PAN: Path Integral Based Convolution for Deep Graph Neural Networks

Zheng Ma [1] **Ming Li** [2 3] **Yu Guang Wang** [4]

arXiv:1904.10996v1 [cs.LG] 24 Apr 2019

Abstract

Convolution operations designed for graph-structured data usually utilize the graph Laplacian, which can be seen as message passing between the adjacent neighbors through a generic random walk. In this paper, we propose PAN, a new graph convolution framework that involves every path linking the message sender and receiver with learnable weights depending on the path length, which corresponds to the maximal entropy random walk. PAN generalizes the graph Laplacian to a new transition matrix we call *maximal entropy transition* (MET) matrix derived from a path integral formalism. Most previous graph convolutional network architectures can be adapted to our framework, and many variations and derivatives based on the path integral idea can be developed. Experimental results show that the path integral based graph neural networks have great learnability and fast convergence rate, and achieve state-of-the-art performance on benchmark tasks.

1. Introduction

The triumph of convolutional neural networks (CNNs) has motivated researchers to develop similar architectures for graph-structured data. The problem is challenging due to the absence of regular grids. One notable proposal is to define convolutions in the Fourier space (Bruna et al., 2014; Bronstein et al., 2017). This method relies on finding the spectrum of the graph Laplacian $I - D^{-1}A$ or $I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ (depending on how normalization is done), where A is the adjacency matrix of the graph and D is the corresponding degree matrix, and then applies filters to the components of input signal X under the basis of the graph Laplacian. Due to the high computational complexity of diagonalizing the graph Laplacian, many simplifications have been proposed (Defferrard et al., 2016; Kipf & Welling, 2017).

The graph Laplacian based methods essentially rely on message passing (Gilmer et al., 2017) between directly connected nodes with equal weights shared among all edges, which is at the heart a generic random walk (GRW) defined on graphs. This can be seen most obviously from the GCN model (Kipf & Welling, 2017), where the normalized adjacency matrix is directly applied to the left hand side of the input. Mathematically, $D^{-1}A$ is known as the transition matrix of a particle doing a random walk on the graph, where the particle hops to all directly connected nodes with equiprobability. Many direct space based methods (Li et al., 2015; Grover & Leskovec, 2016; Yang et al., 2016; Veličković et al., 2018a; Thekumparampil et al., 2018) can be viewed as generalizations of GRW that enable one to do a biased average of the neighbors, although they are in general more complicated and the bias usually depends on trainable parameters.

In this paper, we present PAN, a general framework for graph convolution inspired by the path integral idea in physics. We go beyond the generic diffusion picture and consider the message passing along all possible paths between the sender and receiver on a graph, with trainable weights depending on the path length. This results in a *maximal entropy transition* (MET) matrix, which plays the same role as graph Laplacian. By introducing a fictitious temperature, we can continuously tune our model from a fully localized one (i.e., MLP) to a global structure based model. Great learnability and fast convergence rate of PAN are observed when training the benchmark dataset. Numerous variations of MET can be developed, and many current models can be seen as special cases of the presented framework.

2. Model

In the most general form, we heuristically propose a statistical mechanics model on how information is averaged between different nodes on a given graph. Using the formalism of Feynman's path integral (Feynman & Mechanics, 1979), but modified for discrete graph structures, we write

[1] Department of Physics, Princeton University, New Jersey, USA [2] School of Information Technology in Education, South China Normal University, Guangzhou, China [3] Department of Computer Science and Information Technology, La Trobe University, Melbourne, Australia [4] School of Mathematics and Statistics, The University of New South Wales, Sydney, Australia. Correspondence to: Zheng Ma <zhengm@princeton.edu>.

observable ϕ_i at node i for a graph with N nodes as

$$\phi_i = \frac{1}{Z_i}\sum_{j=1}^{N}\phi_j \sum_{\{\mathbf{l}|l_0=i,l_{|\mathbf{l}|}=j\}} e^{-\frac{E[\mathbf{l}]}{T}}, \tag{1}$$

where Z_i is the normalization factor known as the *partition function* for the i-th node, and $\mathbf{l}$ is any path formed on the graph. Here a path $\mathbf{l}$ is a sequence of connected nodes $(l_0 l_1 \dots l_{|\mathbf{l}|})$ where $A_{l_i l_{i+1}} = 1$, and the length of the path is denoted by $|\mathbf{l}|$. Since a statistical mechanics perspective is more straightforward in our case, we directly change the exponential term, which is originally an integral of Lagrangian, to a Boltzmann's factor with fictitious energy $E[\mathbf{l}]$ and temperature T. (We choose Boltzmann's constant $k_B = 1$.) Nevertheless, we still exploit the fact that the energy is a functional of the path, which gives us a way to weight the influence of other nodes through a certain path. The fictitious temperature controls the excitation level of the system, which reflects that to what extent information is localized or extended. Specifically, low temperature corresponds to a low-pass filter, while high temperature corresponds to a high-pass one. In practice, there is no need to learn the fictitious temperature or energy separately, instead the neural networks can directly learn the overall weights, as would be made clearer later.

To obtain an explicit form of our model, we now introduce some mild assumptions and simplifications. Intuitively, we know that information quality usually decays as the path between the message sender and the receiver becomes longer, thus it is reasonable to assume that the energy is not only a functional of path, but can be further simplified as a function that solely depends on the length of the path. In the random walk picture, this means that the hopping is equiprobable among all the paths that have the same length, which maximizes the Shannon entropy of the probability distribution of paths globally, and thus the random walk is given the name maximal entropy random walk (Burda et al., 2009). For a weighted graph, a feasible choice for the functional form of the energy could be $E(l_{\text{eff}})$, where the effective length of the path l_{eff} can be defined as a summation of the inverse of weights along the path, i.e., $l_{\text{eff}} = \sum_{i=0}^{|l|-1} 1/w_{l_i l_{i+1}}$. After simplification, we can regroup the summation by first conditioning on the length of the path. Define the overall n-th layer weight $k(n;i)$ for node i by

$$k(n;i) = \frac{1}{Z_i}\sum_{j=1}^{N} g(i,j;n) e^{-\frac{E(n)}{T}}, \tag{2}$$

where $g(i,j;n)$ denotes the number of paths between nodes i and j with length of n, or *density of states* for the energy level $E(n)$ with respect to nodes i and j, and the summation is taken over all nodes of the graph. Presumably, the energy $E(n)$ is an increasing function of n, which leads to a decaying weight as n increases.[1] By applying a cutoff of the maximal path length L, we can rewrite (1) as

$$\begin{aligned}\phi_i &= \sum_{n=0}^{L} k(n;i) \sum_{j=1}^{N} \frac{g(i,j;n)}{\sum_{s=1}^{N} g(i,s;n)} \phi_j \\ &= \frac{1}{Z_i}\sum_{n=0}^{L} e^{-\frac{E(n)}{T}} \sum_{j=1}^{N} g(i,j;n)\phi_j,\end{aligned} \tag{3}$$

and the partition function can be explicitly written as

$$Z_i = \sum_{n=0}^{L} e^{-\frac{E(n)}{T}} \sum_{j=1}^{N} g(i,j;n). \tag{4}$$

A nice feature of this formalism is that we can easily compute $g(i,j;n)$ by raising the power of the adjacency matrix A to n, which is a well-known property of the adjacency matrix from graph theory, i.e.,

$$g(i,j;n) = A_{ij}^n. \tag{5}$$

Clearly, from (3) and (5) we have a group of self-consistent equations governed by a transition matrix M (a counterpart of the *propagator* in quantum mechanics), which is defined as

$$M_{ij} = \sum_{n=0}^{L} k(n;i) \frac{A_{ij}^n}{\sum_{s=1}^{N} A_{is}^n}. \tag{6}$$

We call the matrix M *maximal entropy transition* (MET) matrix, with regard to the fact that it realizes maximal entropy under the microcanonical ensemble. This transition matrix replaces the role of the graph Laplacian under our formalism. It can be written in a more compact form

$$M = Z^{-1} \sum_{n=0}^{L} e^{-\frac{E(n)}{T}} A^n, \tag{7}$$

where $Z = \operatorname{diag}(Z_i)$. More generally, for paths with constraints such as shortest paths or self-avoiding paths, A^n can be replaced by another matrix $G(n)$, where $G_{ij}(n) = g(i,j;n)$.

The *eigenstates*, or the basis of the system $\{\psi_i\}$ satisfy

$$M\psi_i = \lambda_i \psi_i. \tag{8}$$

Similar to the basis formed by the graph Laplacian, one can define graph convolution based on the new basis we obtained in (8), which has a distinct new physical meaning. The convolution associated with MET is computationally

[1] This does not mean that $k(n;i)$ should necessarily be a decreasing function, since $g(i,j;n)$ grows exponentially in general. It would be valid to apply a cutoff as long as $E(n) \gg nT \ln \lambda_1$ for large n, where λ_1 is the largest eigenvalue of the adjacency matrix A.

nontrivial since the matrix M now relies on a group of weights $\exp(-E(n)/T)$ which need to be learned and updated. To reduce the high computational complexity of diagonalizing a large matrix, we apply an architecture similar to GCN (Kipf & Welling, 2017) by circumventing the diagonalization of the matrix M, and directly multiply it to the left hand side of the input and accompany it by multiplying another weight matrix W at the right hand side. The convolutional layer would then be reduced to a simple form

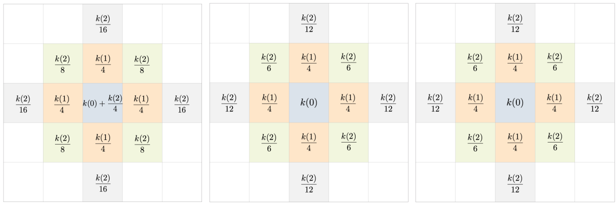
$$X^{(h+1)} = M^{(h)} X^{(h)} W^{(h)}, \tag{9}$$

where h refers to the layer number. Applying M to the input X is essentially a weighted average among neighbors of a given node. Here we call the graph convolution induced by MET the *MET convolution*. The model (9) can be simplified further. Instead of learning the Boltzmann's factors which enter $k(n;i)$ through (6), one may treat $k(n;i)$ as a constant $k(n)$ that is independent of nodes and learn it directly. This simplification circumvents normalizing the summation by Z and eases the training process. The convolutional layer is then simplified as

$$X^{(h+1)} = \sum_{n=0}^{L} k^{(h)}(n) D_n^{-1} A^n X^{(h)} W^{(h)}, \tag{10}$$

where D_n is the degree matrix for A^n. We call the graph convolutional networks in (9) and (10) PAN as the path integral based MET convolution is used.

Note that this simplified model can only be equal to (9) when the graph is regular. Interestingly, if one interprets an image as a regular graph, where pixels with shared edges are considered connected, an analog can be drawn between the model (10) and traditional CNNs. For traditional CNNs, elements of a convolution filter can be associated with the Cartesian coordinates of the neighbors of a given node. For irregular graphs, this coordinate system is not available. However, one can still associate the filter elements (i.e., $k(n)$) with the neighbors by the scalar quantity "distance". As an example, we explicitly map model (10) to a traditional convolution filter in $\mathbb{R}^2$ in Figure 1. We present the mapping for three different types of paths, although in this paper we only focus on the maximal entropy one due to its simple form of $g(i,j;n)$.



		$\frac{k(2)}{16}$		
	$\frac{k(2)}{8}$	$\frac{k(1)}{4}$	$\frac{k(2)}{8}$	
$\frac{k(2)}{16}$	$\frac{k(1)}{4}$	$k(0)+\frac{k(2)}{4}$	$\frac{k(1)}{4}$	$\frac{k(2)}{16}$
	$\frac{k(2)}{8}$	$\frac{k(1)}{4}$	$\frac{k(2)}{8}$	
		$\frac{k(2)}{16}$		

(a) Maximal entropy

		$\frac{k(2)}{12}$		
	$\frac{k(2)}{6}$	$\frac{k(1)}{4}$	$\frac{k(2)}{6}$	
$\frac{k(2)}{12}$	$\frac{k(1)}{4}$	$k(0)$	$\frac{k(1)}{4}$	$\frac{k(2)}{12}$
	$\frac{k(2)}{6}$	$\frac{k(1)}{4}$	$\frac{k(2)}{6}$	
		$\frac{k(2)}{12}$		

(b) Shortest path

		$\frac{k(2)}{12}$		
	$\frac{k(2)}{6}$	$\frac{k(1)}{4}$	$\frac{k(2)}{6}$	
$\frac{k(2)}{12}$	$\frac{k(1)}{4}$	$k(0)$	$\frac{k(1)}{4}$	$\frac{k(2)}{12}$
	$\frac{k(2)}{6}$	$\frac{k(1)}{4}$	$\frac{k(2)}{6}$	
		$\frac{k(2)}{12}$		

(c) Self-avoiding

Figure 1. Mapping model (10) ($L = 2$) to a traditional convolution filter for different types of paths. (a) Maximal entropy paths. (b) Shortest paths. (c) Self-avoiding paths. (b) and (c) happen to be the same for $L = 2$.

3. Related work

The maximal entropy random walk has already shown excellent performance on link prediction (Li et al., 2011) or community detection (Ochab & Burda, 2013) tasks. Essentially, most graph Laplacian based models (Abu-El-Haija et al., 2018; Atwood & Towsley, 2016; Bruna et al., 2014; Defferrard et al., 2016; Kipf & Welling, 2017; Monti et al., 2017; Such et al., 2017; Xu et al., 2019) can be adapted to our framework by replacing the graph Laplacian with the MET matrix M. This represents a change from the GRW model (Perozzi et al., 2014) or its modified versions (Tang et al., 2015; Grover & Leskovec, 2016) which sample random walks by local information or the similarity of nodes, to a global information based parameter-free random walk. Many popular models can be related to or viewed as certain explicit realizations of our framework. Besides the direct link between our model and traditional CNNs mentioned above, the MET matrix can also be interpreted as an operator that acts on the graph input, which works as a kernel that allocates appropriate weights among the neighbors of a given node. This mechanism is similar to the attention mechanism (Veličković et al., 2018a), while we restrict the functional form of M based on physical intuitions. Specifically, we suppose the operator can be expanded as a series $\sum_{n=0}^{r} f_n(A, X) A^n$, it then performs the attention mechanism but preserves a compact form. Although we keep the number of features by applying M, one can easily concatenate the aggregated information of neighbors like GraphSAGE (Hamilton et al., 2017) or GAT (Veličković et al., 2018a). The optimal order r of the series depends on the intrinsic properties of the graph, which is represented by temperature T. Incorporating more terms is analogous to having more particles excited to higher energy level at higher temperature. For instance, in the *low-temperature limit*, $M = I$, the model is reduced to the MLP model. In the *high-temperature limit*, all factors $\exp(-E(n)/T)$ become effectively one, and the summation is dominated by the term with the largest power. This can be seen by noticing

$$A^n = \sum_{i=1}^{N} \lambda_i^n \psi_i \psi_i^T, \tag{11}$$

where $\lambda_1, \ldots, \lambda_N$ is sorted in a descending order. By the Perron-Frobenius theorem we may only keep the leading order term with the unique largest eigenvalue λ_1 when $n \to \infty$. We then reach a prototype of the high temperature model $X^{(h+1)} = (I + \psi_1 \psi_1^T) X^{(h)} W^{(h)}$. Empirically, a moderate order of one to three seems to perform better than both extremes, which well reflects the intrinsic dynamics of the graph. In particular, by choosing $L = 1$ and $E(0) = E(1) = 0$, model (10) is essentially the GCN model (Kipf & Welling, 2017). The trick of adding self-loops is automatically realized in higher powers of A. By replacing A in (10) or (9) with $D^{-1}A$ or $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, we can easily transform our model to a multi-step GRW version, which is indeed the format of LanczosNet (Liao et al., 2019). Moreover, Lanczos algorithm may be directly applied to the MET matrix once it is symmetrically normalized.

4. Experiments

4.1. Datasets and Baselines

We test our method on three public available citation graph datasets: Citeseer, Cora and Pubmed, with the fixed data splits performed in (Yang et al., 2016; Kipf & Welling, 2017), in comparison with some existing methods including node2vec (Grover & Leskovec, 2016), Planetoid (Yang et al., 2016), skip-gram based graph embeddings (DeepWalk) (Perozzi et al., 2014), ChebNet (Defferrard et al., 2016), GCN (Kipf & Welling, 2017) (together with some baselines as compared in their work), attention-based models such as AGNN (Thekumparampil et al., 2018) and GAT (Veličković et al., 2018a), deep graph infomax (DGI) (Veličković et al., 2018b), Bootstrap (Eliav & Cohen, 2018), multi-scale deep graph convolutional networks AdaLNet (Liao et al., 2019), simplified GCN model (SGC) (Wu et al., 2019), and graph wavelet neural network (GWNN) (Xu et al., 2019).

4.2. Results and Discussion

In Table 1, we present partial results for performance comparison. See Supplementary Material for a full comparison list and the detailed experimental setup. The records are averaged classification accuracies (in percentage %) of 10 independent trials. It can be observed that our methods for semi-supervised node classification outperforms most of the existing models, especially for Pubmed. For Cora, the obtained accuracy is slightly less than DGI, AGNN, GAT and GWNN, but higher than all the other models. For Citeseer, our result outperforms most of the models, apart from DGI, SGC, GWNN, GAT. For Pubmed, our model has a better performance than all other models but AGNN. Overall, the performance of our model can be placed in the top five among all models.

We plot the trend of validation loss and accuracy (with mean and standard deviations) for both PAN and GCN on Cora in Figure 2. It illustrates that PAN converges faster and reaches higher accuracy compared to GCN, and finally obtains a better test performance as shown in Table 1. For model (10), a grid search of $k(n)$ may be more reliable than the gradient descent if L is not large. The generalization ability may be improved by further constraining the number of parameters. For example, one may assume a certain form of $E(n)$, such as $E(n) \propto n^{\alpha}$ with $\alpha \geq 1$, and only train the temperature.

Table 1. Performance comparison of PAN and some previously published models for Cora, Citeseer and Pubmed.

Method	Cora	Citeseer	Pubmed
node2vec	74.9	54.7	75.3
Planetoid	75.7	64.7	77.2
DeepWalk	67.2	43.2	65.3
ChebNet	81.2	69.8	74.4
GCN	81.5	70.3	79.0
AGNN	83.1	71.1	79.9
GAT	83.0	72.5	79.0
Bootstrap	78.4	53.6	78.8
DGI	82.3	71.8	76.8
AdaLNet	80.4	68.7	78.1
SGC	81.0	71.9	78.9
GWNN	82.8	71.7	79.1
PAN ($L = 2$)	**82.0**	**71.2**	**79.2**

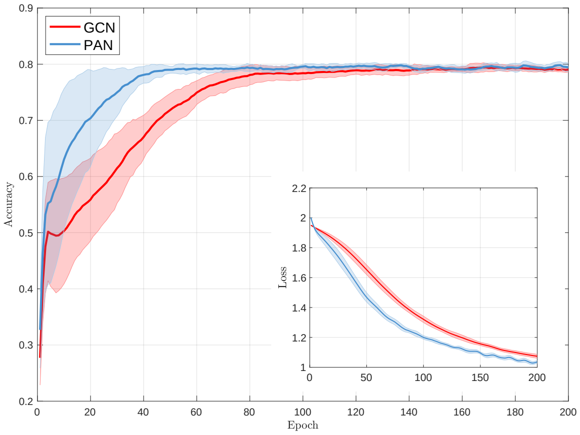


Figure 2. Main figure: Mean and standard deviation of validation accuracies of PAN and GCN on Cora. Figure in lower right corner: Validation loss function of PAN and GCN.

In general, whether maximal entropy random walk or GRW based model performs better depends on the nature of the graph. GRW may underestimate the contribution of the "influencer" in a small-world network (Newman et al., 2011), due to the fact that information sent from an "influencer" is diluted as a result of the large degree it has (Kampffmeyer et al., 2019). But in the maximal entropy random walk model, information transmitted in a path from the sender to the receiver is not affected by the degree of the sender.

5. Conclusions

We present a new graph convolution framework based on the path integral idea, which realizes the attention-like mechanism while preserves the simple form similar to GCN. Although we focus on maximal entropy random walk, our framework can easily accommodate other types of walks including many previous models. Preliminary results on node classification tasks show that our method achieves state-of-the-art performance very efficiently. Many extensions of the present work can be expected, including those for graph classification tasks.

References

Abu-El-Haija, S., Kapoor, A., Perozzi, B., and Lee, J. N-gcn: Multi-scale graph convolution for semi-supervised node classification. *arXiv preprint arXiv:1802.08888*, 2018.

Atwood, J. and Towsley, D. Diffusion-convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 1993–2001, 2016.

Belkin, M., Niyogi, P., and Sindhwani, V. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7(Nov):2399–2434, 2006.

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

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations*, 2014.

Burda, Z., Duda, J., Luck, J.-M., and Waclaw, B. Localization of the maximal entropy random walk. *Physical review letters*, 102(16):160602, 2009.

Chen, J., Ma, T., and Xiao, C. Fastgcn: fast learning with graph convolutional networks via importance sampling. In *International Conference on Learning Representations*, 2018.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pp. 3844–3852, 2016.

Eliav, B. and Cohen, E. Bootstrapped graph diffusions: Exposing the power of nonlinearity. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2 (1):10, 2018.

Feynman, R. and Mechanics, A. H. Q. Path integrals. *Lecture Notes Phys*, 106, 1979.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1263–1272. JMLR. org, 2017.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864. ACM, 2016.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pp. 1024–1034, 2017.

Kampffmeyer, M., Chen, Y., Liang, X., Wang, H., Zhang, Y., and Xing, E. P. Rethinking knowledge graph propagation for zero-shot learning. *International Conference on Learning Representations*, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Li, R.-H., Yu, J. X., and Liu, J. Link prediction: the power of maximal entropy random walk. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pp. 1147–1156. ACM, 2011.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

Liao, R., Zhao, Z., Urtasun, R., and Zemel, R. S. Lanczosnet: Multi-scale deep graph convolutional networks. In *International Conference on Learning Representations*, 2019.

Lu, Q. and Getoor, L. Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning*, pp. 496–503, 2003.

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5115–5124, 2017.

Newman, M., Barabasi, A.-L., and Watts, D. J. *The structure and dynamics of networks*, volume 12. Princeton University Press, 2011.

Ochab, J. and Burda, Z. Maximal entropy random walk in community detection. *The European Physical Journal Special Topics*, 216(1):73–81, 2013.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 701–710. ACM, 2014.

Such, F. P., Sah, S., Dominguez, M. A., Pillai, S., Zhang, C., Michael, A., Cahill, N. D., and Ptucha, R. Robust spatial filtering with graph convolutional neural networks. *IEEE Journal of Selected Topics in Signal Processing*, 11(6): 884–896, 2017.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pp. 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

Thekumparampil, K. K., Wang, C., Oh, S., and Li, L.-J. Attention-based graph neural network for semi-supervised learning. *arXiv preprint arXiv:1803.03735*, 2018.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018a.

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., and Hjelm, R. D. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018b.

Verma, N., Boyer, E., and Verbeek, J. Dynamic filters in graph convolutional networks. *arXiv preprint arXiv:1706.05206*, 2017.

Weston, J., Ratle, F., Mobahi, H., and Collobert, R. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pp. 639–655. Springer, 2012.

Wu, F., Zhang, T., Souza Jr, A. H. d., Fifty, C., Yu, T., and Weinberger, K. Q. Simplifying graph convolutional networks. *arXiv preprint arXiv:1902.07153*, 2019.

Xu, B., Shen, H., Cao, Q., Qiu, Y., and Cheng, X. Graph wavelet neural network. In *International Conference on Learning Representations*, 2019.

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

Zhu, X., Ghahramani, Z., and Lafferty, J. D. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pp. 912–919, 2003.

A. Experiments

Full list of baselines. We test our method on three public available citation graph datasets: Citeseer, Cora and Pubmed, with the fixed data splits performed in (Yang et al., 2016; Kipf & Welling, 2017). We compare our proposed model with 24 existing methods: MoNet (Monti et al., 2017), node2vec (Grover & Leskovec, 2016), Diffusion-CNN (DCNN) (Atwood & Towsley, 2016), GCN model (Kipf & Welling, 2017) and some baselines as compared in their work: MLP, ChebNet (Defferrard et al., 2016), label propagation (LP) (Zhu et al., 2003), manifold regularization (ManiReg) (Belkin et al., 2006), semi-supervised embedding (SemiEmb) (Weston et al., 2012), skip-gram based graph embeddings (DeepWalk) (Perozzi et al., 2014), the iterative classification algorithm (ICA) (Lu & Getoor, 2003) and Planetoid (Yang et al., 2016). To make a comprehensive comparison, we also include the recent proposed Graph-CNN (Such et al., 2017), DynamicFilter (Verma et al., 2017), FastGCN (Chen et al., 2018), graph linear network (GLN) (Thekumparampil et al., 2018), attention-based models, such as AGNN (Thekumparampil et al., 2018) and GAT (Veličković et al., 2018a), deep graph infomax (DGI) (Veličković et al., 2018b), Bootstrap (Eliav & Cohen, 2018), multi-scale deep graph convolutional networks such as LNet and AdaLNet (Liao et al., 2019), simplified GCN model (SGC) (Wu et al., 2019), and graph wavelet neural network (GWNN) (Xu et al., 2019).

Experimental settings. In our experiments, we apply the same model architecture as used in (Kipf & Welling, 2017), i.e., a two-layer model with 16 hidden neurons in the first hidden layer. Our models for all these three datasets are trained by Adam SGD optimizer (Kingma & Ba, 2014) with an initial learning rate 0.01, where Glorot strategy (Glorot & Bengio, 2010) is used for weights initialization. The maximum number of epochs for Cora and Citeseer is 200, while for Pubmed is 100. Dropout rate is set as 0.5 for Cora and Citeseer while 0.4 for Pubmed. Weight decay is set as 5e-3 for Cora, 1e-2 for Citeseer, 3e-3 for Pubmed. We use the same early stopping strategy on validation loss (Kipf & Welling, 2017) with a patience of 50 epochs for Cora and Citeseer, and 15 epochs for Pubmed. Results of PAN in Table 1 and 2 are obtained based on the above parameter setting.

Variations of PAN. We test the performance of some variations of PAN. Depending on the normalization method for adjacency matrix A and D_n which is the degree matrix for A^n, we propose totally seven different versions of PAN in (9). In the following, $\tilde{D}_n$ is the degree matrix for $\tilde{A}^n$ with $\tilde{A} = A + I$, $\hat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$, and $\tilde{D}$ is the degree matrix for $\tilde{A}$.

Table 2. Summary of results in terms of classification accuracies in percentage (%), for Cora, Citeseer and Pubmed, with a fixed (public) split of data from (Kipf & Welling, 2017). '-' stands for the missing values when the existing work does not test the associated dataset.

Method	**Cora**	**Citeseer**	**Pubmed**	**Method**	**Cora**	**Citeseer**	**Pubmed**
LP	68.0	45.3	45.3	Graph-CNN	76.3	-	-
ICA	75.1	69.1	73.9	DynamicFilter	81.6	-	79.0
ManiReg	59.5	60.1	70.7	FastGCN	81.8	-	77.6
SemiEmb	59.0	59.6	71.7	GLN	81.2	70.9	78.9
DeepWalk	67.2	43.2	65.3	AGNN	83.1	71.1	79.9
ChebNet	81.2	69.8	74.4	GAT	83.0	72.5	79.0
DCNN	76.8	-	73.0	Bootstrap	78.4	53.6	78.8
node2vec	74.9	54.7	75.3	DGI	82.3	71.8	76.8
Planetoid	75.7	64.7	77.2	LNet	79.5	66.2	78.3
MoNet	81.7	-	78.8	AdaLNet	80.4	68.7	78.1
MLP	55.1	46.5	71.4	SGC	81.0	71.9	78.9
GCN	81.5	70.3	79.0	GWNN	82.8	71.7	79.1
PAN	**82.0**	**71.2**	**79.2**				

Method 1

$$X^{(h+1)} = Z^{-1} \sum_{n=0}^{L} e^{-\frac{E(n)}{T}} A^n X^{(h)} W^{(h)}$$

Method 2

$$X^{(h+1)} = Z^{-1/2} \sum_{n=0}^{L} e^{-\frac{E(n)}{T}} A^n Z^{-1/2} X^{(h)} W^{(h)}$$

Method 3

$$X^{(h+1)} = \sum_{n=0}^{L} k^{(h)}(n) D_n^{-1} A^n X^{(h)} W^{(h)}$$

Method 4

$$X^{(h+1)} = \sum_{n=0}^{L} k^{(h)}(n) \tilde{D}_n^{-1} \tilde{A}^n X^{(h)} W^{(h)}$$

Method 5

$$X^{(h+1)} = \sum_{n=0}^{L} k^{(h)}(n) \hat{A}^n X^{(h)} W^{(h)}$$

Method 6

$$X^{(h+1)} = \sum_{n=0}^{L} k^{(h)}(n) D_n^{-1/2} A^n D_n^{-1/2} X^{(h)} W^{(h)}$$

Method 7

$$X^{(h+1)} = Z^{-1} \sum_{n=0}^{L} e^{-\frac{E(n)}{T}} \hat{A}^n X^{(h)} W^{(h)}$$

The results of Methods 1–7 are shown in Table 3, where backpropagation (BP) algorithm is used for training the weights $k^{(h)}(n)$ while in the bottom line we use grid search to seek for the optimal $k^{(h)}(n)$. Based on our empirical study, the highest accuracies for all these datasets are achieved at $k^{(h)}(0) = 0$, $k^{(h)}(1) = k^{(h)}(2)$, as shown in the last row of Table 3. Method 2 also appears to be a strong candidate. More theoretical analysis on why these models are favourable is expected in our future work.

Table 3. Performance comparison for Method 1-7 on Cora, Citeseer and Pubmed with $L = 2$.

Method	**Cora**	**Citeseer**	**Pubmed**
Method 1	80.7	69.2	77.7
Method 2	81.3	70.1	77.7
Method 3	80.8	69.2	78.8
Method 4	80.9	68.9	78.8
Method 5	79.8	66.6	75.3
Method 6	80.1	68.5	75.8
Method 7	80.8	69.6	78.4
Method 5 (grid search)	**82.0**	**71.2**	**79.2**