Fractional Graph Convolutional Networks (FGCN) for Semi-Supervised Learning

Anonymous authors
Paper under double-blind review

Abstract

Due to high utility in many applications, from social networks to blockchain to power grids, deep learning on non-Euclidean objects such as graphs and manifolds continues to gain an ever increasing interest. Most currently available techniques are based on the idea of performing a convolution operation in the spectral domain with a suitably chosen nonlinear trainable filter and then approximating the filter with finite order polynomials. However, such polynomial approximation approaches tend to be both non-robust to changes in the graph structure and to capture primarily the global graph topology. In this paper we propose a new Fractional Generalized Graph Convolutional Networks (FGCN) method for semi-supervised learning, which casts the Lévy Fights into random walks on graphs and, as a result, allows to more accurately account for the intrinsic graph topology and to substantially improve classification performance, especially for heterogeneous graphs.

1 Introduction

Adaptation of deep learning (DL) to graphs and other non-Euclidean objects has recently witnessed an ever increasing interest, leading to the new subfield of *geometric deep learning*. In particular, geometric deep learning is an emerging direction in machine learning which aims at generalizing the concepts of deep learning for data in non-Euclidean spaces, e.g., graphs and manifolds, by bridging the gap between graph theory and deep neural networks (see discussion by Bronstein et al., 2017; Monti et al., 2017; Monti, 2019, and references therein).

Many such DL approaches for non-Euclidian objects are based on the idea of performing a convolution operation in the spectral domain with a suitably chosen nonlinear trainable filter. As a result, node features are mapped into some Euclidian space. Next, graph filters are approximated with various finite order polynomials, e.g., Chebyshev polynomials (i.e., the ChebNet model family of Defferrard et al., 2016; Kipf & Welling, 2017), Cayley transform (i.e., CayleyNet of Levie et al., 2018) or the generalization of polynomial filters in a form of Auto-Regressive Moving Average (ARMA) models (Bianchi et al., 2019). However, deep learning approaches based on approximation with finite order polynomials tend to be non-robust to even minor changes in the graph structure and to largely disregard the local graph topology that often plays the critical role for learning on heterogeneous graphs. In contrast, as noted by Bianchi et al. (2019), one of the primary benefits of the ARMA filters over polynomial ones is that ARMA filters are not computed in the Fourier space induced by a graph Laplacian, and as a result, ARMA filters are local in the node space and enable to more flexibly and accurately capture the underlying graph topology.

In this paper we further advance this localized approach to DL on graphs and propose a fractional generalized graph-based convolutional filter for semi-supervised learning, which casts the Lévy Fights into random walks on graphs. As a result, our new Fractional Generalized Graph Convolutional Networks (FGCN) method allows to more accurately account for the intrinsic local graph topology and to substantially improve classification performance, especially for heterogeneous graphs.

The key contributions of our paper can be summarized as follows:

- we propose a new fractional generalized graph-based convolutional filter for semi-supervised learning, which casts the Lévy Fights into random walks on graphs and, as a result, provides a more efficient exploration of the graph structure.

- we develop a new Fractional Generalized Graph Convolutional Networks (FGCN) method that substantially improves accuracy of node classification for graphs of smaller sizes and with a lower number of node features.

- the proposed architecture of FGCN uses two state-of-the-art operations – gated max-average pooling and residual block. The architecture has been shown to significantly improve the training convergence and model output stability.

2 RELATED WORK

Many earlier semi-supervised learning approaches on graphs, e.g., Gaussian mixture models, co-training, harmonic function, and label propagation, employed only the label information (i.e., labeled instances) to train models based on the smoothness assumption over the labels (Zhu & Goldberg, 2009) and largely disregarded the graph structure. To improve performance, several learning methods on graphs propose to incorporate intrinsic "graph-based" information by designing a classifying function via generalizing the normalized cut and adding smooth function with respect to the intrinsic structure (Zhou et al., 2004; Zhou & Burges, 2007). A recent approach of Avrachenkov et al. (2012) proposes a generalized optimization framework through considering the above two methods as particular cases. However, one of the major limitations to these graph-based semi-supervised learning methods is disregarding important information contained in graph edges.

To address this disadvantage, Defferrard et al. (2016) propose a formulation of convolutional neural networks (CNN) based on spectral graph theory – ChebNet, which employs approximation via finite order polynomials and, in particular, is based in the Chebyshev expansion for fast filtering instead of the expensive eigen-decomposition. Graph Convolutional Networks (GCN) of Kipf & Welling (2017) simplifies ChebNet and further addresses the gradient vanishing problem and reduces the number of optimization. Other related approaches to graph learning with deep neural networks include, for instance, mixture model networks (MoNet) (Monti et al., 2017), graph attention networks (GAT) (Veličković et al., 2017), graph convolutional recurrent networks (Seo et al., 2018), dual graph convolutional networks (Zhuang & Ma, 2018), FastGCN (Chen et al., 2018), and simplified version of GCN (Wu et al., 2019).

To extend the success of GCN on undirected graphs to directed graphs, MotifNet of Monti et al. (2018) replaces the normalized Laplacian with the *motif Laplacian* in a multivariate polynomial filter, where the motifs information can help capture the network structure. Finally, the most recent approach of Bianchi et al. (2019) provides more flexible responses than GCN by using parallel and periodic concatenations of the convolutional kernel via the ARMA filter. As a result, the ARMA approach which is applicable to both directed and undirected networks allows to more accurately incorporate the underlying local graph structure into the graph learning process.

3 METHODOLOGY

Given a graph structure $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, W\}$, with $\mathcal{V}$ the set of nodes ($N = |\mathcal{V}|$) and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the set of edges between any two nodes. Let square matrix W represents the adjacency matrix of a graph whose entries $\{\omega_{ij}\}_{1 \leq i,j \leq N}$ are the edge weight function such that each edge $e_{ij} \in E$ has a weight ω_{ij}. We call W the symmetric adjacency matrix ($\omega_{ij} = 1$ if there is an edge between vertex v_i and vertex v_j) if we assume that graph $\mathcal{G}$ is undirected. In reality, however, undirected graph is a simplified representation of complex networks, we also consider directed graph (in which case $W^{\top} \neq W$) and we can easily transform the non-symmetric W to the symmetric one through $W^{'} = (W^{\top} + W)/2$.

We take $N \times Q$ feature matrix X as the input to a semi-supervised learning algorithm where Q is the number of different node features. Suppose we would like to classify N data points into K classes (communities), we define $N \times K$ label matrix Y as:

$$Y_{ik} = \begin{cases} 1, & \text{if vertex } i \text{ is labelled as a class } k, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Here we refer to each column $Y_{\cdot k}$ of matrix Y as a *labeling function*. Also define an $N \times K$ matrix F and call its columns $F_{\cdot k}$ *classification functions*.

3.1 GRAPH SIGNAL PROCESSING

In graph signal processing, a real-symmetric matrix has N real eigenvalues and its N real eigenvectors form an orthonormal basis. Given the symmetric adjacency matrix W of a graph, let D be the degree matrix where $d_{ii} = \sum_{j=1}^{N} w_{ij}$ and denote $L = U^\top \Lambda U$ the Standard Laplacian matrix, where $\Lambda = diag(\lambda_0, \ldots, \lambda_{N-1})$ and $U = [u_0, \ldots, u_{N-1}]$ is the matrix of eigenvectors.

In the following, we will revisit three popular semi-supervised learning methods - graph-based semi-supervised learning, fractional graph-based semi-supervised learning, and graph convolutional networks and gain new insights for improving their modeling capabilities.

Graph-based semi-supervised learning. Graph-based semi-supervised learning (G-SSL) has received much attention as an alternative approach to the population paradigm of supervised learning in recent years. G-SSL develops a generalized optimization framework, which has three particular cases (i) the Standard Laplacian (SL); (ii) Normalized Laplacian (NL); (iii) PageRank (PR). The optimization forulation with the following expression:

$$\min_F \left\{ 2F_{\cdot k}^T D^{\sigma-1} L D^{\sigma-1} F_{\cdot k} + \mu (F_{\cdot k} - Y_{\cdot k})^T D^{2\sigma-1} (F_{\cdot k} - Y_{\cdot k}) \right\}, \tag{2}$$

where μ is a regularization parameter. The Eq. 2 allows us to obtain the Standard Laplacian based formulation ($\sigma = 1$), the Normalized Laplacian formulation ($\sigma = \frac{1}{2}$), and PageRank formulation ($\sigma = 0$). The objective of the generalized optimization framework for G-SSL is a convex function and the corresponding classification function:

$$F_{\cdot k} = (1 - \alpha) \left(I - \alpha D^{-\sigma} W D^{\sigma-1} \right)^{-1} Y_{\cdot k}, \tag{3}$$

where $\alpha = \frac{2}{2+\mu}$ and for $k = 1, \ldots, K$.

Tuning the parameter σ on the power of degree matrix D, we can obtain three mentioned above particular semi-supervised learning methods:

- SL method ($\sigma = 1$): $F_{\cdot k} = (1 - \alpha) \left(I - \alpha D^{-1} W \right)^{-1} Y_{\cdot k}$

- NL method ($\sigma = \frac{1}{2}$): $F_{\cdot k} = (1 - \alpha) \left(I - \alpha D^{\frac{-1}{2}} W D^{\frac{-1}{2}} \right)^{-1} Y_{\cdot k}$

- PR method ($\sigma = 0$): $F_{\cdot k} = (1 - \alpha) \left(I - \alpha W D^{-1} \right)^{-1} Y_{\cdot k}$

From above formulations, the classification function F, i.e., the result of random walk process which provides the connection to the probabilistic interpretation of G-SSL. The parameter α controls the strength of the ground truth label matrix Y in the generalized optimization framework.

Fractional graph-based semi-supervised learning. To improve the classification performance (in particular, fuzzy graphs and unbalanced labeled data) of G-SSL, fractional graph-based semi-supervised learning (De Nigris et al., 2017) embeds Lévy Fights into random walks on graphs by constructing from powers of the Laplacian matrix, i.e., the L^γ operator. This operation can be used to generate different transition probabilities (i.e., corresponding to stochastic adjacency matrix) based on different γ values. Based on a fractional Laplacian matrix, $0 < \gamma < 1$, the anomalous (fractional) diffusion processes on networks can be constructed from the spectra data and eigenvectors of the Laplacian matrix. The fractional powers of L allows Lévy random walks with long-range navigation on a network. For example, the long-range transitions on a network can directly move node u and node v with the transition probability $m_{u \to v}^{(\gamma)}$ through a random walker, where $m_{u \to v}^{(\gamma)}$ is an element in the fractional transition matrix $\mathbf{M}^{(\gamma)}$ (see the graph within dotted circle in Figure 1). The transition probability $m_{u \to v}^{(\gamma)}$ between any two nodes whose geodesic distance is not infinite can be summarized as follow:

$$m_{u \to v}^{(\gamma)} = \delta_{uv} - \frac{(L^\gamma)_{uv}}{k_u^{(\gamma)}}, \tag{4}$$

where δ_{uv} represents the Kronecker delta, $k_u^{(\gamma)}$ denotes the fractional degree of the node u and $k_u^{(\gamma)} \equiv (L^\gamma)_{uu}$. Through replacing the L operator with $L^\gamma = U^\top \Lambda^\gamma U$, the new optimization formulation leaves us with the following expression:

$$\min_F \left\{ 2 F_{\cdot k}^T D_\gamma^{\sigma-1} L^\gamma D_\gamma^{\sigma-1} F_{\cdot k} + \mu (F_{\cdot k} - Y_{\cdot k})^T D_\gamma^{2\sigma-1} (F_{\cdot k} - Y_{\cdot k}) \right\}, \tag{5}$$

where $(D_\gamma)_{ii} = (L^\gamma)_{ii}$.

Let $0 < \gamma < 1$, the closed form solution for Eq. 5 can be obtain as fellow:

$$F_{\cdot k} = (1 - \alpha) \left(I - \alpha D_\gamma^{-\sigma} W_\gamma D_\gamma^{\sigma-1} \right)^{-1} Y_{\cdot k}, \tag{6}$$

for $k = 1, \ldots, K$. Therefore, we can conclude three particular fractional semi-supervised learning mehtods like G-SSL:

- Fractional SL method ($\sigma = 1$): $F_{\cdot k} = (1 - \alpha) \left(I - \alpha D_\gamma^{-1} W_\gamma \right)^{-1} Y_{\cdot k}$

- Fractional NL method ($\sigma = \frac{1}{2}$): $F_{\cdot k} = (1 - \alpha) \left(I - \alpha D_\gamma^{\frac{-1}{2}} W_\gamma D_\gamma^{\frac{-1}{2}} \right)^{-1} Y_{\cdot k}$

- Fractional PR method ($\sigma = 0$): $F_{\cdot k} = (1 - \alpha) \left(I - \alpha W_\gamma D_\gamma^{-1} \right)^{-1} Y_{\cdot k}$

3.2 Proposed Fractional Generalized Graph Convolutional Networks for Semi-Supervised Node Classification

Although both G-SSL and fractional G-SSL achieve comparable and consistent (low variance) performance on some datasets, e.g., Les Miserables, Wikipedia-math, and MNIST, these approaches consider only the given adjacency matrix W and the label matrix Y without using the feature matrix X. Such limitation is crucial especially when dealing with datasets that not only exhibit a sophisticated topological graph structure but also provide node feature information, such as citation, biological, financial, and power grid networks. To address this drawback, through using the feature matrix X instead of the label matrix Y and encoding the graph structure by using neural network framework, many graph-based neural networks methods, e.g., graph convolutional networks (GCN), have been recently proposed and have been shown to demonstrate very impressive advances in semi-supervised learning performance on graphs. Next, we turn to discussing the new Fractional Generalized Graph Convolutional Networks (FGCN) for semi-supervised node classification.

Fractional Generalized Graph Convolutional Networks (FGCN). The key idea behind our proposed method is Fractional Generalized Sigma-based (FGS) filter $g_{FGS}(\sigma, \gamma) = (1 - \alpha) \left(I - \alpha D_\gamma^{-\sigma} W_\gamma D_\gamma^{\sigma-1} \right)^{-1} = (1 - \alpha)(I - \alpha \tilde{L})^{-1}$. Let us insert Taylor series expansion into the expression of the FGS filter and this has the advantage of avoiding the inverse computation. We have the following expression:

$$g_{FGS}(\alpha, \sigma, \gamma) = (1 - \alpha) \left(I + (\alpha\tilde{L})^1 + (\alpha\tilde{L})^2 + (\alpha\tilde{L})^3 + \cdots \right) = (1 - \alpha) \sum_{i=0}^{\infty} \left(\alpha\tilde{L} \right)^i, \tag{7}$$

where $0 < \alpha \le 1, 0 \le \sigma \le 1, 0 < \gamma \le 1$. We then obtain the general classification function by multiplying the feature matrix X:

$$\bar{\mathcal{X}} = g_{FGS}(\alpha, \sigma, \gamma) X$$
$$= (1 - \alpha) \Big(\underbrace{X}_{\text{the first item}} + \underbrace{X + \alpha\tilde{L}X}_{\text{the second item}} + \underbrace{X + \alpha\tilde{L}X + \alpha^2\tilde{L}^2 X}_{\text{the third item}} + \cdots \Big), \tag{8}$$

where, recursively,

$$(X')_i = X + \alpha\tilde{L}(X')_{i-1}, \quad (X')_0 = X, \quad (i = 1, 2, \cdots). \tag{9}$$

Convolutional layer. During the training process, the convolutional model need to train the trainable parameters $(\mathbf{W}, \mathbf{b})$ of the graph filter which can scan the given input feature matrix into a series of

feature maps with neurons. Thereby, we provide an implementation of Eq. 8 as a FGS convolutional layer:

$$H^{(t+1)} = \sigma \left((1-\alpha) \sum_{i=0}^{\infty} \left(\alpha \tilde{L} \right)^i H^{(t)} \mathbf{W}^{(t)} \right), \qquad (10)$$

where $H^{(t+1)}$ is the hidden layer output matrix of activations in the t-th layer and $H^{(0)} = X$, $\sigma(\cdot)$ is the adopted activation function, and $\mathbf{W}^t$ is the trainable weight in the t-th layer. Besides, we borrow the concept of the parallel system in reliability theory to improve the consistency of our proposed method. A parallel system is a configuration that the entire system works as long as not all involved components in the system fail. Hence, the parallel system structure is more robust against noisy inputs, compared to a single system structure. Suppose there are n components with the reliability R_i (where $i = 1, \cdots, n$) in a parallel system, then the reliability of this parallel system R_{PS} can be obtained with the following expression:

$$R_{PS} = 1 - \prod_{i=1}^{n} (1 - R_i) = 1 - (1 - R_1) \times (1 - R_2) \times \cdots \times (1 - R_n). \qquad (11)$$

According to Eq. 11, we see the parallel system can be enhanced if there exists a component with strong reliability. In this way, we establish both theoretical and practical guarantees for our proposed model to reach stable over a large set of hyperparameters, small datasets, and noisy labels based on this parallel implementation.
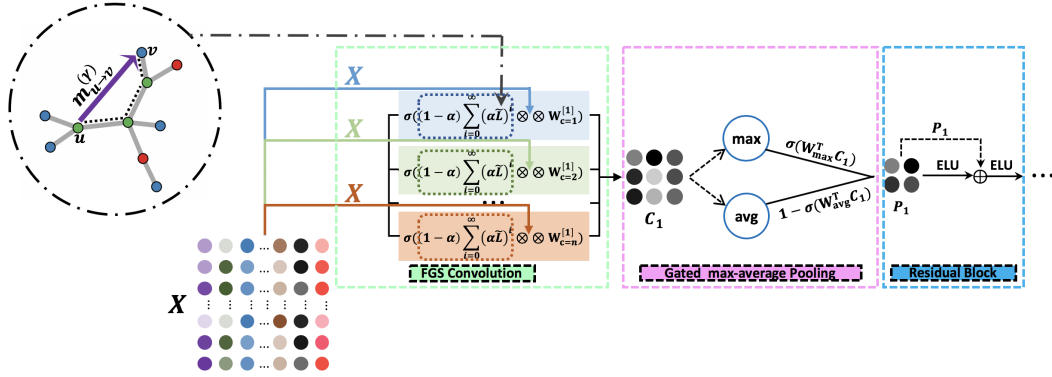

Figure 1: Illustration Fractional Generalized Graph Convolutional Network model. The input is the feature matrix X and the graph within dotted circle represents embedding Lévy Fights into random walks on graph (where L^γ is the Laplacian matrix L to a power γ). FGCN architecture consists of three main components: (i) FGS convolutional layer with parallel structure; (ii) gated max-average pooling layer; (iii) activation block for residual learning.

Pooling layer. When implementing the form of pooling operation to aggregate information from the outputs of parallel FGS convolutional layer, instead of using some popular pooling functions such as max and average pooling, we apply the state-of-art pooling operation - *gated max-average pooling* to capture the local and global information from all the nodes and graph structure. The rationale behind the *gated max-average pooling*, is that it consider "responsive" strategy (i.e., improving translation invariance and scale invariance via considering input in each gating mask) based on the *mixed max-average pooling* equation. We have:

$$f_{\text{gate}}(X_{FGS}) = \sigma \left(\mathbf{W}^\top X \right) f_{\max}(X_{FGS}) + \left(1 - \sigma \left(\mathbf{W}^\top X_{FGS} \right) \right) f_{\text{avg}}(X_{FGS}), \qquad (12)$$

where $\mathbf{W}$ is the trainable weight matrix, X_{FGS} is the output matrix from the parallel FGS convolutional layer after concatenation operation.

Residual building block. Inspired by residual learning, we apply a residual block by adding the skip connection after the pooling layer. One of the advantages of the residual learning is the *identity* mapping which provides a direct path for propagating information. When using the residual building block, we adopt a similar scheme in He et al. (2016) to deal with the output of the pooling layer. Let

$\mathcal{H}(x)$ be an underlying mapping and we cast it as $\mathcal{H}(x) = \mathcal{F}(x) + x$, where $\mathcal{F}(x)$ is the residual mapping defined by $\mathcal{H}(x) - x$. In other words, it is easier to optimize the residual mapping $\mathcal{F}(x)$ than optimizing the direct mapping $\mathcal{H}(x)$ and helps to avoid the gradient vanishing problem during training. To this end, we use an exponential linear unit (ELU) in direct mapping and place a rectified linear unit (ReLU) after addition in our model.

4 EXPERIMENTS

4.1 DATASETS

Undirected networks. Cora-ML (this Cora dataset consists of Machine Learning papers), Cite-Seer and PubMed are three standard citation networks benchmark datasets used for semi-supervised learning evaluation. In these citation networks, nodes represent publications, edges denote citation, the input feature matrix are bag of words and label matrix contain the class label of publication. We use the same data format as GCN, i.e., 20 labels per class in each citation network.

Directed networks. We evaluate our method on two directed networks - Cora and IEEE 118-bus system. Unlike the small subset of the Cora-ML dataset (above), Cora is extracted from original dataset which contains more than $50,000$ computer science research papers, where nodes represent scientific papers and directed edges between node u and node v ($u \rightarrow v$) represent citation of paper v in paper u. In addition to citation networks, we also test our method on power grid network, i.e., IEEE 118-bus system. We consider a graph as a model for the IEEE 118-bus system where nodes represent units such as generators, loads and buses, and edges represent the transmission lines. The input features of power grid network are generator active power upper bound (PMAX) and real power demand (PD) obtained from MATPOWER case struct. On directed networks, we test them separately from two scenarios: (i) on Cora, the number of features are decomposed into a low dimension (130 components) based on principal component analysis (PCA), the dataset is trained with 10% label rate and evaluated in another 10% fraction whose labels are withheld; (ii) on IEEE 118-bus system, 10% label rate for training set, 20% for validation and 70% for test sets. The statistics of data we used in the experimental section are summarized in Table 1.

Table 1: Dataset statistics.

Dataset	Vertices	Edges	Features	Classes	Label rate	Train/Valid/Test
Cora-ML	2,708	5,429	1,433	7	0.052	140 / 500 / 1,000
CiteSeer	3,327	4,732	3,703	6	0.036	120 / 500 / 1,000
PubMed	19,717	44,338	500	3	0.003	60 / 500 / 1,000
Cora	19,793	65,311	8,710	70	0.100	1,979 / 1,979 / 1,979
IEEE 118-Bus	118	182	2	3	0.100	11 / 35 / 70

4.2 SETUP AND BASELINES

Baselines. In this setting, on undirected networks, we compare FGCN with the state-of-the art semi-supervised classification approaches including (i) using the label matrix as input: label propagation (LP) (Zhu & Ghahramani, 2002); (ii) using the feature matrix as input: DeepWalk (DW)(Perozzi et al., 2014), graph attention networks (GAT), GNN with ChebNet polynomials filter (ChebNet), GCN, and GNNs with convolutional ARMA filters (ARMA). In addition, on undirected networks, we use MotifNet (using symmetric motif adjacency matrix instead of Normalized Laplacian matrix in multivariate polynomial filters), ChebNet, GCN, and ARMA as the benchmarks.

Training setting details. The training is done by using Adam optimizer with learning rate $lr_1 = 0.01$ for undirected networks and $lr_2 = \{0.1; 0.01\}$ for directed networks. To prevent the model from overfitting, we consider both adding dropout layer before two graph convolutional layers and kernel regularizers (ℓ_2) in each layer. For undirected networks: we follow the experimental setup in Kipf & Welling (2017) to set the parameters of baselines (GAT, ChebNet, GCN and ARMA) by using two graph convolutional layers with 16 hidden units, ℓ_2 regularization term with coefficient 5×10^{-4}, and dropout probability $p_{drop.}$ of 0.5 (for ARMA, except for number of hidden units, the

hyperparameters setting are significantly different from others). For directed networks: we consider the MotifNet setting with using dropout rate $p_{drop.}$ of 0.5, ℓ_2 regularization term with coefficient 0.01, and the degree of multivariate polynomials $k = 4$ for MotifNet; for other three baselines (i.e., ChebNet, GCN, and ARMA), we also use a two-layer network but with 64 hidden units, learning rate lr_2 with 0.1, 0.5 dropout rate and ℓ_2 regularization weight of 5×10^{-4}. In the following, Table 2 displays the best hyperparameter configurations of FGCN for each dataset by using standard grid search mechanism (the optimal kernel regularization weight ℓ_2 always equal to 5×10^{-4}).

Table 2: Hyperparameters setting of FGCN

Dataset	$p_{drop.}$	$n_h^{[1]}$	α	σ	γ	n_{FGS}	$\{\#\mathcal{C}^{[1]}; \#\mathcal{C}^{[2]}\}$	$\phi(RB)$
Cora-ML	0.75	32	0.6	0.50	1.000	7	{5;2}	ELU + ReLU
CiteSeer	0.75	32	0.6	0.53	1.000	7	{3;2}	ELU + ELU
PubMed	0.00	32	0.6	0.50	1.000	7	{5;2}	ELU + ReLU
Cora	0.00	128	0.4	0.90	1.000	10	{2;1}	ELU + ELU
IEEE 118-Bus	0.00	128	0.1	0.20	0.004	40	{7;2}	ELU + ELU

* $n_h^{[1]}$ denotes the number of hidden units in the first layer; $\#\mathcal{C}^{[1]}$ and $\#\mathcal{C}^{[2]}$ represents the number of statistically independent parallel components in the first layer and the second layer respectively; n_{FGS} denotes the number of items in Taylor series polynomial expansion of FGS filter; $\phi(RB)$ represents the type of activation functions in residual block structure.

5 RESULTS

Performance comparison of semi-supervised node classification. Table 3 reports the average accuracy delivered by FGCN and competing methods calculated over 100 runs for fixed train/valid/test sizes (see Table 1). The best performance for each dataset is marked in bold.

Table 3 shows that FGCN outperforms all competing approaches in all data sets, except for CiteSeer where FGCN delivers the second best accuracy result after the ARMA approach and the difference between FGCN and ARMA for CiteSeer is less than 0.05%. The improvement gain of FGCN over the next most accurate method ranges from 0.70% (for undirected PubMed over GCN) to 16.5% (for directed IEEE 118-Bus over ARMA). Remarkably, methods that are applicable both to undirected and directed networks (i.e., ChebNet, GCN, and ARMA) tend to deliver noticeably lower accuracy results for a directed networks, while the new FGCN method yields a more stable performance across both directed and undirected networks. Note also that directed networks tend to exhibit lower label rates than undirected ones, and all methods yield the best accuracy performance for undirected Cora-ML with a label rate of 0.052 and the lowest performance for directed Cora with a label rate of 0.100 – in both these cases FGCN outperforms all other competing approaches.

Furthermore, the IEEE 118-Bus data set is the smallest among the considered data (see Table 1), and we might expect to observe lower accuracy results for this data set due to a limited training set. However, the accuracy yielded by FGCN is among the highest ones across all data sets.

Evaluation of FGCN-specific parameters. During grid search over three parameters (i.e., α, σ, and γ), we find that: (i) the regularization parameter α which used to specify the relative importance of a graph in clustering strongly relates to the probability of initial conditions for random walks when the self-refreshing process works, and it strongly influences the network's generalization ability and node classification performance for all datasets; (ii) the free unifying parameter σ provides enough flexibility to construct a canonical formulation of different graph-based semi-supervised methods. Table 3 indicates that the optimal value of σ depends on both the types of networks (undirected and directed) and label rate not on the size of network; (iii) the fractional power parameter γ substantially impacts the accuracy of node classification for the small datasets (see e.g., Figure 2), however, no similarly strong influence is found in the larger datasets.

Sensitivity analysis. In the sensitivity analysis setting, we have the ability to analyze the sensitivity of the node classification accuracy to variation from three FGCN-specific parameters - $\alpha \in \{0.1, \cdots, 1\}$, $\sigma \in \{0, 0.1, \cdots, 1\}$, and $\gamma \in \{0.001, 0.01, 0.1, 1\}$. In this case, we only show the results from sensitively analysis for FGCN model on IEEE 118-bus dataset. First, we perform the parameter learning experiments on four scenarios with a fixed parameter γ. Figure 2 shows

Table 3: Comparison of average accuracy (%) of semi-supervised classification approaches for both undirected networks and directed networks.

Method	Undirect networks			Directed networks	
	Cora-ML	CiteSeer	PubMed	Cora	IEEE 118-Bus
LP	68.70	46.32	65.92	-	-
DW	67.20	43.27	65.33	-	-
MotifNet	-	-	-	60.00	65.75
GAT	83.11	70.85	78.56	-	-
ChebNet	81.45	70.23	78.40	58.93	60.00
GCN	81.50	71.11	79.00	57.75	52.86
ARMA	82.80	**72.30**	78.80	58.99	70.55
FGCN (ours)	**84.35**	71.89	**79.56**	**60.70**	**82.20**

that the accuracy substantially decreases when α is larger than 0.8, especially in γ equals to 0.001 and 0.01 (see Figure 2a, 2b). Setting $\gamma = \{0.1, 1\}$, we observe that the classification accuracy nearly monotonic decreases while increasing α. Additionally, there is little difference of classification accuracy between $\gamma = 0.001$ and $\gamma = 0.01$ and FGCN generally gives consistent and higher accuracy when the α parameter is within the range of $\{0.1, 0.2, 0.3, 0.4\}$. We then explore the variation of accuracy based on tuning parameter γ within the range of $[0.001, 0.002, \cdots, 0.01]$ (setting $\sigma \in \{0, 0.1, \cdots, 1\}$ at the same time), however, it is hard to obtain the optimal $(\hat{\sigma}, \hat{\gamma})$ combination through gathering finite experimental results (100 runs) since some of the results are very close. Therefore, we run the following experiments to demonstrate the impact evaluation of γ:

- *Step 1:* Setting $\gamma \in \{0.001, 0.002, \cdots, 0.1\}$.
- *Step 2:* For each γ (i.e., fix the γ value in each experiment), we run our proposed model 100 times separately for σ from 0 to 1 by 0.01; Then we can obtain the 100 average accuracies for each γ under fixed γ, that is, $\{AAC_{\sigma=0}, ACC_{\sigma=0.01}, \cdots, AAC_{\sigma=1}\}^{[i]}$, where $i = 1, \cdots, 10$.
- *Step 3:* Fitting the Gaussian distribution to $\{AAC_{\sigma=0}, ACC_{\sigma=0.01}, \cdots, AAC_{\sigma=1}\}^{[i]}$ (see Figure 3b).

Similar to γ, we can fit the Gaussian distribution to $\{AAC_{\gamma=0.001}, ACC_{\gamma=0.002}, \cdots, AAC_{\gamma=0.01}\}^{[j]}$ by fixing σ, where $j = 1, \cdots, 11$ (see Figure 3a).

From Figure 3, we find that there exist larger difference between the shapes of approximate Gaussian distributions by fixing the parameter σ than fixing the parameter γ which means the parameter σ is more important factor in FGCN approach for small datasets.

6 CONCLUSION

In this paper we have proposed a new Fractional Generalized Graph Convolutional Networks (FGCN) method to semi-supervised learning on graphs that enables to better capture the intrinsic local graph topology. The key idea behind our new approach is a new fractional generalized graph-based convolutional filter which casts the Lévy Fights into random walks on graphs and, as a result, allows for a more efficient, accurate and robust exploration of the local graph structure which often plays the key role in graph learning performance, especially for heterogeneous graphs.

Our numerical studies have indicated that the new FGCN method tends to outperform all competing deep learning approaches on both directed and undirected graphs in all considered data sets, except of CiteSeer for which the difference between the best result delivered by the ARMA filter and FGCN is less than 0.05%. The gain in learning accuracy of FGCN over the next best competitor ranges from 0.70% to 16.5%, and the highest gain has been achieved for the IEEE 118-Bus data set which is the smallest among the considered data sets. Furthermore, in contrast to the competing approaches, FGCN tends to deliver a more stable performance across directed and undirected networks regardless of the label rate.
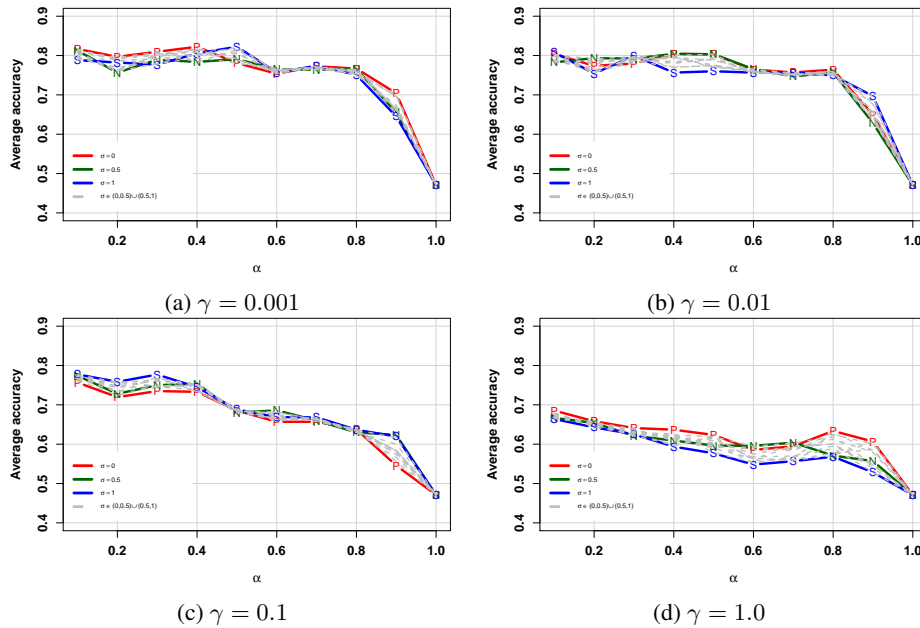
Figure 2: Accuracy of FGCN for PageRank, Normalized Laplacian, and Standard Laplacian, which depends on the function of α and fixes the fractional power $\gamma = \{0.001$ (a), 0.01 (b), 0.1 (c), 1.0 (d)$\}$
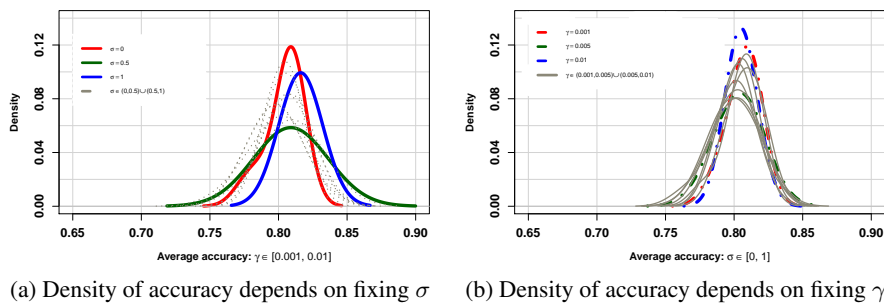

Figure 3: Generalized Gaussian density of accuracy of FGCN: (a) the solid curves represent three particular scenarios in FGCN, where the red curve denotes the PR-based method ($\sigma = 0$), green curve denotes the NL-based method ($\sigma = 0.5$), and blue curve denotes the SL-based method ($\sigma = 1$); the dotted curves represent common scenarios in FGCN model, that is, the generalized parameter σ falls within the range of $(0, 0.5) \cup (0.5, 1)$. (b) the red, green, blue dotted curve represents the scenario when setting the fractional parameter γ equals to 0.001, 0.005, and 0.01, respectively; The solid curves represent scenarios where γ in the range of $(0.001, 0.005) \cup (0.005, 0.01)$.

In the future we plan to advance the proposed FGCN technique to learning on multilayer networks and to enhance graph learning process with topological information on the underlying deep neural network.

REFERENCES

K. Avrachenkov, A. Mishenin, P. Gonçalves, and M. Sokol. Generalized optimization framework for graph-based semi-supervised learning. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, pp. 966–974. SIAM, 2012.

Filippo Maria Bianchi, Daniele Grattarola, Lorenzo Livi, and Cesare Alippi. Graph neural networks with convolutional arma filters. *arXiv preprint arXiv:1901.01343*, 2019.

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

J. Chen, T. Ma, and C. Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018.

S. De Nigris, E. Bautista, P. Abry, K. Avrachenkov, and P. Gonçalves. Fractional graph-based semi-supervised learning. In *25th EUSIPCO*, pp. 356–360, 2017.

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

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

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

F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *CVRPR*, pp. 5115–5124, 2017.

F. Monti, K. Otness, and M. M. Bronstein. Motifnet: a motif-based graph convolutional network for directed graphs. In *2018 IEEE DSW*, pp. 225–228, 2018.

F. et al. Monti. Geometric deep learning, 2019. http://geometricdeeplearning.com/.

B. Perozzi, R. Al-Rfou, and S. Skiena. 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.

Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing*, pp. 362–373. Springer, 2018.

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

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

D. Zhou and C. JC Burges. Spectral clustering and transductive learning with multiple views. In *ICML*, pp. 1159–1166, 2007.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, pp. 321–328, 2004.

X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. 2002.

X. Zhu and A. B. Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

C. Zhuang and Q. Ma. Dual graph convolutional networks for graph-based semi-supervised classification. In *Proceedings of the 2018 World Wide Web Conference*, pp. 499–508. International World Wide Web Conferences Steering Committee, 2018.