\hat{A}$  is computed.

2) *Randomization of the Features:* The features were randomized by swapping feature vectors between a percentage  $p_X$  of randomly chosen nodes following the procedure described in Algorithm 2.

**Algorithm 2: Randomization of the Features**

**Input:** A feature matrix  $X \in R^{N \times C^0}$ , and a randomization percentage  $0 \leq p_X \leq 100$ .

**Output:** A randomized feature matrix  $X_{p_X} \in R^{N \times C^0}$

1. Choose at random  $N_r$  rows from  $X$ , where  $N_r = \lfloor N p_X/100 \rfloor$ .
2. Swap randomly the  $N_r$  rows to obtain  $X_{p_X}$ .

A fundamental difference between the two randomization schemes is that the graph randomization alters its spectral properties as it gradually destroys the graph structure, whereas the randomization of the features preserves its spectral properties in the principal component analysis (PCA) sense, i.e., the principal values are the same, but the loadings on the components are swapped. Hence, the feature randomization still alters the classification performance because the features are reassigned to nodes that have a different environment, thereby changing the result of the convolution operation defined by the  $H^\ell$  activation matrices (3).

**B. Limiting Cases**

To interrogate the role that the graph plays in the classification performance of a GCN, it is instructive to consider three limiting cases.

- 1) *No Graph:*  $A = \mathbf{00}^T$ . If we remove all the edges in the graph, the classifier becomes equivalent to an MLP, a classic feed-forward ANN. The classification is based

solely on the information contained in the features, as no graph structure is present to guide the label propagation.

- 2) *Complete Graph:*  $A = \mathbf{11}^T - I_N$ . In this case, the mixing of features is immediate and homogeneous, corresponding to a mean-field approximation of the information contained in the features.
- 3) *No Features:*  $X = I_N$ . In this case, the label propagation and assignment are purely based on graph topology.

An illustration of these limiting cases can be found in the top row of Table II.

**C. Spectral Alignment Measure**

In order to quantify the alignment between the features, the graph, and the ground truth, we propose a measure based on the chordal distance between subspaces, as follows.

1) *Chordal Distance Between Two Subspaces:* Recent work by Ye and Lim [26] has shown that the distance between two subspaces of different dimension in  $\mathbb{R}^n$  is necessarily defined in terms of their principal angles.

Let  $\mathcal{A}$  and  $\mathcal{B}$  be two subspaces of the ambient space  $\mathbb{R}^n$  with dimensions  $\alpha$  and  $\beta$ , respectively, with  $\alpha \leq \beta < n$ . The principal angles between  $\mathcal{A}$  and  $\mathcal{B}$  denoted  $0 \leq \theta_1 \leq \theta_2 \leq \dots \leq \theta_\alpha \leq (\pi/2)$  are defined recursively as follows [27], [28]:

$$\theta_1 = \min_{a_1 \in \mathcal{A}, b_1 \in \mathcal{B}} \arccos \left( \frac{|a_1^T b_1|}{\|a_1\| \|b_1\|} \right)$$

$$\theta_j = \min_{\substack{a_j \in \mathcal{A}, b_j \in \mathcal{B} \\ a_j \perp a_1, \dots, a_{j-1} \\ b_j \perp b_1, \dots, b_{j-1}}} \arccos \left( \frac{|a_j^T b_j|}{\|a_j\| \|b_j\|} \right), \quad j = 2, \dots, \alpha.$$

If the minimal principal angle is small, then the two subspaces are nearly linearly dependent, i.e., almost perfectly aligned. A numerically stable algorithm that computes the canonical correlations (i.e., the cosine of the principal angles) between subspaces is given in Algorithm 3.

**Algorithm 3: Principal Angles [27], [28]**

**Input:** matrices  $A_{n \times \alpha}$  and  $B_{n \times \beta}$  with  $\alpha \leq \beta < n$ .

**Output:** cosines of the principal angles

$$\theta_1 \leq \theta_2 \leq \dots \leq \theta_\alpha \text{ between } \mathcal{R}(A) \text{ and } \mathcal{R}(B),$$

the column spaces of  $A$  and  $B$ .

1. Find orthonormal bases  $Q_A$  and  $Q_B$  for  $A$  and  $B$  using the QR decomposition:  $Q_A^T Q_A = Q_B^T Q_B = I$ ;  $\mathcal{R}(Q_A) = \mathcal{R}(A)$ ,  $\mathcal{R}(Q_B) = \mathcal{R}(B)$ .
2. Compute the singular value decomposition (SVD):  $Q_A^T Q_B = U C V^T$ .
3. Extract the diagonal elements of  $C$ :  $C_{ii} = \cos \theta_i$ , to obtain the canonical correlations  $\{\cos \theta_1, \dots, \cos \theta_\alpha\}$ .

The principal angles are the basic ingredient of a number of the well-defined Grassmannian distances between subspaces [26]. Here, we use the chordal distance given by

$$d(\mathcal{A}, \mathcal{B}) = \sqrt{\sum_{j=1}^{\alpha} \sin^2 \theta_j}. \quad (8)$$

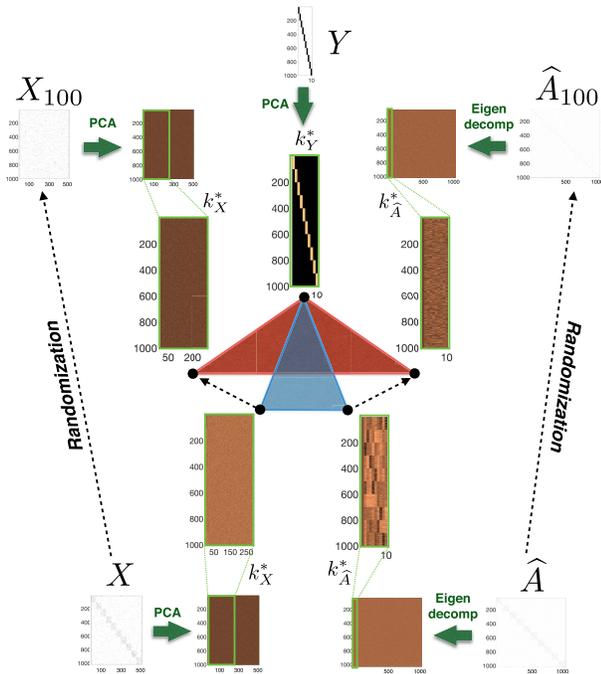


Fig. 2. Method to determine relevant subspaces [see (11)]. Using the constructive example, we illustrate the subspaces representing features, graph, and ground truth. The feature and ground-truth matrices are decomposed via PCA, and the graph matrix is similarly eigendecomposed. Fixing  $k_Y^* = F$ , we optimize (11) to find the dimensions  $k_X^*$  and  $k_{\hat{A}}^*$  that maximize the difference between the area of the blue triangle, which reflects the alignment of the three subspaces  $(X, \hat{A}, Y)$  of the original data, and the area of the red triangle, which corresponds to the alignment of the subspaces  $(X_{100}, \hat{A}_{100}, Y)$  of the fully randomized data. The edges of the triangles correspond to the pairwise chordal distances (e.g., the base of the blue triangle corresponds to  $d(X, \hat{A})$ ).

The larger the chordal distance  $d(\mathcal{A}, \mathcal{B})$ , the worse the alignment between the subspaces  $\mathcal{A}$  and  $\mathcal{B}$ .

We remark that the last inequality in  $\alpha \leq \beta < n$  is strict. If a subspace spans the whole ambient space (i.e.,  $\beta = n$ ), then its distance to all other strict subspaces of  $\mathbb{R}^n$  is trivially zero, as it is always possible to find a rotation that aligns the strict subspace with the whole space.

2) *Alignment Metric*: Our task involves establishing the alignment between three subspaces associated with the features  $X$ , the graph  $\hat{A}$ , and the ground truth  $Y$ . To do so, we consider the distance matrix containing all the pairwise chordal distances

$$D(X, \hat{A}, Y) = \begin{bmatrix} 0 & d(X, \hat{A}) & d(X, Y) \\ d(X, \hat{A}) & 0 & d(\hat{A}, Y) \\ d(X, Y) & d(\hat{A}, Y) & 0 \end{bmatrix} \quad (9)$$

and we take the Frobenius norm [28] of this matrix  $D$  as our subspace alignment measure (SAM)

$$\mathcal{S}(X, \hat{A}, Y) = \|D(X, \hat{A}, Y)\|_F = \sqrt{\sum_{i=1}^3 \sum_{j=1}^3 D_{ij}^2}. \quad (10)$$

The larger  $\|D\|_F$  is, the worse the alignment between the three subspaces. This alignment measure has a geometric interpretation related to the area of the triangle with sides  $d(X, \hat{A}), d(X, Y), d(\hat{A}, Y)$  (the smaller blue shaded triangle in Fig. 2).

3) *Determining the Dimension of the Subspaces*: The feature, graph, and ground-truth matrices  $(X, \hat{A}, Y)$  are associated with subspaces of the ambient space  $\mathbb{R}^N$ , where  $N$  is the number of nodes (or samples). These subspaces are spanned by the eigenvectors of  $\hat{A}$ , the principal components of the feature matrix  $X$ , and the principal components of the ground-truth matrix  $Y$ , respectively [29]. The dimension of the graph subspace is  $N$ ; the dimension of the feature subspace is the number of features  $C^0 < N$  (in our examples); and the dimension of the ground-truth subspace is the number of classes  $F < C^0 < N$ .

The pairwise chordal distances  $D_{ij}$  in (9) are computed from a number of minimal angles, corresponding to the smaller of the two dimensions of the subspaces being compared. Hence, the dimensions of the subspaces  $(k_X, k_{\hat{A}}, k_Y)$  need to be defined to compute the distance matrix  $D$ . Here, we are interested in finding low dimensional subspaces of features, graph, and ground truth with dimensions  $(k_X^*, k_{\hat{A}}^*, k_Y^*)$  such that they provide maximum discriminatory power between the original problem and the fully randomized (null) model. To do this, we propose the following criterion:

$$k_Y^* = F \\ (k_X^*, k_{\hat{A}}^*) = \max_{k_X, k_{\hat{A}}} (\|D(X_{100}, \hat{A}_{100}, Y)\|_F - \|D(X, \hat{A}, Y)\|_F). \quad (11)$$

We choose  $k_Y^*$  equal to the number of ground-truth classes since they are nonoverlapping [29]. Our optimization selects  $k_X^*$  and  $k_{\hat{A}}^*$  such that the difference in alignment between the original problem with no randomization ( $p_X = p_{\hat{A}} = 0$ ) and an ensemble of 100 fully randomized (feature and graph,  $p_X = p_{\hat{A}} = 100$ ) problems is maximized (see the Supplementary Material for details on the optimization scheme). This criterion maximizes the range of values that  $\|D\|_F$  can take, thus augmenting the discriminatory power of the alignment measure when finding the alignment between both data sources and the ground truth, beyond what is expected purely at random. Importantly, the reduced dimension of features and graphs are found simultaneously since our objective is to quantify the alignment (or amount of shared information) contained in the three subspaces. Our criterion effectively amounts to finding the dimensions of the subspaces that maximize a difference in the surfaces of the larger (red) and smaller (blue) shaded triangles in Fig. 2.

We provide the code to compute our proposed alignment measure at <https://github.com/haczqyf/gcn-data-alignment>.

## IV. EXPERIMENTS

### A. Data Sets

Relevant statistics of the data sets, including number of nodes and edges, dimension of feature vectors, and number of ground-truth classes, are reported in Table I.

1) *Constructive Example*: To illustrate the alignment measure in a controlled setting, we build a constructive example, consisting of 1000 nodes assigned to ten planted communities  $C_1, \dots, C_{10}$  of equal size. We then generate both a feature matrix and a graph matrix whose structures are aligned with

TABLE I  
SOME STATISTICS OF THE DATA SETS IN OUR STUDY.

Data sets	Nodes ( $N$ )	Edges	Features ( $C^0$ )	Classes ( $F$ )
Constructive	1,000	6,541	500	10
CORA	2,485	5,069	1,433	7
AMiner	2,072	4,299	500	7
Wikipedia	20,525	215,056	100	12
Wikipedia I	2,414	8,163	100	5
Wikipedia II	1,858	8,444	100	5

the ground-truth assignment matrix. The graph structure is generated using a stochastic block model that reproduces the ground-truth structure with some noise: two nodes are connected with a probability  $p_{in} = 0.07$  if they belong to the same community  $C_i$  and  $p_{out} = 0.007$  otherwise. The feature matrix is constructed in a similar way. The feature vectors are 500-dimensional and binary, i.e., a node either possesses a feature or it does not. Each ground-truth cluster is associated with 50 features that are present with a probability of  $p_{in} = 0.07$ . Each node also has a probability  $p_{out} = 0.007$  of possessing each feature characterizing other clusters. Using the same stochastic block structure for both features and graph ensures that they are maximally aligned with the ground truth. This constructive example is then randomized in a controlled way to detect the loss of alignment and the impact this loss of alignment has on the classification performance.

2) *CORA*: The CORA data set is a benchmark for classification algorithms using text and citation data.<sup>1</sup> Each article is labeled as belonging to one of seven categories (Case\_Based, Genetic\_Algorithms, Neural\_Networks, Probabilistic\_Methods, Reinforcement\_Learning, Rule\_Learning, and Theory), which gives the ground truth  $Y$ . The text of each article is described by a 0/1 vector indicating the absence/presence of words in a dictionary of 1433 unique words, the dimension of the feature space. The feature matrix  $X$  is made from these word vectors. We extracted the largest connected component of this citation graph (undirected) to form the graph adjacency matrix  $A$ .

3) *AMiner*: For additional comparisons, we produced a new data set with similar characteristics to CORA from the academic citation site AMiner. AMiner is a popular scholarly social network service for research purposes only [30], which provides an open database<sup>2</sup> with more than ten data sets encompassing researchers, conferences, and publication data. Among these, the academic social network<sup>3</sup> is the largest one and includes information on articles, citations, authors, and scientific collaborations. In 2012, the Chinese Computer Federation (CCF) released a catalog including ten subfields of computer science. Using the AMiner academic social network, Qian *et al.* [31] extracted 102887 articles published from 2010 to 2012 and mapped each article with a unique subfield of computer science according to the publication venue. Here, we use these assigned categories as the ground truth for a classification task. Using all the articles in [31] that have both abstract and references, we created a data set of similar size to CORA. We extracted the largest connected component from the citation network of all articles in seven

subfields (computer systems/high-performance computing, computer networks, network/information security, software engineering/software/programming language, databases/data mining/information retrieval, theoretical computer science, and computer graphics/multimedia) from 2010 to 2011. The resulting AMiner citation network consists of 2072 articles with 4299 edges. Just as with CORA, we treat the citations as undirected edges and obtain an adjacency matrix  $A$ . We further extracted the most frequent 500 stemmed terms from the corpus of abstracts of articles and constructed the feature matrix  $X$  for AMiner using bag-of-words.

4) *Wikipedia*: As a contrasting example, we produced three data sets from the English Wikipedia. Wikipedia provides an interlinked corpus of documents (articles) in different fields that “cite” each other via hyperlinks. We first constructed a large corpus of articles, consisting of a mixture of popular and random pages, so as to obtain a balanced data set. We retrieved the 5000 most accessed articles during the week before the construction of the data set (July 2017) and additional 20000 documents at random using the Wikipedia built-in random function.<sup>4</sup> The text and subcategories of each document, together with the names of documents connected to it, were obtained using the Python library *Wikipedia*.<sup>5</sup> A few documents (e.g., those with no subcategories) were filtered out during this process. We constructed the citation network of the documents retrieved and extracted the largest connected component. The resulting citation network contained 20525 nodes and 215056 edges. The text content of each document was converted into a bag-of-words representation based on the 100 most frequent words. To establish the ground truth, we used 12 categories from the application programming interface (API) (People, Geography, Culture, Society, History, Nature, Sports, Technology, Health, Religion, Mathematics, and Philosophy) and assigned each document to one of them. As part of our investigation, we split this large Wikipedia data set into two smaller subsets of nonoverlapping categories: Wikipedia I, consisting of Health, Mathematics, Nature, Sports, and Technology; and Wikipedia II, with the categories Culture, Geography, History, Society, and People.

### B. GCN Architecture, Hyperparameters, and Implementation

We used the GCN architecture [16] and implementation<sup>6</sup> provided by Kipf and Welling [16] and followed closely their experimental setup to train and test the GCN on our data sets. We used a two-layer GCN as described in Section II-C with the maximum number of training iterations (epochs) set to 400 [32], a learning rate of 0.01, and early stopping with a window size of 100, i.e., training stops if the validation loss does not decrease for 100 consecutive epochs. Other hyperparameters used were: 1) dropout rate: 0.5; 2) L2 regularization:  $5 \times 10^{-4}$ ; and 3) number of hidden units: 16. We initialized the weights, as described in [33], and, accordingly, row-normalized the input feature vectors. For the training, validation, and test of the GCN, we used the following split:

<sup>1</sup><https://linqs.soe.ucsc.edu/data>

<sup>2</sup><https://aminer.org/data>

<sup>3</sup><https://aminer.org/aminetwork>

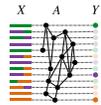
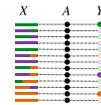
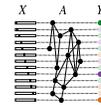
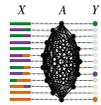
<sup>4</sup><https://en.wikipedia.org/wiki/Wikipedia:Random>

<sup>5</sup><https://github.com/goldsmith/Wikipedia>

<sup>6</sup><https://github.com/tkipf/gcn>

TABLE II

CLASSIFICATION ACCURACY OF GCN AND LIMITING CASES FOR OUR DATA SETS. THE BEST PERFORMANCE IS INDICATED IN BOLD. ERROR BARS ARE EVALUATED OVER 100 RUNS. THE GCN WITH ORIGINAL DATA PERFORMS BEST IN MOST CASES BUT IS OUTPERFORMED BY MLP IN THE FULL WIKIPEDIA DATA SET AND ITS SUBSET WIKIPEDIA II

Data sets	GCN (original)	GCN (limiting cases)		
		No graph = MLP (Only features) $A = 00^T$	No features (Only graph) (Mean field) $X = I_N$	Complete graph (Mean field) $A = 11^T - I_N$
				
Constructive	<b>0.932 ± 0.006</b>	0.416 ± 0.010	0.764 ± 0.009	0.100 ± 0.003
CORA	<b>0.811 ± 0.005</b>	0.548 ± 0.014	0.691 ± 0.006	0.121 ± 0.066
AMiner	<b>0.748 ± 0.005</b>	0.547 ± 0.013	0.591 ± 0.006	0.123 ± 0.045
Wikipedia	0.392 ± 0.010	<b>0.450 ± 0.007</b>	0.254 ± 0.037	O.O.M.
Wikipedia I	<b>0.861 ± 0.006</b>	0.796 ± 0.005	0.824 ± 0.003	0.163 ± 0.135
Wikipedia II	0.566 ± 0.021	<b>0.659 ± 0.011</b>	0.347 ± 0.012	0.155 ± 0.176

1) 5% of instances as training set; 2) 10% as validation set; and 3) the remaining 85% as test set. We used this split for all data sets with exception of the full Wikipedia data set, where we used: 1) 3.5% of instances as training set; 2) 11.5% as validation set; and 3) the remaining 85% as test set. This modification of the split was necessary to ensure that the instances in the training set were evenly distributed across categories.

## V. RESULTS

The GCN performance is evaluated using the standard classification accuracy defined as the proportion of nodes correctly classified in the test set.

### A. GCN: Original Graph Versus Limiting Cases

For each data set in Table I, we trained and tested a GCN with the original graph and features matrices, and GCN models under the three limiting cases described in Section III-B. We computed the average accuracy of 100 runs with random weight initializations (see Table II).

The GCN using all the information available in the features and the graph outperforms MLP (the no graph limit) except in the case of the large Wikipedia set. Hence, using the additional information contained in the graph does not necessarily increase the performance of GCN. To investigate this issue further, we split the Wikipedia data set into two subsets: Wikipedia I, with articles in topics that tend to be more self-referential (e.g., Mathematics or Technology), and Wikipedia II, containing pages in areas that are less self-contained (e.g., Culture or Society). We observed that GCN outperforms MLP for Wikipedia I, but the opposite is still true for Wikipedia II. Finally, we also observe that the performance of “No features” is always lower than the performance of GCN, and as expected, the performance of “Complete graph” (i.e., mean field) is very low and close to pure chance (i.e.,  $\sim 1/F$ ).

### B. Performance of GCN Under Randomization

The abovementioned results lead us to pose the hypothesis that a degree of synergy between features, graph, and ground

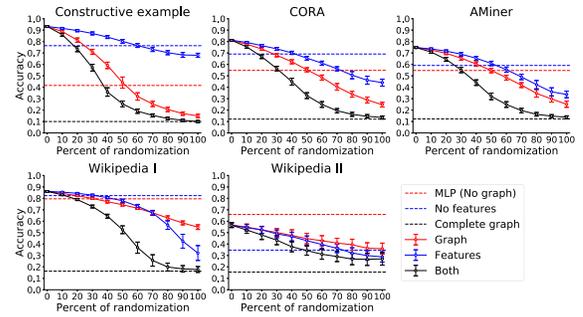


Fig. 3. Degradation of classification performance as a function of randomization. Each panel shows the degradation of the classification accuracy as a function of the randomization of graph, features, and both, for a different data set. Error bars are evaluated over 100 realizations: for zero percent randomization, we report 100 runs with random weight initializations; and for the rest, we report 1 run with random weight initializations for 100 random realizations. The horizontal lines correspond to the limiting cases in Table II. The full Wikipedia data set was not analyzed here since the eigendecomposition of  $\hat{A}$  needed to obtain  $k_X^*$ ,  $k_A^*$  is computationally intensive.

truth is needed for GCN to perform well. To investigate this hypothesis, we use the randomization schemes described in Section III-A to systematically degrade the information content of the graph and/or the features in our data sets. Fig. 3 presents the performance of the GCN as a function of the percent of randomization of the graph structure, the features, or both. As expected, the accuracy decreases for all data sets as the information contained in the graph, features, or both is scrambled, yet with differences in the decay rate of each of the ingredients for the different examples.

Note that the chance-level performance of the “Complete graph” (mean field) limiting case is achieved only when both graph and features are fully randomized, whereas the accuracy of the two other limiting cases (“No graph—MLP,” “No features”) is reached around the half-point ( $\sim 50\%$ ) of randomization of the graph or of the features, respectively. This indicates that using the scrambled information above a certain degree of randomization becomes more detrimental to the classification performance than simply ignoring it.

### C. Relating GCN Performance and Subspace Alignment

We tested whether the degradation of GCN performance is linked to the increased misalignment of features, graph, and ground truth given by the SAM

$$S^*(X, \hat{A}, Y) = \|D(X, \hat{A}, Y; k_X^*, k_A^*, k_Y^*)\|_F \quad (12)$$

which corresponds to (10) computed with the dimensions  $(k_X^*, k_A^*, k_Y^*)$  obtained using (11) (see Table III and the Supplementary Material for the optimization scheme used). Fig. 4 shows that the GCN accuracy is clearly (anti)correlated with the subspace alignment distance (12) in all our examples (mean correlation =  $-0.92$ ). As we randomize the graph and/or features, the subspace misalignment increases, and the GCN performance decreases. In addition to the Chordal distance, [26] studies other subspace distances. While all the distances can be expressed in terms of the principal angles  $\theta_j$ , some rely on all the angles whereas others only use

TABLE III  
DIMENSIONS OF THE THREE SUBSPACES OBTAINED  
ACCORDING TO (11) FOR OUR DATA SETS

Data sets	$k_X^*$	$k_{\hat{A}}$	$k_Y^*$
Constructive example	287	10	10
CORA	1,291	190	7
AMiner	500	57	7
Wikipedia I	68	1,699	5
Wikipedia II	100	1,125	5

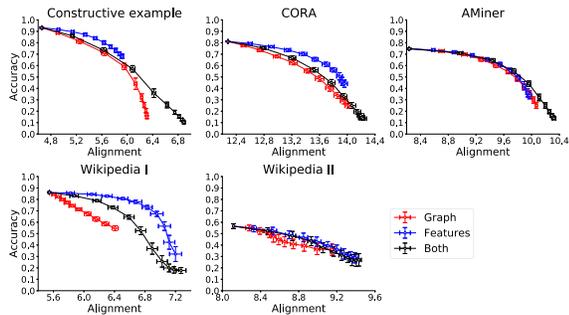


Fig. 4. Classification performance versus the SAM. Each panel shows the accuracy of GCN versus the SAM (12) for all the runs presented in Fig. 3. Error bars are evaluated over 100 randomizations.

the maximum principal angle. We obtain similar results for distances that use all the principal angles (e.g., Chordal and Grassmann), but we find that extremal distances based on the maximum principal angle (e.g., the Projection distance) do not correlate as well with GCN performance. This highlights the importance of the information captured by all principal angles to quantify the alignment between subspaces. For results based on the Grassmann and Projection distances, see Appendix (Section III) in the Supplementary Material.

## VI. DISCUSSION

Our first set of experiments (see Table II) reflects the varying amount of information that GCN can extract from features, graph, and their combination, for the purpose of classification. For a classifier to perform well, it is necessary to find (possibly nonlinear) combinations of features that map differentially and distinctively onto the categories of the ground truth. The larger the difference (or distance on the projected space) between the samples of each category, the easier it is to “separate” them, and the better the classifier. In the MLP setting, for instance, the weights between layers ( $W^\ell$ ) are trained to maximize this separation. As seen by the different accuracies in the “No graph” column (see Table II), the features of each example contain a variable amount of information that is mappable on its ground truth. Similar reasoning applies to classification based on graph information alone, but, in this case, it is the eigenvectors of  $\hat{A}$  that needs to be combined to produce distinguishing features between the categories in the ground truth (e.g., if the graph substructures across scales [34] do not map onto the separation lines of the ground-truth categories, then the classification performance based on the graph will deteriorate). The accuracy in the “No features” column indicates that some of the graphs contain more congruent information with the ground truth than others. Therefore,

the “No graph” and “No features” limiting cases inform about the relative congruence of each type of information with respect to the ground truth. One can then conjecture that, if the performance of the “No features” case is higher than the “No graph” case, GCN will yield better results than MLP.

In addition, our numerics show that, although combining both sources of information generally leads to improved classification performance (“GCN original” column in Table II), this is not always necessarily the case. Indeed, for the Wikipedia and Wikipedia II examples, the classification performance of the MLP (“No graph”), which is agnostic to relationships between samples, is better than when the additional layer of relational information about the samples (i.e., the graph) is incorporated via the GCN architecture. This suggests that, for improved GCN classification, the information contained in features and graphs needs to be constructively aligned with the ground truth. This phenomenon can be intuitively understood as follows. In the absence of a graph (i.e., the MLP setting), the training of the layer weights is done independently over the samples, without assuming any relationship between them. In GCN, on the other hand, the role of the graph is to guide the training of the weights by averaging the features of a node with those of its graph neighbors. The underlying assumption is that the relationships represented by the graph should be consistent with the information of the features, i.e., the features of nodes that are graph neighbors are expected to be more similar than otherwise; hence, the training process is biased toward convolving the diffusing information on the graph to extract improved feature descriptions for the classifier. However, if feature similarities and graph neighborhoods (or more generally, graph communities [34]) are not congruent, this graph-based averaging during the training is not beneficial.

To explore this issue in a controlled fashion, our second set of experiments (see Fig. 3) studied the degradation of the classification performance induced by the systematic randomization of graph structure and/or features. The erosion of information is not uniform across our examples, reflecting the relative salience of each of the components (features and graph) for classification. Note that the GCN is able to leverage the information present in any of the two components and is only degraded to chance-level performance when both graph and features are fully randomized. Interestingly, this fully randomized (chance-level) performance coincides with that of the “Complete graph” (or mean field) limiting case, where the classifier is trained on features averaged over all the samples, thus leading to a uniform representation that has zero discriminating power when it comes to category assignment.

These results suggest that a degree of constructive alignment between the matrices of features, graph, and ground truth ( $X, \hat{A}, Y$ ) is necessary for GCN to operate successfully beyond standard classifiers. To capture this idea, we proposed a simple SAM (12) that uses the minimal principal angles to capture the consistency of pairwise projections between subspaces. Fig. 4 shows that SAM correlates well with the classification performance and captures the monotonic dependence remarkably, given that SAM is a simple linear measure being applied to the outcome of a highly nonlinear, optimized system. The results are consistent for other versions of GCN. In particular,

in the Supplementary Material (Section II), we show that the alignment measure correlates well with the performance of the recently proposed simple graph convolution (SGC) [35].

The alignment measure can be used to evaluate the relative importance of features and graphs for classification without explicitly running the GCN, by comparing the SAM under full randomization of features against the SAM under full randomization of the graph. If  $\mathcal{S}^*(X_{100}, \widehat{A}, Y) > \mathcal{S}^*(X, \widehat{A}_{100}, Y)$ , the features play a more important role in GCN classification. Conversely, if  $\mathcal{S}^*(X_{100}, \widehat{A}, Y) < \mathcal{S}^*(X, \widehat{A}_{100}, Y)$ , the graph is more important in GCN classification. While we have focused here on node classification, it would be interesting in future work to extend our measure to other tasks, such as graph classification, link prediction, and regression.

## VII. CONCLUSION

In this article, we have introduced SAM (12), a measure that quantifies the consistency between the feature and graph ingredients of data sets, and we showed that it correlates well with the classification performance of GCNs. Our experiments show that a degree of alignment is needed for a GCN approach to be beneficial, and using a GCN can actually be detrimental to the classification performance if the feature and graph subspaces associated with the data are not constructively aligned (e.g., Wikipedia and Wikipedia II). More generally, the SAM has potentially a wider range of applications in the quantification of data alignment including, among others: quantifying the alignment of different graphs associated with, or obtained from, particular data sets; evaluating the quality of classifications found using unsupervised methods; and aiding in choosing the classifier architecture most advantageous computationally for a particular data set.

Our approach has a number of limitations that could be addressed in future work. First, it contains two parameters (i.e., the dimensions of the subspaces,  $k_X^*$  and  $k_A^*$ ) that need to be tuned through a computational search. Second, the alignment is not directly comparable across data sets since the subspace dimensions are adjusted for each data set. To facilitate comparisons across data sets, normalized versions of the alignment measure will be the object of future work. Third, the current measure is not suitable for very large data sets as the eigendecomposition of large matrices is computationally demanding. For very large data sets, approximations (e.g., using the Lanczos algorithm to explore only leading eigenvectors) might be necessary to optimize the subspace dimensions.

## REFERENCES

- [1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.
- [2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
- [3] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
- [4] 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, Jul. 2017.
- [5] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found. Trends Signal Process.*, vol. 7, nos. 3–4, pp. 197–387, Jun. 2014.
- [6] E. P. Simoncelli and B. A. Olshausen, "Natural image statistics and neural representation," *Annu. Rev. Neurosci.*, vol. 24, no. 1, pp. 1193–1216, Mar. 2001.
- [7] D. J. Field, "What the statistics of natural images tell us about visual coding," *Hum. Vis., Visual Process., Digit. Display*, vol. 1077, pp. 269–277, Aug. 1989.
- [8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [9] Y. LeCun *et al.*, "Handwritten digit recognition with a back-propagation network," in *Proc. Adv. Neural Inf. Process. Syst.*, 1990, pp. 396–404.
- [10] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1872–1886, Aug. 2013.
- [11] D. Lazer *et al.*, "Life in the network: The coming age of computational social science," *Science*, vol. 323, no. 5915, p. 721, 2009.
- [12] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–17.
- [13] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.
- [14] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 24, 2020, doi: [10.1109/TNNLS.2020.2978386](https://doi.org/10.1109/TNNLS.2020.2978386).
- [15] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.
- [16] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–7.
- [17] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2017, pp. 729–734.
- [18] Y. Li, D. Tarlow, M. Brockschmidt, and R. S. Zemel, "Gated graph sequence neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–20.
- [19] D. K. Duvenaud *et al.*, "Convolutional networks on graphs for learning molecular fingerprints," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2224–2232.
- [20] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1993–2001.
- [21] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2014–2023.
- [22] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014, pp. 1–14.
- [23] P. De Jong, C. Sprenger, and F. Van Veen, "On extreme values of moran's  $i$  and geary's  $c$ ," *Geograph. Anal.*, vol. 16, no. 1, pp. 17–24, 1984.
- [24] T. Waldhor, "Moran's spatial autocorrelation coefficient," in *Encyclopedia Statistical Science*, vol. 12, S. Kotz, N. Balakrishna, C. Read, B. Vidakovic, N. Johnson, Eds. Hoboken, NJ, USA: Wiley, 2006, pp. 7875–7878.
- [25] M. E. J. Newman, "The structure and function of complex networks," *SIAM Rev.*, vol. 45, no. 2, pp. 167–256, 2003.
- [26] K. Ye and L.-H. Lim, "Schubert varieties and distances between subspaces of different dimensions," *SIAM J. Matrix Anal. Appl.*, vol. 37, no. 3, pp. 1176–1197, Jan. 2016.
- [27] A. Björck and G. H. Golub, "Numerical methods for computing angles between linear subspaces," *Math. Comput.*, vol. 27, no. 123, pp. 579–594, 1973.
- [28] G. H. Golub and C. F. Van Loan, *Matrix Computations*, vol. 3. Baltimore, MD, USA: JHU Press, 2012.
- [29] U. von Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, Dec. 2007.
- [30] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "ArnetMiner: Extraction and mining of academic social networks," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 990–998.
- [31] Y. Qian, W. Rong, N. Jiang, J. Tang, and Z. Xiong, "Citation regression analysis of computer science publications in different ranking categories and subfields," *Scientometrics*, vol. 110, no. 3, pp. 1351–1374, Mar. 2017.
- [32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–5.

- [33] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
- [34] R. Lambiotte, J.-C. Delvenne, and M. Barahona, "Random walks, Markov processes and the multiscale modular organization of complex networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 1, no. 2, pp. 76–90, Jul. 2014.
- [35] F. Wu, T. Zhang, A. H. D. Souza, Jr., C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6861–6871.



**Yifan Qian** received the B.Sc. degree in information and computing science and the M.Sc. degree in computer science from Beihang University, Beijing, China, in 2014 and 2017, respectively. He is currently pursuing the Ph.D. degree with the Queen Mary University of London, London, U.K.

His research interest is broadly concerned with computational social science and combines theories and methods from network science, sociology, machine learning, and data science.



**Paul Expert** received the Ph.D. degree in physics from Imperial College London, London, U.K., in 2012.

He was in Neuroimaging at King's College London, London, and Mathematics at Imperial College London. He is currently a Research Associate with the Global Digital Health Unit, Imperial College London, and a Visiting Associate Professor with the Tokyo Institute of Technology, Tokyo, Japan. His research interest is concerned with understanding the interaction between the structure and function of complex systems, with applications ranging from neuroscience to public health.



**Tom Rieu** received the M.Sc. degree in Engineering from the Engineering School CentraleSupélec Paris, France, and the M.Sc. degree in applied mathematics from the Imperial College London, London, U.K., both in 2017.

He is currently a Data Scientist with Facebook, London, U.K. His academic research was focused on machine learning and particularly the application of deep models to networks. Currently, his work as part of the Facebook Product team is concerned with producing data science insights to drive improvements on the core and business products of Facebook's online advertising platform and retargeting technology.



**Pietro Panzarasa** received the Ph.D. degree from Bocconi University Milan, Italy, in 2000.

He is a Professor of Networks and Innovation with the School of Business and Management, Queen Mary University of London, London, U.K. He became a Research Fellow with the University of Southampton, Southampton, U.K. He also held visiting positions at Columbia University, New York, NY, USA, and Carnegie Mellon University, Pittsburgh, PA, USA. He draws on network science, computational social science, and big data analytics to study social capital and dynamics of social interaction in complex large-scale networks.



**Mauricio Barahona** received the Ph.D. degree from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 1996.

He is a Professor with the Department of Mathematics and the Director of the EPSRC Centre for Mathematics of Precision Healthcare, Imperial College London, London, U.K. He held Fellowships with Stanford University, Stanford, CA, USA, and the California Institute of Technology, Pasadena, CA. He is broadly interested in applied mathematics in engineering, physical, social, and biological systems using methods from graph theory, stochastic processes, dynamical systems, and machine learning.