BAYESIAN GRAPH CONVOLUTIONAL NEURAL NETWORKS USING NON-PARAMETRIC GRAPH LEARNING

Soumyasundar Pal, Florence Regol & Mark Coates
Department of Electrical and Computer Engineering,
McGill University, Montréal, Québec, Canada.
{soumyasundar.pal,florence.robert-regol}@mail.mcgill.ca, mark.coates@mcgill.ca

ABSTRACT

Graph convolutional neural networks (GCNN) have been successfully applied to many different graph based learning tasks including node and graph classification, matrix completion, and learning of node embeddings. Despite their impressive performance, the techniques have a limited capability to incorporate the uncertainty in the underlined graph structure. In order to address this issue, a Bayesian GCNN (BGCN) framework was recently proposed. In this framework, the observed graph is considered to be a random realization from a parametric random graph model and the joint Bayesian inference of the graph and GCNN weights is performed. In this paper, we propose a non-parametric generative model for graphs and incorporate it within the BGCN framework. In addition to the observed graph, our approach effectively uses the node features and training labels in the posterior inference of graphs and attains superior or comparable performance in benchmark node classification tasks.

1 INTRODUCTION

Application of convolutional neural networks to the analysis of data with an underlying graph structure has been an active area of research in recent years. Earlier works towards the development of Graph Convolutional Neural Networks (GCNNs) include (Bruna et al., 2013; Henaff et al., 2015; Duvenaud et al., 2015). (Defferrard et al., 2016) introduced an approach based on spectral filtering, which was adapted in subsequent works (Levie et al., 2019; Chen et al., 2018b; Kipf & Welling, 2017). On the other hand, spatial filtering or aggregation strategies are considered in (Atwood & Towsley, 2016; Hamilton et al., 2017). (Monti et al., 2017) present a general framework for applying neural networks on graphs and manifolds, which encompasses many existing approaches.

Several modifications have been proposed in the literature to improve the performance of GCNNs. These include incorporating attention nodes (Veličković et al., 2018), gates (Li et al., 2016; Bresson & Laurent, 2017), edge conditioning and skip connections (Sukhbaatar et al., 2016; Simonovsky & Komodakis, 2017). Other approaches consider an ensemble of graphs (Anirudh & Thiagarajan, 2017), multiple adjacency matrices (P. Such et al., 2017), the dual graph (Monti et al., 2018) and random perturbation of the graph (Sun et al., 2019). Scalable training for large networks can be achieved through neighbour sampling (Hamilton et al., 2017), performing importance sampling (Chen et al., 2018b) or using control variate based stochastic approximation (Chen et al., 2018a).

Most existing approaches process the graph as if it represents the true relationship between nodes. However, in many cases the graphs employed in applications are themselves derived from noisy data or inaccurate modelling assumptions. The presence of spurious edges or the absence of edges between nodes with very strong relationships in these noisy graphs can affect learning adversely. This can be addressed to some extent by attention mechanisms (Veličković et al., 2018) or generating an ensemble of multiple graphs by erasing some edges (Anirudh & Thiagarajan, 2017), but these approaches do not consider creating any edges that were not present in the observed graph.

In order to account for the uncertainty in the graph structure, (Zhang et al., 2019) present a Bayesian framework where the observed graph is viewed as a random sample from a collection described by a parametric random graph model. This permits joint inference of the graph and the GCNN weights. This technique significantly outperforms the state-of-the-art algorithms when only a limited amount of training labels is available. While the approach is effective, choosing an appropriate random graph model is very important and the correct choice can vary greatly for different problems and datasets. Another significant drawback of the technique is that the posterior inference of the graph is carried out solely conditioned on the observed graph. As a result any information provided by the node features and the training labels is completely disregarded. This can be highly undesirable in scenarios where the features and labels are highly correlated with the true graph connectivity.

In this paper, we propose an alternative approach which formulates the posterior inference of the graph in a non-parametric fashion, conditioned on the observed graph, features and training labels. Experimental results show that our approach obtains impressive performance for the semi-supervised node classification task with a limited number of training labels.

The rest of the paper is organized as follows. We review the GCNN in Section 2 and present the proposed approach in Section 3. The numerical experiments are described and the results are discussed in Section 4. The concluding remarks are summarized in Section 5.

2 Graph convolutional neural networks (GCNNs)

Although graph convolutional neural networks have been applied in a variety of inference tasks, here we consider the node classification problem in a graph for conciseness. In this setting, we have access to an observed graph $\mathcal{G}_{obs} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of N nodes and $\mathcal{E}$ denotes the set of edges. For every node i we observe a feature vector $\boldsymbol{x}_i \in \mathbf{R}^{d\times 1}$, but the label $\boldsymbol{y}_i$ is known for only a subset of the nodes $\mathcal{L} \subset \mathcal{V}$. The goal is to infer the labels of the remaining nodes using the information provided by the observed graph $\mathcal{G}_{obs}$, the feature matrix $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]^T$ and the training labels $\mathbf{Y}_{\mathcal{L}} = \{\boldsymbol{y}_i : i \in \mathcal{L}\}$.

A GCNN performs graph convolution operations within a neural network architecture to address this task. Although there are many different versions of the graph convolution operation, the layerwise propagation rule for the simpler architectures (Defferrard et al., 2016; Kipf & Welling, 2017) can be expressed as:

$$\mathbf{H}^{(1)} = \sigma(\hat{\mathbf{A}}_{\mathcal{G}}\mathbf{X}\mathbf{W}^{(0)})\,, \tag{1}$$

$$\mathbf{H}^{(l+1)} = \sigma(\hat{\mathbf{A}}_{\mathcal{G}}\mathbf{H}^{(l)}\mathbf{W}^{(l)})\,. \tag{2}$$

Here $\mathbf{H}^{(l)}$ are the output features from layer $l-1$, and σ is a pointwise non-linear activation function. The normalized adjacency operator $\hat{\mathbf{A}}_{\mathcal{G}}$, which is derived from the observed graph, determines the mixing of the output features across the graph at each layer. $\mathbf{W}^{(l)}$ denotes the weights of the neural network at layer l. We use $\mathbf{W} = \{\mathbf{W}^l\}_{l=1}^{L}$ to denote all GCNN weights.

For an L-layer network, the final output is $\mathbf{Z} = \mathbf{H}^{(L)}$. Learning of the weights of the neural network is carried out by backpropagation with the objective of minimizing an error metric between the training labels $\mathbf{Y}_{\mathcal{L}}$ and the network predictions $\mathbf{Z}_{\mathcal{L}} = \{z_i : i \in \mathcal{L}\}$, at the nodes in the training set.

3 Methodology

We consider a Bayesian approach, by constructing a joint posterior distribution of the graph, the weights in the GCNN and the node labels. Our goal is to compute the marginal posterior probability of the node labels, which is expressed as follows:

$$p(\mathbf{Z}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \mathcal{G}_{obs}) = \int p(\mathbf{Z}|\mathbf{W}, \mathcal{G}_{obs}, \mathbf{X})p(\mathbf{W}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \mathcal{G})p(\mathcal{G}|\mathcal{G}_{obs}, \mathbf{X}, \mathbf{Y}_{\mathcal{L}})\,d\mathbf{W}\,d\mathcal{G}\,. \tag{3}$$

Here $\mathbf{W}$ denotes the random weights of a Bayesian GCNN over graph $\mathcal{G}$. In a node classification setting, the term $p(\mathbf{Z}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \mathcal{G}_{obs})$ is modelled using a categorical distribution by applying a softmax function to the output of the GCNN. As the integral in equation 3 can not be computed analytically, a Monte Carlo approximation is formed as follows:

$$p(\mathbf{Z}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \mathcal{G}_{obs}) \approx \frac{1}{S}\sum_{s=1}^{S}\frac{1}{N_G}\sum_{i=1}^{N_G} p(\mathbf{Z}|\mathbf{W}_{s,i}, \mathcal{G}_{obs}, \mathbf{X})\,. \tag{4}$$

In this approximation, N_G graphs $\mathcal{G}_i$ are sampled from $p(\mathcal{G}|\mathcal{G}_{obs}, \mathbf{X}, \mathbf{Y}_{\mathcal{L}})$ and S weight samples $\mathbf{W}_{s,i}$ are drawn from $p(\mathbf{W}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \mathcal{G}_i)$ by training a Bayesian GCN corresponding to the graph $\mathcal{G}_i$.

(Zhang et al., 2019) assume that $\mathcal{G}_{obs}$ is a sample from a collection of graphs associated with a parametric random graph model and their approach targets inference of $p(\mathcal{G}|\mathcal{G}_{obs})$ via marginalization of the random graph parameters, ignoring any possible dependence of the graph $\mathcal{G}$ on the features $\mathbf{X}$ and the labels $\mathbf{Y}_{\mathcal{L}}$. By contrast, we consider a non-parametric posterior distribution of the graph $\mathcal{G}$ as $p(\mathcal{G}|\mathcal{G}_{obs}, \mathbf{X}, \mathbf{Y}_{\mathcal{L}})$. This allows us to incorporate the information provided by the features $\mathbf{X}$ and the labels $\mathbf{Y}_{\mathcal{L}}$ in the graph inference process.

We denote the symmetric adjacency matrix with non-negative entries of the random undirected graph $\mathcal{G}$ by $A_{\mathcal{G}}$. The prior distribution for $\mathcal{G}$ is defined as

$$p(\mathcal{G}) \propto \begin{cases} \exp\left(\alpha \mathbf{1}^T \log(A_{\mathcal{G}}\mathbf{1}) - \beta \|A_{\mathcal{G}}\|_F^2\right), & \text{if } A_{\mathcal{G}} \geq \mathbf{0}, A_{\mathcal{G}} = A_{\mathcal{G}}^T \\ 0, & \text{otherwise}\,. \end{cases} \tag{5}$$

For allowable graphs, the first term in the log prior prevents any isolated node in $\mathcal{G}$ and the second encourages low weights for the links. α and β are hyperparameters which control the scale and sparsity of $A_{\mathcal{G}}$. The joint likelihood of $\mathbf{X}$, $\mathbf{Y}_{\mathcal{L}}$ and $\mathcal{G}_{obs}$ conditioned on $\mathcal{G}$ is:

$$p(\mathbf{X}, \mathbf{Y}_{\mathcal{L}}, \mathcal{G}_{obs}|\mathcal{G}) \propto \exp\left(-\|A_{\mathcal{G}} \circ Z(\mathbf{X}, \mathbf{Y}_{\mathcal{L}}, \mathcal{G}_{obs})\|_{1,1}\right), \tag{6}$$

where $Z(\mathbf{X}, \mathbf{Y}_{\mathcal{L}}, \mathcal{G}_{obs}) \geq \mathbf{0}$ is a symmetric pairwise distance matrix between the nodes. The symbol $\circ$ denotes the Hadamard product and $\|\cdot\|_{1,1}$ denotes the elementwise ℓ_1 norm. We propose to use

$$Z(\mathbf{X}, \mathbf{Y}_{\mathcal{L}}, \mathcal{G}_{obs}) = Z_1(\mathbf{X}, \mathcal{G}_{obs}) + \delta Z_2(\mathbf{X}, \mathbf{Y}_{\mathcal{L}}, \mathcal{G}_{obs}), \tag{7}$$

where, the (i, j)'th entries of Z_1 and Z_2 are defined as follows:

$$Z_{1,ij}(\mathbf{X}, \mathcal{G}_{obs}) = \|\boldsymbol{e}_i - \boldsymbol{e}_j\|^2, \tag{8}$$

$$Z_{2,ij}(\mathbf{X}, \mathbf{Y}_{\mathcal{L}}, \mathcal{G}_{obs}) = \frac{1}{|\mathcal{N}_i||\mathcal{N}_j|} \sum_{k \in \mathcal{N}_i} \sum_{l \in \mathcal{N}_j} \mathbf{1}_{(\hat{c}_k \neq \hat{c}_l)}\,. \tag{9}$$

Here, $\boldsymbol{e}_i$ is any suitable embedding of node i and $\hat{c}_i$ is the label obtained at node i by a base classification algorithm. Z_1 summarizes the pairwise distance in terms of the observed topology and features and Z_2 encodes the dissimilarity in node labels robustly by considering the obtained labels of the neighbours in the observed graph. In this paper, we choose the Graph Variational Auto-Encoder (GVAE) algorithm (Kipf & Welling, 2016) as the node embedding method to obtain the $\boldsymbol{e}_i$ vectors and use the GCNN proposed by (Kipf & Welling, 2017) as the base classifier to obtain the $\hat{c}_i$ values. The neighbourhood is defined as:

$$\mathcal{N}_i = \{j|(i,j) \in \mathcal{E}_{\mathcal{G}_{obs}}\} \cup \{i\}\,.$$

δ is a hyperparameter which controls the importance of Z_2 relative to Z_1. We set:

$$\delta = \frac{\max\limits_{i,j} Z_{1,ij}}{\max\limits_{i,j} Z_{2,ij}}\,.$$

In order to use the approximation in equation 4, we need to obtain samples from the posterior of $\mathcal{G}$. However, the design of a suitable MCMC in this high dimensional ($\mathcal{O}(N^2)$, where N is the number of the nodes) space is extremely challenging and computationally expensive. Instead we replace the integral over $\mathcal{G}$ in equation 3 by the maximum a posteriori estimate of $\mathcal{G}$, following the approach of (MacKay, 1996). We solve the following optimization problem

$$\hat{\mathcal{G}} = \arg\max_{\mathcal{G}} p(\mathcal{G}|\mathcal{G}_{obs}, \mathbf{X}, \mathbf{Y}_{\mathcal{L}}), \tag{10}$$

and approximate the integral in equation 3 as follows:

$$p(\mathbf{Z}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \mathcal{G}_{obs}) \approx \frac{1}{S} \sum_{s=1}^{S} p(\mathbf{Z}|\mathbf{W}_s, \mathcal{G}_{obs}, \mathbf{X})\,. \tag{11}$$

Algorithm 1 Bayesian GCN using non-parametric graph learning

```
1: Input: $\mathcal{G}_{obs}$, $\mathbf{X}$, $\mathbf{Y}_{\mathcal{L}}$
2: Output: $p(\mathbf{Z}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \mathcal{G}_{obs})$
3: Train a node embedding algorithm using $\mathcal{G}_{obs}$ and $\mathbf{X}$ to obtain $\boldsymbol{e}_i$ for $1 \leq i \leq N$. Compute $Z_1$ using equation 8.
4: Train a base classifier using $\mathcal{G}_{obs}$, $\mathbf{X}$ and $\mathbf{Y}_{\mathcal{L}}$ to obtain $\hat{c}_i$ for $1 \leq i \leq N$. Compute $Z_2$ using equation 9.
5: Compute $Z$ using equation 7.
6: Solve the optimization problem in 12 to obtain $A_{\hat{\mathcal{G}}}$ (equivalently, $\hat{\mathcal{G}}$).
7: for $s = 1$ to $S$ do
8:     Sample weights $W_s$ using MC dropout by training a GCNN over the graph $\hat{\mathcal{G}}$.
9: end for
10: Approximate $p(\mathbf{Z}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \mathcal{G}_{obs})$ using equation 11.
```

Here, S weight samples $\mathbf{W}_s$ are drawn from $p(\mathbf{W}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \hat{\mathcal{G}})$. The MAP inference in 10 is equivalent to learning a $N \times N$ symmetric adjacency matrix of $\hat{\mathcal{G}}$.

$$A_{\hat{\mathcal{G}}} = \underset{\substack{A_{\mathcal{G}} \in \mathbf{R}_+{}^{N\times N}, \\ A_{\mathcal{G}} = A_{\mathcal{G}}^T}}{\arg\min} \; \|A_{\mathcal{G}} \circ Z\|_{1,1} - \alpha \mathbf{1}^T \log(A_{\mathcal{G}}\mathbf{1}) + \beta \|A_{\mathcal{G}}\|_F^2 \,. \tag{12}$$

The optimization problem in 12 has been studied in the context of graph learning from smooth signals. In (Kalofolias, 2016), a primal-dual optimization algorithm is employed to solve this problem. However the complexity of this approach scales as $\mathcal{O}(N^2)$, which can be prohibitive for large graphs. We use the approximate algorithm in (Kalofolias & Perraudin, 2017), which has an approximate $\mathcal{O}(N \log N)$ complexity. This formulation allows us to effectively use one hyperparameter instead of α and β to control the sparsity of the solution and provides a useful heuristic for choosing a suitable value.

Various techniques such as expectation propagation (Hernández-Lobato & Adams, 2015), variational inference (Gal & Ghahramani, 2016; Sun et al., 2017; Louizos & Welling, 2017), and Markov Chain Monte Carlo methods (Neal, 1993; Korattikara et al., 2015; Li et al., 2016) can be employed for the posterior inference of the GCNN weights. Following the approach in (Zhang et al., 2019), we train a GCNN over the inferred graph $\hat{\mathcal{G}}$ and use Monte Carlo dropout (Gal & Ghahramani, 2016) to sample $\mathbf{W}_s$ from a particular variational approximation of $p(\mathbf{W}|\mathbf{Y}_{\mathcal{L}}, \mathbf{X}, \hat{\mathcal{G}})$. The resulting algorithm is described in Algorithm 1.

4 Experimental Results

We investigate the performance of the proposed Bayesian GCNN on three citation datasets (Sen et al., 2008): Cora, CiteSeer, and Pubmed. In these datasets each node corresponds to a document and the undirected edges are citation links. Each node has a sparse bag-of-words feature vector associated with it and the node label represents the topic of the document. We address a semi-supervised node classification task where we have access to the labels of a few nodes per class and the goal is to infer labels for the others. We consider three different experimental settings where we have 5, 10 and 20 labels per class in the training set. The partitioning of the data in 20 labels per class case is the same as in (Yang et al., 2016) whereas in the other two cases, we construct the training set by including the first 5 or 10 labels from the previous partition. The hyperparameters of GCNN are borrowed from (Kipf & Welling, 2017) and are used for the BGCN algorithms as well.

We compare the proposed BGCN in this paper with ChebyNet (Defferrard et al., 2016), GCNN (Kipf & Welling, 2017), GAT (Veličković et al., 2018) and the BGCN in (Zhang et al., 2019). Table 1 shows the summary of results based on 50 runs with random weight initializations.

The results in Table 1 show that the proposed algorithm yields higher classification accuracy compared to the other algorithms in most cases. Figure 1 demonstrates that in most cases, for the Cora and the Citeseer datasets, the proposed BGCN algorithm corrects more errors of the GCNN base classifier for low degree nodes. The same trend is observed for the Pubmed dataset as well. In

Dataset	**Cora**			**Citeseer**			**Pubmed**		
Labels/class	**5**	**10**	**20**	**5**	**10**	**20**	**5**	**10**	**20**
ChebyNet	67.9±3.1	72.7±2.4	80.4±0.7	53.0±1.9	67.7±1.2	70.2±0.9	68.1±2.5	69.4±1.6	76.0±1.2
GCNN	74.4±0.8	74.9±0.7	**81.6±0.5**	55.4±1.1	65.8±1.1	70.8±0.7	69.7±0.5	72.8±0.5	78.9±0.3
GAT	73.5±2.2	74.5±1.3	81.6±0.9	55.4±2.6	66.1±1.7	70.8±1.0	70.0±0.6	71.6±0.9	76.9±0.5
BGCN	75.3±0.8	76.6±0.8	81.2±0.8	57.3±0.8	70.8±0.6	72.2±0.6	70.9±0.8	72.3±0.8	76.6±0.7
BGCN (ours)	**76.0±1.1**	**76.8±0.9**	80.3±0.6	**59.0±1.5**	**71.7±0.8**	**72.6±0.6**	**73.3±0.7**	**73.9±0.9**	**79.2±0.5**

Table 1: Classification accuracy (percentage of correctly predicted labels) for the three datasets.

Figure 2, the adjacency matrix ($A_{\hat{\mathcal{G}}}$) of the MAP estimate graph $\hat{\mathcal{G}}$ is shown along with the observed adjacency matrix $A_{\mathcal{G}_{obs}}$ for the Cora dataset. We observe that compared to $A_{\mathcal{G}_{obs}}$, $A_{\hat{\mathcal{G}}}$ has denser connectivity among the nodes with the same label.

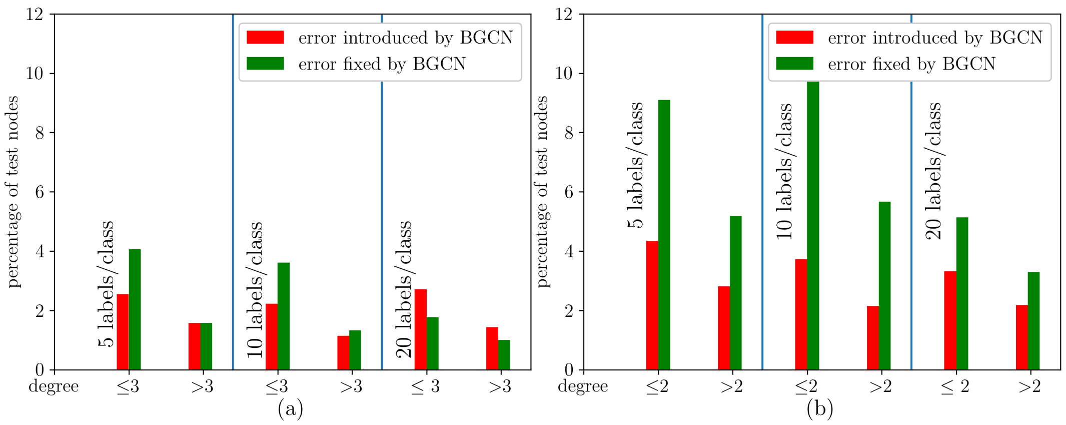

Figure 1: Barplot of different categories of nodes in the (a) Cora and (b) Citeseer datasets based on the classification results of the GCNN and the proposed BGCN algorithms. The two groups are formed by thresholding the degree of the nodes in the test set at the median value.

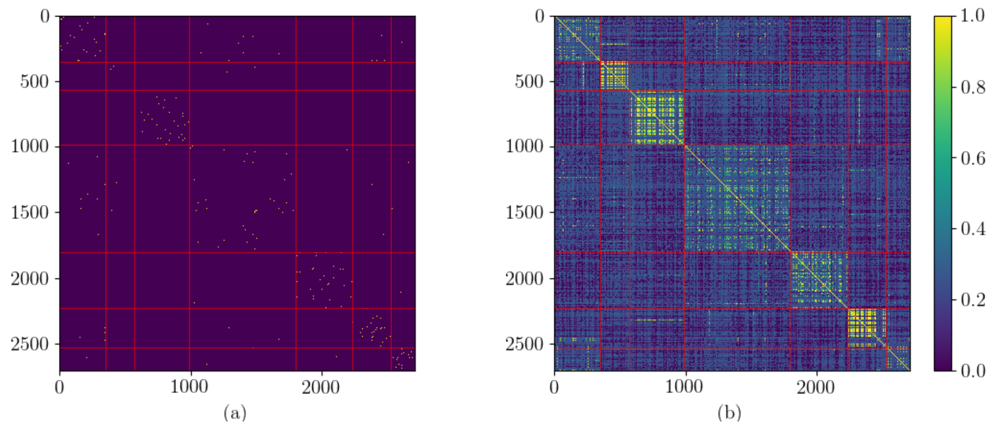

Figure 2: (a) the observed adjacency matrix ($A_{\mathcal{G}_{obs}}$) and (b) the MAP estimate of adjacency matrix ($A_{\hat{\mathcal{G}}}$) for the Cora dataset.

5 Conclusion

In this paper, we present a Bayesian GCNN using a non-parametric graph inference technique. The proposed algorithm achieves superior performance when the amount of available labels during the training process is limited. Future work will investigate extending the methodology to other graph based learning tasks, incorporating other generative models for graphs and developing scalable techniques to perform effective inference for those models.

REFERENCES

Rushil Anirudh and Jayaraman J. Thiagarajan. Bootstrapping graph convolutional neural networks for autism spectrum disorder classification. *arXiv:1704.07487*, 2017.

James Atwood and Don Towsley. Diffusion-convolutional neural networks. In *Proc. Adv. Neural Inf. Proc. Systems*, 2016.

Xavier Bresson and Thomas Laurent. Residual gated graph convnets. *arXiv:1711.07553*, 2017.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In *Proc. Int. Conf. Learning Representations*, Scottsdale, AZ, USA, 2013.

Jianfei Chen, Jun Zhu, and Le Song. Stochastic Training of Graph Convolutional Networks with Variance Reduction. In *Proc. Int. Conf. Machine Learning*, 2018a.

Jie Chen, Tengfei Ma, and Cao Xiao. FastGCN: fast learning with graph convolutional networks via importance sampling. In *Proc. Int. Conf. Learning Representations*, 2018b.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proc. Adv. Neural Inf. Proc. Systems*, 2016.

David Duvenaud, Dougal Maclaurin, et al. Convolutional networks on graphs for learning molecular fingerprints. In *Proc. Adv. Neural Inf. Proc. Systems*, 2015.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proc. Int. Conf. Machine Learning*, 2016.

William Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proc. Adv. Neural Inf. Proc. Systems*, 2017.

Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv:1506.05163*, 2015.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *Proc. Int. Conf. Machine Learning*, 2015.

Vassilis Kalofolias. How to learn a graph from smooth signals. In *Proc. Artificial Intelligence and Statistics*, 2016.

Vassilis Kalofolias and Nathanaël Perraudin. Large Scale Graph Learning from Smooth Signals. *arXiv e-print arXiv : 1710.05654*, 2017.

Thomas Kipf and Max Welling. Variational graph auto-encoders. *arXiv:1611.07308*, 2016.

Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proc. Int. Conf. Learning Representations*, 2017.

Anoop Korattikara, Vivek Rathod, Kevin Murphy, and Max Welling. Bayesian dark knowledge. In *Proc. Adv. Neural Inf. Proc. Systems*, 2015.

Ron Levie, Federico Monti, Xavier Bresson, and Michael M Bronstein. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Trans. Signal Processing*, 67(1), Jan. 2019.

Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. Pre-conditioned stochastic gradient Langevin dynamics for deep neural networks. In *Proc. AAAI Conf. Artificial Intelligence*, 2016.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In *Proc. Int. Conf. Learning Representations*, 2016.

Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. *arXiv:1703.01961*, 2017.

David J. C. MacKay. *Maximum entropy and Bayesian methods*, chapter Hyperparameters: Optimize, or Integrate Out?, pp. 43–59. Springer Netherlands, 1996.

Federico Monti, Davide Boscaini, et al. Geometric deep learning on graphs and manifolds using mixture model CNNs. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, Jul. 2017.

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

Radford M. Neal. Bayesian learning via stochastic dynamics. In *Proc. Adv. Neural Inf. Proc. Systems*, pp. 475–482, 1993.

Felipe P. Such, Shagan Sah, et al. Robust spatial filtering with graph convolutional neural networks. *IEEE J. Sel. Topics Signal Proc.*, 11(6):884–896, Sept 2017.

Prithviraj Sen, Galileo Namata, et al. Collective classification in network data. *AI Magazine*, 29(3): 93, 2008.

Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. *arXiv:1704.02901*, 2017.

Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning multiagent communication with backpropagation. In *Proc. Adv. Neural Inf. Proc. Systems*, 2016.

Ke Sun, Piotr Koniusz, and Jeff Wang. Fisher-Bures adversary graph convolutional networks. *arXiv e-prints, arXiv : 1903.04154*, 2019.

Shengyang Sun, Changyou Chen, and Lawrence Carin. Learning structured weight uncertainty in Bayesian neural networks. In *Proc. Artificial Intelligence and Statistics*, 2017.

Petar Veličković, Guillem Cucurull, et al. Graph attention networks. In *Proc. Int. Conf. Learning Representations*, Vancouver, Canada, Apr. 2018.

Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *arXiv preprint arXiv:1603.08861*, 2016.

Yingxue Zhang, Soumyasundar Pal, Mark Coates, and Deniz Üstebay. Bayesian graph convolutional neural networks for semi-supervised classification. In *Proc. AAAI Conf. Artificial Intelligence*, 2019.