Differentiable Graph Module (DGM) for Graph Convolutional Networks

Anees Kazi [*1] Luca Cosmo [*2 3] Seyed-Ahmad Ahmadi [4] Nassir Navab [1 5] Michael Bronstein [3 6 7]

Abstract

Graph deep learning has recently emerged as a powerful ML concept allowing to generalize successful deep neural architectures to non-Euclidean structured data. Such methods have shown promising results on a broad spectrum of applications ranging from social science, biomedicine, and particle physics to computer vision, graphics, and chemistry. One of the limitations of the majority of current graph neural network architectures is that they are often restricted to the transductive setting and rely on the assumption that the underlying graph is *known* and *fixed*. In many settings, such as those arising in medical and healthcare applications, this assumption is not necessarily true since the graph may be noisy, partially- or even completely unknown, and one is thus interested in inferring it from the data. This is especially important in inductive settings when dealing with nodes not present in the graph at training time. In this paper, we introduce Differentiable Graph Module (DGM), a learnable function predicting the edge probability in the graph relevant for the task, that can be combined with convolutional graph neural network layers and trained in an end-to-end fashion. We provide an extensive evaluation of applications from the domains of healthcare (disease prediction) and brain imaging (age prediction). We show that our model provides a significant improvement over baselines both in transductive and inductive settings and achieves state-of-the-art results.

1. Introduction

Geometric deep learning (GDL) is a novel emerging branch of deep learning attempting to generalize deep neural networks to non-Euclidean structured data such as graphs and manifolds (Bronstein et al., 2017; Hamilton et al., 2017; Battaglia et al., 2018). Graphs in particular, being general abstract descriptions of relation and interaction systems, are ubiquitous in different branches of science. Graph-based learning models have been successfully applied in social sciences (Zhang & Chen, 2018; Qi et al., 2018), computer vision and graphics (Qi et al., 2017; Monti et al., 2017; Wang et al., 2019), medical, and biological (Parisot et al., 2018; 2017; Mellema et al., 2019; Kazi et al., 2019b; Zitnik et al., 2018; 2019; Gainza et al., 2019) sciences. Graph Neural Networks (GNNs) are a popular approach for learning on graphs. Today's wide variety of GNN architectures includes spectral (Bruna et al., 2013) and spectral-like (Defferrard et al., 2016; Kipf & Welling, 2016; Levie et al., 2018; Bianchi et al., 2019) methods, local charting (Monti et al., 2017), and attention (Veličković et al., 2017; Kondor, 2018; Bruna & Li, 2017; Monti et al., 2018). Battaglia et al. (2018) showed that most GNNs can be formulated in terms of message passing (Gilmer et al.).

A notable drawback of most GNN architectures is the assumption that the underlying graph is *given* and *fixed*, while graph convolution-like operations typically amount to modifying the node-wise features. Architectures like message passing neural networks (Gilmer et al.) or primal-dual convolutions (Monti et al., 2018) also allow to update the edge features, but the graph *topology* is always kept the same. This often happens to be a limiting assumption. In many problems, the data can be assumed to have some underlying graph structure, however, the graph itself might not be explicitly given (Liu et al., 2012), a setting we refer to as *latent graph*. This is the case, for example, in medical and healthcare applications, where the graph may be noisy, partially- or even completely unknown, and one is thus interested in inferring it from the data. This is especially important in inductive settings where some nodes might be present in the graph at testing but not training. Furthermore, sometimes the graph may be even more important than the downstream task as it conveys some interpretability of the model.

Graph topology inference is a long-standing problem that

[*]Equal contribution [1]Computer Aided Medical Procedures (CAMP), Technical University of Munich, Germany [2]Sapienza University of Rome, Italy [3]University of Lugano, Switzerland [4]German Center for Vertigo and Balance Disorders, Ludwig Maximilians Universität München, Germany [5]Whiting School of Engineering, Johns Hopkins University, Baltimore, USA [6]Imperial College London, UK [7]Twitter, UK. Correspondence to: Anees Kazi <anees.kazi@tum.de>.

Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020. Copyright 2020 by the author(s).

was addressed using signal processing techniques (Dong et al., 2019; Mateos et al., 2019). In the machine learning literature, several models dealing with latent graphs have recently been proposed (Li et al., 2018; Huang et al., 2018; Jiang et al., 2019). Wang et al. (Wang et al., 2019) proposed dynamic graph CNNs (DGCNN) for the analysis of point clouds, where a KNN graph is constructed on the fly in the feature space of the neural network. Franceschi et al. (2019) formulated graph learning as a bilevel optimization problem, by modeling the graph as a hyper-parameter and optimizing it with a separate loss. This method is transductive and does not scale.

Main contributions We propose a general deep learning model that simultaneously learns the graph and graph convolutional filters on it. We study several settings of our model, including a continuous and discrete differentiable graph construction, and show how to optimize it. We show that previous methods for learning the graph can be considered as particular settings of our model. We provide extensive ablation studies of our model and evaluate it for applications from the domains of healthcare and brain imaging (disease and age prediction). Our model shows significant improvement over baselines and achieves state-of-the-art results both in transductive and inductive settings.

2. Background

Given a set of N data points of dimension d, denoted $\mathbf{X} \in \mathbb{R}^{N \times d}$, a common problem in machine learning is to produce a representation that is aware of the underlying structure of the data. Such structure can be represented as a (weighted) graph $\mathcal{G} = (\mathcal{V}, \mathbf{A})$ where $\mathcal{V} = \{1, \dots, n\}$ is the vertex set and $\mathbf{A} = (a_{ij})$ is a (weighted) adjacency matrix.

Graph neural networks. Assuming this structure is provided together with the data, we have a node-attributed graph $\mathcal{G} = (\mathcal{V}, \mathbf{A}, \mathbf{X})$, on which a graph neural network (GNN) can be applied. GNN attempts to find an *embedding* $\mathbf{Z} = f_{\mathbf{\Theta}}(\mathbf{X}, \mathbf{A})$ by doing message passing (Gilmer et al.; Battaglia et al., 2018)

$$\mathbf{z}_i = \sum_{j \in \mathcal{N}_i} h_{\mathbf{\Theta}}(\mathbf{x}_i, \mathbf{x}_j, a_{ij}) \tag{1}$$

in a local neighborhood $\mathcal{N}_i = \{j : (i,j) \in \mathcal{E}\}$ of the node. Here $h_{\mathbf{\Theta}}$ denotes a learnable function shared across nodes, whose parameters $\mathbf{\Theta}$ are chosen to minimize a downstream loss. Equation (1) is also referred to as *edge convolution* (EC) (Wang et al., 2019) due to its generalization of the classical convolution operation on grids. A particular case of (1) with node-wise linear transformation $h_{\mathbf{\Theta}} = a_{ij}\mathbf{\Theta}\mathbf{x}_j$ by matrix $\mathbf{\Theta}$, is called *graph convolution* (GC) (Defferrard et al., 2016; Kipf & Welling, 2016).

Latent graphs. We are interested in the setting when the underlying graph is *unknown*, and try to learn it. Learning the graph serves two purposes: First, it is used to represent the structure of the data. Second, it is used as the support for graph-based convolutions to obtain the embeddings of the data points. The main obstacle for including the graph construction as a part of the deep learning pipeline is that it is a discrete structure, and as such non-differentiable. The most related approach to this paper is Wang et al. (2019). In their method the graph convolutional filters and the layer activations are optimised towards the downstream classification and segmentation tasks. The graph, however, is constructed ad-hoc as a KNN graph on the activations after each layer, without a dedicated loss. As such, the graph is built dynamically but not learned, and the underlying latent graph of the domain is not recovered. Our method, described in the next section, aims at addressing these issues.

3. Method

3.1. Architecture

We propose a general technique i) to learn the graph based on the output features of each layer and ii) to optimize these graphs along with the network parameters during the training. Our architecture comprises two main blocks, the **Differentiable Graph Module (DGM)** and **Diffusion Module**, which are shown in Figure 1 and detailed in the following.

Differentiable Graph Module: The DGM is tasked with building the (weighted) graph representing the input space. It takes the feature matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$ (and optionally an initial graph $\mathcal{G}_0$) as input, and yields a graph $\mathcal{G}$ as output. Since the node set $\mathcal{V}$ is fixed, the two graphs can be represented by their adjacency matrices $\mathbf{A}_0$, $\mathbf{A}$.

The input features $\mathbf{X} \in \mathbb{R}^{N \times d}$ are first transformed into *auxiliary features* $\hat{\mathbf{X}} = f_{\mathbf{\Theta}}(\mathbf{X}) \in \mathbb{R}^{N \times \hat{d}}$ by means of a parametric function $f_{\mathbf{\Theta}}$ typically reducing the input dimension ($\hat{d} \ll d$). If the initial graph $\mathcal{G}_0$ is provided, we can use $f_{\mathbf{\Theta}}$ of the general form (1), where new features $\hat{\mathbf{X}}$ are computed by edge- or graph-convolution on $\mathcal{G}_0$. Otherwise, $f_{\mathbf{\Theta}}$ is applied to each node feature independently, acting row-wise on the matrix $\mathbf{X}$.

Second, the auxiliary features $\hat{\mathbf{X}}$ are used for graph construction. We define the edge probability $p_{ij}(\mathbf{X}; \mathbf{\Theta}, t) = e^{-t\|\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j\|_2^2} = e^{-t\|f_{\mathbf{\Theta}}(\mathbf{x}_i) - f_{\mathbf{\Theta}}(\mathbf{x}_j)\|_2^2}$, where t is also a learnable parameter. Our choice of using a Euclidean metric for defining the edge probability is for the sake of simplicity, and other metrics, e.g. hyperbolic (Krioukov et al., 2010; Nickel & Kiela, 2017), could also be used. A straightforward way to derive a graph $\mathcal{G}$ is to transform the probability matrix $\mathbf{P}(\mathbf{X}; \mathbf{\Theta}, t)$ into a weighted adjacency matrix, e.g. by soft-thresholding the distances $\|\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j\|$ using the sigmoid function $a_{ij} = 1/(1 + p_{ij}e^{tT})$, where T denotes the threshold. In this way, the graph is represented by the adjacency matrix $\mathbf{A}(\mathbf{X}; \mathbf{\Theta}, t, T)$ parametrized through $\mathbf{\Theta}, t$ and

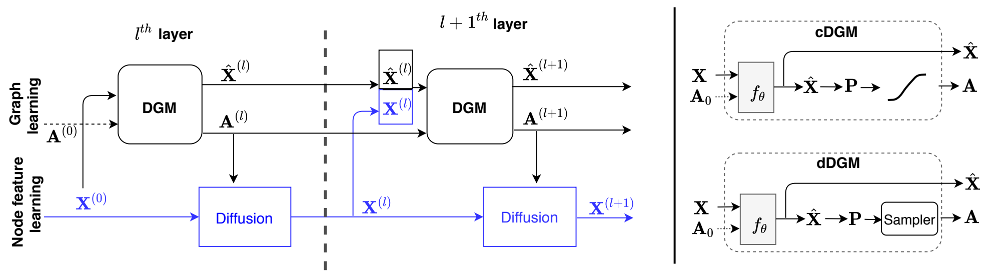

Figure 1. Left: Two-layered architecture including Differentiable Graph Module (DGM) that learns the graph, and Diffusion Module that uses the graph convolutional filters. *Right:* Details of DGM in its two variants, cDGM and dDGM.

the additional parameter T, and is differentiable w.r.t. these parameters. We refer to this variant of our architecture as *continuous DGM (cDGM)*.

One of the possible shortcomings of cDGM is that it may produce dense adjacency matrix, i.e. a fully connected graph with many edges having near-zero weight. As an efficient alternative, we can produce a sparse k-degree graph by using the *Gumbel-Top-k trick* (Kool et al., 2019) to sample edges from the probability $\mathbf{P}(\mathbf{X};\mathbf{\Theta},t)$. Such sampling can be regarded as a stochastic relaxation of the KNN rule. For each node i, we extract k edges $(i,j_{i,1}),\ldots,(i,j_{i,k})$ as the first k elements of $\operatorname{argsort}(\log(\mathbf{p}_i)-\log(-\log(\mathbf{q}))$, where $\mathbf{q}\in\mathbb{R}^N$ is uniform i.i.d. in the interval $[0,1]$. Samples extracted this way follow the categorical distribution $p_{ij}/\sum_r p_{ir}$ (Kool et al., 2019). We define the edge set of the sparse graph $\mathcal{G}$ constructed this way as $\mathcal{E}(\mathbf{X};\mathbf{\Theta},t)=\{(i,j_{i,1}),\ldots,(i,j_{i,k}):i=1,\ldots,N\}$ and represent it by the unweighted adjacency matrix $\mathbf{A}(\mathbf{X};\mathbf{\Theta},t)$. The key advantage is that this matrix is sparse. We show next how to efficiently learn the parameters $\mathbf{\Theta},t$. We refer to this variant of our architecture as *discrete DGM (dDGM)*. Note that, since the dDGM graph sampling scheme is stochastic, the prediction of the network at inference time is not deterministic. We can actually take advantage of this, and implement a consensus scheme over several samplings.

Diffusion Module: This module takes the graph $\mathcal{G}$ produced by the DGM and the features $\mathbf{X}$ as inputs, and yields a new set of features $\mathbf{X}'=g_{\mathbf{\Phi}}(\mathbf{X})$ as output. Here, $g_{\mathbf{\Phi}}$ represents a general function of the form (1); in our experiments, it is either edge- or graph-convolution on $\mathcal{G}$.

Combined model: We use a multi-layer network where each layer, numbered as $l=1,\ldots,L$, comprises a **DGM** and **Diffusion Module**, as shown in Figure 1. The lth layer produces the output

$$\mathbf{X}^{(l+1)}=g_{\mathbf{\Phi}^{(l)}}(\mathbf{A}^{(l+1)},\mathbf{X}^{(l)}) \tag{2}$$

where $\mathbf{A}^{(l+1)}\sim\mathbf{P}^{(l)}(\hat{\mathbf{X}}^{(l+1)})$ is the sampled graph and $\hat{\mathbf{X}}^{(l+1)}=f_{\mathbf{\Theta}}^{(l+1)}([\mathbf{X}^{(l)}\,|\,\hat{\mathbf{X}}^{(l)}],\mathbf{A}^{(l)})$ are the *auxiliary features*. We assume $\mathbf{X}^{(0)}=\mathbf{X}$ and unless some initial knowledge of the structure of the data is available, $\mathbf{A}^{(0)}=\mathbf{I}$ (i.e., the initial graph is $\mathcal{G}^{(0)}=(\mathcal{V},\emptyset)$) and $f_{\mathbf{\Theta}}^{(0)}$ is a node-wise function (MLP). Depending on the task, the final node features $\mathbf{X}^{(L)}$ of the last layer L can then be given as input to a MLP to obtain the final node predictions

DGCNN can be obtained as a particular setting of our model with $f_{\mathbf{\Theta}}=\text{id}$ in the **DGM** module and using edge convolution in the **Diffusion** module.

3.2. Training

The sampling scheme we adopt in dDGM does not allow the gradient of the downstream classification loss function to flow through the graph prediction branch of our network, as it involves only graph features $\hat{\mathbf{X}}$. To allow its optimization, we exploit tools from reinforcement learning (Williams, 1992), rewarding edges involved in a correct classification and penalizing edges that led to misclassification. Let $\mathbf{y}$ denote the vector of node-wise labels predicted by our model and $\tilde{\mathbf{y}}$ the groundtruth labels. We define the reward function $\delta(y_i,\tilde{y}_i)$ taking value -1 if $y_i=\tilde{y}_i$ and 1 otherwise. We derive the graph loss as

$$L_{\text{graph}}=\sum_{i=1}^{N}\delta(y_i,\tilde{y}_i)\sum_{l=1}^{L}\sum_{j:(i,j)\in\mathcal{E}^{(l)}}\log\, p_{ij}^{(l)}(\mathbf{\Theta}^{(l)}) \tag{3}$$

whose gradient approximates the gradient of the expectation $\mathbb{E}_{(\mathcal{G}^{(1)},\ldots,\mathcal{G}^{(L)})\sim(\mathbf{P}(\mathbf{\Theta}^{(1)}),\ldots,\mathbf{P}(\mathbf{\Theta}^{(L)}))}\sum_i\delta(y_i,\tilde{y}_i)$ with respect to the parameters of the graphs in all the layers. Graph loss L_{graph} is then optimized by adding it to the classification loss.

4. Experiments and Results

We test our method on two applications from the domains of healthcare (disease and age prediction), and provide an ablation study of different configurations of our architecture.

Healthcare and Brain imaging applications We use two datasets: *Tadpole* (Marinescu et al., 2018) consists of 564 patients each with a 354 dimensional representation

Table 1. Classification accuracy in % on Tadpole in the transductive setting. The top three performance scores are highlighted in color as: **First**, **Second**, **Third**.

Method	Accuracy
Linear classifier	70.22±6.32
Multi-GCN (Kazi et al., 2019a)	76.06±0.72
Spectral-GCN (Parisot et al., 2017)	81.00±6.40
InceptionGCN (Kazi et al., 2019b)	84.11±4.50
DGCNN (Wang et al., 2019)	84.59±4.33
LDS (Franceschi et al., 2019)	**87.06±3.67**
cDGM	**92.91±2.50**
dDGM	**94.14±2.12**

derived from imaging (MRI, fMRI, PET) and non-imaging (demographics and genotypes) features. The task is to classify each patient as 'Normal Control', 'Alzheimer's Disease' and 'Mild Cognitive Impairment'. *UK Biobank* (Miller et al., 2016) consists of 14,503 individuals, each equipped with a 440 dimensional feature derived from brain MRI and fMRI imaging. The task is to classify the age group of the patient (50-59,60-69, 70-79, and 80-89). Both tasks are made either transductive or inductive, where in the former setting all the nodes are given during training but the labels of the test nodes are withheld, while in the latter setting, test nodes are completely removed from the population during training, and reintroduced only during testing.

Previous methods (Parisot et al., 2017; Kazi et al., 2019a;b;c) used GNNs with hand-crafted patient population graphs built from non-imaging meta-features like the age and sex of patients. Our method allows to learn the graph directly from the input patients features, avoiding any handcrafted construction. We use the following baselines: simple linear classifier as a non-graph method; Multi-GCN (Kazi et al., 2019a), Spectral-GCN (Parisot et al., 2017), InceptionGCN (Kazi et al., 2019b) as graph methods with hand-crafted graph; and DGCNN (Wang et al., 2019) and LDS (Franceschi et al., 2019) as methods that learn the graph. We note that LDS does not support inductive learning. Results are reported in Tables 1 and 3 using 10-fold cross validation. Our model significantly outperforms the state-of-the-art on all the tasks.

Ablation study We perform a detailed ablation study of different configurations of our architecture, in particular, the choice of functions f and g (identity, node-wise (MLP), graph convolution (GC), edge convolution (EC)) and the graph construction strategy (cDGM and dDGM), using the Tadpole dataset in the transductive setting.

We use two convolutional layers with output size of 16 and a final linear layer of size 16 for classification. Results are shown in Table 2. We observe that with $f = \text{id}$ we obtain the DGCNN, here a kNN selection based graph is computed dynamically only on the node feature representation space during each epoch; this setting is significantly inferior to

Table 2. Ablation study on Tadpole transductive task. Shown is classification accuracy for different architectural choices. Notation f+g refers to the choice of the DGM and Diffusion modules (GC: graph convolution, EC: edge convolution; I: identity, MLP: multilayer perceptron). *Configuration equivalent to DGCNN.

Method	cDGM	dDGM
I + EC	—	84.27±4.20*
MLP + GC	92.42±3.82	93.47±3.82
GC + GC	90.68±4.58	**94.09±1.81**
MLP + EC	92.29±4.18	93.27±3.20
GC + EC	91.78±3.21	**94.14±2.12**

the use of graph-based convolution. The best combination is using graph convolution for both f and g.

5. Discussion and conclusion

In this paper, we tackled the challenge of graph learning in convolutional graph neural networks. We have proposed a novel Differentiable Graph Module (DGM) that predicts a probabilistic graph, allowing a discrete graph to be sampled in order to be used in any graph convolutional operator. DGM is generic and adaptable to any graph convolution based method. We prove this by an etensive ablation study and by using it to solve a wide variety of tasks in healthcare (disease prediction and age prediction) in both transductive and inductive settings.

There are some open questions with the proposed method. Our method, despite being computationally more lightweight than existing approaches (e.g. (Jang et al., 2019)), still has a quadratic complexity with respect to the number of input nodes, as it requires the computation of all pairwise distances. Restricting the computation of probabilities in a neighborhood of the node and using a tree-based algorithm could help in reducing the complexity to $\mathcal{O}(n \log n)$. Further, our choice of sampling k neighbors does not consider the heterogeneity of the graph in terms of the degree distribution of nodes. Other sampling schemes (e.g. threshold-based sampling (Jang et al., 2019)) could be investigated. It would be also interesting to take into consideration previous knowledge about the graph, e.g. by imposing a node degree distribution, or providing an initial input graph to be optimized for a specific task.

Table 3. Classification accuracy for disease and age prediction tasks in the *transductive* and *inductive* settings on the Tadpole and UK Biobank datasets. †Does not support inductive setting.

Method	TADPOLE		UK Biobank	
	Transductive	Inductive	Transductive	Inductive
DGCNN	84.59±4.33	**82.99±4.91**	58.35±0.91	**51.84±8.16**
LDS	**87.06±3.67**	†	OOM	†
cDGM	**92.91±2.50**	**91.85±2.62**	**61.32±1.51**	**55.91±3.49**
dDGM	**94.10±2.12**	**92.17±3.65**	**63.22±1.12**	**57.34±5.32**

6. Acknowledgements

The study was carried out with financial support of TUM-ICL incentive funding, Freunde und Förderer der Augenklinik, München, Germany and ERC Consolidator grant No. 724228 (LEMAN) and German Federal Ministry of Education and Health (BMBF) in connection with the foundation of the German Center for Vertigo and Balance Disorders (DSGZ) [grant number 01 EO0901]. The UK Biobank data is used under the application id 51541.

References

Battaglia, P. W. et al. Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261*, 2018.

Bianchi, F. M., Grattarola, D., Alippi, C., and Livi, L. Graph neural networks with convolutional arma filters. *arXiv:1901.01343*, 2019.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. Geometric deep learning: going beyond euclidean data. *IEEE Signal Process. Mag.*, 34(4):18–42, 2017.

Bruna, J. and Li, X. Community detection with graph neural networks. *Stat*, 1050:27, 2017.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, pp. 3844–3852, 2016.

Dong, X., Thanou, D., Rabbat, M., and Frossard, P. Learning graphs from data: A signal representation perspective. *IEEE Signal Processing Magazine*, 36(3):44–63, 2019.

Franceschi, L., Niepert, M., Pontil, M., and He, X. Learning discrete structures for graph neural networks. 2019.

Gainza, P., Sverrisson, F., Monti, F., Rodola, E., Bronstein, M. M., and Correia, B. E. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*, 17:184–192, 2019.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *Proc. ICML*.

Hamilton, W. L., Ying, R., and Leskovec, J. Representation learning on graphs: Methods and applications. *arXiv:1709.05584*, 2017.

Huang, W., Zhang, T., Rong, Y., and Huang, J. Adaptive sampling towards fast graph representation learning. In *NeurIPS*, pp. 4558–4567, 2018.

Jang, S., Moon, S., and Lee, J. Brain signal classification via learning connectivity structure. *CoRR*, abs/1905.11678, 2019. URL http://arxiv.org/abs/1905.11678.

Jiang, B., Zhang, Z., Lin, D., Tang, J., and Luo, B. Semi-supervised learning with graph learning-convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11313–11320, 2019.

Kazi, A., Shekarforoush, S., Kortuem, K., Albarqouni, S., Navab, N., et al. Self-attention equipped graph convolutions for disease prediction. In *ISBI*, pp. 1896–1899. IEEE, 2019a.

Kazi, A., Shekarforoush, S., Krishna, S. A., Burwinkel, H., Vivar, G., Kortüm, K., Ahmadi, S.-A., Albarqouni, S., and Navab, N. Inceptiongcn: Receptive field aware graph convolutional network for disease prediction. In *IPMI*. Springer, 2019b.

Kazi, A., Shekarforoush, S., Krishna, S. A., Burwinkel, H., Vivar, G., Wiestler, B., Kortüm, K., Ahmadi, S.-A., Albarqouni, S., and Navab, N. Graph convolution based attention model for personalized disease prediction. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 122–130. Springer, 2019c.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Kondor, R. N-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. *arXiv:1803.01588*, 2018.

Kool, W., van Hoof, H., and Welling, M. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. *CoRR*, abs/1903.06059, 2019. URL http://arxiv.org/abs/1903.06059.

Krioukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., and Boguná, M. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010.

Levie, R., Monti, F., Bresson, X., and Bronstein, M. M. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Trans. Signal Process.*, 67(1):97–109, 2018.

Li, R., Wang, S., Zhu, F., and Huang, J. Adaptive graph convolutional neural networks. In *AAAI*, 2018.

Liu, W., Wang, J., and Chang, S.-F. Robust and scalable graph-based semisupervised learning. *Proc. IEEE*, 100 (9):2624–2638, 2012.

Marinescu, R. V., Oxtoby, N. P., Young, A. L., Bron, E. E., Toga, A. W., Weiner, M. W., Barkhof, F., Fox, N. C., Klein, S., Alexander, D. C., et al. Tadpole challenge: Prediction of longitudinal evolution in alzheimer's disease. *arXiv preprint arXiv:1805.03909*, 2018.

Mateos, G., Segarra, S., Marques, A. G., and Ribeiro, A. Connecting the dots: Identifying network structure via graph signal processing. *IEEE Signal Processing Magazine*, 36(3):16–43, 2019.

Mellema, C., Treacher, A., Nguyen, K., and Montillo, A. Multiple deep learning architectures achieve superior performance diagnosing autism spectrum disorder using features previously extracted from structural and functional mri. In *2019 IEEE ISBI)*, pp. 1891–1895. IEEE, 2019.

Miller, K. L., Alfaro-Almagro, F., Bangerter, N. K., Thomas, D. L., Yacoub, E., Xu, J., Bartsch, A. J., Jbabdi, S., Sotiropoulos, S. N., Andersson, J. L., et al. Multimodal population brain imaging in the uk biobank prospective epidemiological study. *Nature neuroscience*, 19(11):1523, 2016.

Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., and Bronstein, M. M. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proc. CVPR*, 2017.

Monti, F. et al. Dual-primal graph convolutional networks. *arXiv:1806.00770*, 2018.

Nickel, M. and Kiela, D. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pp. 6338–6347, 2017.

Parisot, S., Ktena, S. I., Ferrante, E., Lee, M., Moreno, R. G., Glocker, B., and Rueckert, D. Spectral graph convolutions for population-based disease prediction. In *MICCAI*, pp. 177–185. Springer, 2017.

Parisot, S., Ktena, S. I., Ferrante, E., Lee, M., Guerrero, R., Glocker, B., and Rueckert, D. Disease prediction using graph convolutional networks: Application to autism spectrum disorder and alzheimer's disease. *Med Image Anal*, 48:117–130, 2018.

Qi, S., Wang, W., Jia, B., Shen, J., and Zhu, S.-C. Learning human-object interactions by graph parsing neural networks. In *ECCV*, pp. 401–417, 2018.

Qi, X., Liao, R., Jia, J., Fidler, S., and Urtasun, R. 3d graph neural networks for rgbd semantic segmentation. In *ICCV*, pp. 5199–5208, 2017.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. *arXiv:1710.10903*, 2017.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., and Solomon, J. M. Dynamic graph cnn for learning on point clouds. *ACM TOG*, 38(5):146, 2019.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Zhang, M. and Chen, Y. Link prediction based on graph neural networks. In *Proc. NeurIPS*, 2018.

Zitnik, M., Agrawal, M., and Leskovec, J. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466, 2018.

Zitnik, M., Nguyen, F., Wang, B., Leskovec, J., Goldenberg, A., and Hoffman, M. M. Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities. *Information Fusion*, 50:71–91, 2019.

7. Supplementary material

We show an ablation study for the time in table 4 and report the training and testing times of our model for different dataset size, which shows that it is on par with DGCNN and about three orders of magnitude faster than LDS. All the setting for the training are kept same as the ablation test in the paper with two convolutional layers with output size of 16 and a final linear layer of size 16 for classification.

Table 4. Scalability: training and test iteration times for different number of nodes.

Training iteration			
Method	n =564	5k	10k
DGCNN(Wang et al., 2019)	6.99ms	28.2ms	104ms
LDS (Franceschi et al., 2019)	1.84s	>1m	>1m
cDGN	7.35ms	47.8ms	211ms
dDGN	8.29ms	37.0ms	141ms

Test iteration			
Method	n =564	5k	10k
DGCNN(Wang et al., 2019)	4.60ms	25.2ms	102ms
LDS (Franceschi et al., 2019)	1.84s	>1m	>1m
cDGN	3.02ms	15.1ms	51ms
dDGN	3.97ms	24.6ms	104ms