Epidemic Graph Convolutional Network

Tyler Derr
Data Science and Engineering Lab
Michigan State University
derrtyle@msu.edu

Yao Ma
Data Science and Engineering Lab
Michigan State University
mayao4@msu.edu

Wenqi Fan
Department of Computer Science
City University of Hong Kong
wenqifan03@gmail.com

Xiaorui Liu
Data Science and Engineering Lab
Michigan State University
xiaorui@msu.edu

Charu Aggarwal
IBM T.J. Watson Research Center
charu@us.ibm.com

Jiliang Tang
Data Science and Engineering Lab
Michigan State University
tangjili@msu.edu

ABSTRACT

A growing trend recently is to harness the structure of today's big data, where much of the data can be represented as graphs. Simultaneously, graph convolutional networks (GCNs) have been proposed and since seen rapid development. More recently, due to the scalability issues that arise when attempting to utilize these powerful models on real-world data, methodologies have sought the use of sampling techniques. More specifically, minibatches of nodes are formed and then sets of nodes are sampled to aggregate from in one or more layers. Among these methods, the two prominent ways are based on sampling nodes from either a local or global perspective. In this work, we first observe the similarities in the two sampling strategies to that of epidemic and diffusion network models. Then we harness this understanding to fuse together the benefits of sampling from both a local and global perspective while alleviating some of the inherent issues found in both through the use of a low-dimensional approximation for the path-based Katz similarity measure. Our proposed framework, Epidemic Graph Convolutional Network (EGCN), is thus able to achieve improved performance over sampling from just one of the two perspectives alone. Empirical experiments are performed on several public benchmark datasets to verify the effectiveness over existing methodologies for the node classification task and we furthermore present some empirical parameter analysis of EGCN.

KEYWORDS

Graph Neural Networks; Node Classification; Epidemic Models

ACM Reference Format:
Tyler Derr, Yao Ma, Wenqi Fan, Xiaorui Liu, Charu Aggarwal, and Jiliang Tang. 2020. Epidemic Graph Convolutional Network. In *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20), February 3–7, 2020, Houston, TX, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3336191.3371807

WSDM '20, February 3–7, 2020, Houston, TX, USA

ACM ISBN 978-1-4503-6822-3/20/02…$15.00
https://doi.org/10.1145/3336191.3371807

1 INTRODUCTION

Recently much of the big data that is being created inherently has an underlying structure. Thus, a growing trend is to try and harness this structure by representing the data in the form of graphs, such that hopefully more meaningful analysis and predictions can be made from the data. Simultaneously, graph convolutional networks (GCNs) [7, 13, 17, 22, 29, 37], which are adaptations from the classical convolutional neural networks[31] for graph structured data, have seen rapid development. This is primarily due to their fruitful usage in a variety of tasks such as node classification[11, 22, 29], link prediction[16, 39, 46], community detection[6, 8, 10], and recommendation [19, 42, 44].

In [29] a GCN was presented that was able to achieve superior performance in the semi-supervised node classification tasks, but lacked the scalability to handle larger networks. Thus, recent work has focused on sampling methods that can be used to allow these powerful neural network models to be applicable to more real-world networks. More specifically, these more recent GCNs are trained in minibatch form and utilize the sampling techniques to provide a set of nodes to aggregate from. While these methods can still be applied in the typical GCN format of having multiple layers of aggregation (e.g., typically two layers are used) they both have their unique nuances that can both benefit and unfortunately hinder the training process towards obtaining a well-performing trained model.

Among the sampling methods used, the two most prominent are based on sampling nodes at either a local or global perspective. GraphSAGE [22] was the first work towards performing a localized sampling for each minibatch node. Their main idea is to first construct a minibatch, and then for each of the nodes, they uniformly sample a set of up to s nodes from their set of neighbors, which is recursively applied for each of the layers. In this fashion, only a localized view is obtained and dependent on the number of layers in the model. However, the number of samples required grows exponentially in terms of s with the number of layers. Furthermore, even with more layers, this results in a smoothing effect over the aggregated signal due to a node aggregating from such a large portion of the graph. Thus, FastGCN [9] was introduced that performs a global sampling, which mainly focused on alleviating the issue related to the exponential growth in the number of required locally sampled nodes. More specifically, their sampling is performed upon for each layer by independently sampling a set of nodes according to a global distribution of node importance. Thus, rather than sampling for each node according to their local neighborhood, they instead

focus on sampling a subset of globally important nodes and then aggregating from that set. However, although the sample size grows linearly in terms of s with the number of layers, for larger graphs it becomes more difficult to obtain a meaningful global sample that is connected to the set of nodes in the minibatch. In this work, we seek to harness the power of both the local and global sampling, while also avoiding the pitfalls from each respective method.

Although it is clear that a better sampling methodology is desired, it is still unclear how to overcome the associated challenges in both the local and global methods. For example, in both sampling methods, we face the challenge of not being able to easily aggregate information from a larger perspective. With the local sampling we face the challenge that the size of the sampled set grows exponentially and also that due to a uniform sampling we would be susceptible to be influenced by noisy (or perhaps adversarially constructed) links. The global method in comparison, especially on larger graphs, would potentially struggle to have nodes in the minibatch connect to the sampled set due to requiring direct links to those sampled. Thus, we seek to solve these challenges while combining the two sampling methods by first drawing the connection of these methods to classical epidemic and diffusion network models [27, 36]. Then, given these insights, we formulate our proposed Epidemic Graph Convolutional Neural Network (EGCN).

Our proposed framework is built on utilizing a low-dimensional approximation of the path-based Katz similarity measure [25] towards merging the local and global sampling methodologies. More specifically, the use of this similarity measure is able to provide us with a mechanism for restricting the local sampling to the top-s most similar nodes, while also allowing for a more meaningful weighted aggregation according to this similarity measure (thus avoiding potentially spurious or noisy links). In the case of the global sampling, we harness the Susceptible-Infected-Susceptible (SIS) epidemic model [36] where those nodes currently infected are used as our sampled set. We present an analysis that shows those more often infected are nodes having higher centrality. Then, through the use of the approximated similarity measure, we are able to proportionally aggregate from the entire globally important set of infected nodes during the training process alleviating the disconnected problem in traditional global sampling. This allows our proposed Epidemic Graph Convolutional Network to aggregate from the most meaningful local neighborhood, while also incorporating weighted information from a globally important set of nodes. Our major contributions are listed as follows:

- Propose the use of both local and global sampling methods together in a principled approach that utilizes a low-dimensional approximation of the Katz similarity measure to overcome some of the challenges associated in each method.
- We analyze the relation between current GCN sampling methods to that of classical epidemic and diffusion models, where we then harness this knowledge towards constructing our proposed Epidemic Graph Convolutional Network (EGCN) framework.
- Conduct experiments on three real-world datasets to show the effectiveness of our proposed EGCN framework towards the node classification task, while also providing a parameter analysis to better understand EGCN.

The rest of the paper is organized as follows. In Section 2, we describe the node representation learning problem. Some background about GCNs as well as epidemic and diffusion modeling are provided in Section 3. In Section 4, we motive and introduce the proposed Epidemic GCN (EGCN). In Section 5 we perform experimental study and present some parameter analysis. Section 6 summarizes the related works and we conclude in Section 7.

2 PROBLEM STATEMENT

We aim to learn inductive node representations in a network for the node classification task. Thus, here we introduce the needed notations and formally define the problem.

A graph can be represented as $\mathcal{G}_a = \{\mathcal{V}_a, \mathcal{E}_a\}$, where $\mathcal{V}_a = \{v_{a1}, \ldots, v_{aN_a}\}$ is the set of N_a nodes and $\mathcal{E}_a = \{e_{a1}, \ldots, e_{aM_a}\}$ is the set of M_a edges. The connections between nodes in graph $\mathcal{G}_a$ can be also summarized as an adjacency matrix $\mathbf{A}_a \in \mathbb{R}^{N_a \times N_a}$, where $\mathbf{A}_{a[i,j]} = 1$ only when there is an edge between node v_{ai} and node v_{aj}, otherwise 0. Each node v_{ai} is associated with a feature vector $\mathbf{x}_{ai} \in \mathbb{R}^{1 \times d}$. The features for all the nodes can be summarized as a matrix $\mathbf{X}_a \in \mathbb{R}^{N_a \times d}$. Thus, more generally, a graph with features can be represented as $\mathcal{G}_a = \{\mathbf{A}_a, \mathbf{X}_a\}$. In the task of node classification, a portion of nodes have known labels and denoted as $\mathcal{V}$ (where $|\mathcal{V}| = N$) while the rest are assumed to be hidden. The set of nodes without a known label is $\mathcal{V}_h$, where $|\mathcal{V}| + |\mathcal{V}_h| = |\mathcal{V}_a| = N_a$. A label for a node $v \in \mathcal{V}$ can be denoted as y and the labels for all nodes in $\mathcal{V}$ can be denoted as $\mathbf{Y}$. We are to use the information of nodes in $\mathcal{V}$ to infer the labels for nodes in $\mathcal{V}_h$. The information of a node in $\mathcal{V}$ includes its label, its associated feature vector, and its connection with other nodes. The features for nodes in $\mathcal{V}$ can be denoted as $\mathbf{X} \in \mathbb{R}^{N \times d}$. Furthermore, a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ can be induced from the set of nodes $\mathcal{V}$, where $\mathcal{E}$ consists of all the edges that both their nodes are in $\mathcal{V}$. Correspondingly, the adjacency matrix for this induced graph can be denoted as $\mathbf{A} \in \mathbb{R}^{N \times N}$. During the training process, only the information of the nodes in $\mathcal{V}$ can be accessed, which includes $\mathbf{A}$, $\mathbf{X}$, and $\mathbf{Y}$. We aim to use $\mathbf{A}$, $\mathbf{X}$, and $\mathbf{Y}$ to train a model $f(\cdot, \cdot|\theta)$ such that it can perform label prediction for unseen nodes in $\mathcal{V}_h$. Mathematically, the process can be summarized as:

$$Q(\mathbf{A}, \mathbf{X}, \mathbf{Y}, f(\cdot, \cdot|\theta)) \rightarrow \hat{\theta}$$

where Q denotes the learning process and $\hat{\theta}$ denotes the learned parameters of model $f(\cdot, \cdot|\theta)$. We can then use the learned model to predict the labels for the unseen nodes in $\mathcal{V}_h$ as:

$$f(\mathbf{A}_a, \mathbf{X}_a|\hat{\theta}) \rightarrow \hat{\mathbf{Y}}_h$$

where $\hat{\mathbf{Y}}_h$ is the predicted labels for the unseen nodes $\mathcal{V}_h$. Note that the model is allowed to access all the entire information of $\mathbf{A}_a$ and $\mathbf{X}_a$ (i.e., not only $\mathbf{A}$ and $\mathbf{X}$) during the testing.

3 BACKGROUND

In this section we will briefly introduce Graph Convolutional Networks along with relevant epidemic and diffusion models.

3.1 Graph Convolutional Networks

Currently, GCN models are constructed in such a way to utilize a convolutional operator to enable the neural network weights to be shared across the graph. This provides a plethora of benefits over

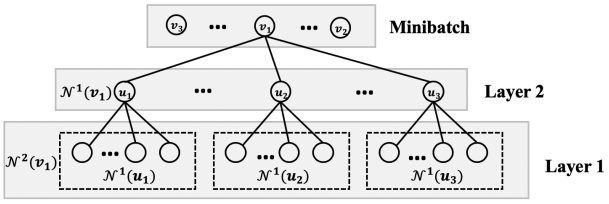


Figure 1: A GCN with two local sampling layers.

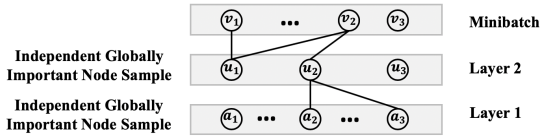


Figure 2: A GCN with two global sampling layers.

more naive structures (e.g., a fully connected network) including the ability to handle networks of different size, avoiding potential overfitting and parameter explosion, while also providing a mechanism for learning inductive node representations that can be used in a wide range of network related tasks, e.g., node classification. The original GCN model [29] performs aggregation for each node from all its direct neighbors, which becomes prohibitively expensive when the size of graph gets larger. To alleviate this issue, some have proposed to reduce the possible connections [11, 44]. However, the most common recent methods have proposed to use sampling to reduce the number of nodes to aggregate from. Among these methods, there are two main ways to perform the sampling.

In [22], a local sampling approach was first proposed. As shown in Figure 1, to perform aggregation for a node v_1 in an arbitrary layer, a fixed number of nodes are sampled from the directed neighbors $\mathcal{N}^1(v_1)$ of node v_1. Then, the aggregation process is performed over the sampled neighboring nodes instead of all node neighbors. The sampling process uses only local information, since for each node it only samples from its direct neighbors.

A global sampling process was first proposed in [9]. As shown in Figure 2, for each layer, a set of nodes is sampled according to some global importance measure. The nodes in different layers (including the mini-batch layer) are connected according to the topology in the original graph. The aggregation process is then performed only over the newly built structure. Note that the nodes are still aggregating from its directed neighbors, but only those neighbors that have been globally sampled. This could lead to an issue that in a given layer there could be some nodes without any directed neighbors sampled in the next layer. For example, in Figure 2 node u_1 does not have any sampled directed neighbors in the layer 1, which means it cannot aggregate information from other nodes.

3.2 Epidemic and Diffusion Modeling

Epidemic models are typically used for modeling processes such as the spread of an infection on the nodes of a network over their links [27, 36]. However, this can also be seen from a diffusion perspective, where the infected nodes are passing information to some of those around them and potentially have the chance of forgetting that information themselves.

The simplest epidemic model is Susceptible-Infection (SI) model [1] where susceptible nodes could possibly catch a disease from a

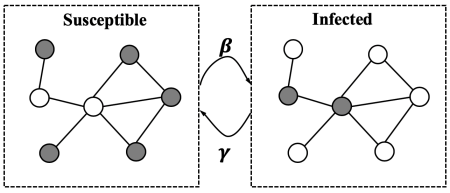


Figure 3: A visualization of the SIS model where nodes can be either susceptible or infected.

neighboring infected node. The extended Susceptible-Infection-Susceptible (SIS) model [2, 36] also considers the case that some infected nodes might recover from the disease and possibly become infected again later on. The epidemic process of the SIS model is shown in Figure 3 where the infection rate β represents the probability that infected nodes passes the disease to their neighbors who are currently susceptible nodes (per unit time), while the recovery rate γ controls the probability that an infected node recovers from the disease and moves back to the susceptible state. There are also other extensions [30] such as SIR and SIRS, but in this paper we focus on the SIS model (with details discussed in Section 4.2).

Two popular diffusion process models are the Linear Threshold Model (LTM) and Independent Cascade Model (ICM) [27]. In ICM, when a node v_i becomes active, it is given a chance to activate its inactive neighbor v_j in the next step and succeeds with a probability p_{v_i,v_j}, which is independent of the history. But if it fails, it cannot make any further attempts to activate it's neighbors. The process continues until no more activations are possible. In LTM, the process of becoming activated requires that a given percentage of a node's neighbors are also currently activated. Thus, given an initial set of active nodes and a threshold (either globally or per node) the process is similar to ICM, but instead requires nodes to only activate when enough of their neighbors are activated.

Next, in Section 4 we will first draw a connection between the two GCN aggregation sampling methods with the epidemic and diffusion models. More specifically, we will show the local sampling in GraphSAGE is similar to ICM, while the global sampling in FastGCN is similar to SIS with using the infected nodes as the globally sampled set. Then, we will provide the details of our proposed Epidemic Graph Convolutional Network.

4 EPIDEMIC GRAPH CONVOLUTIONAL NETWORK

Now, having briefly introduced Graph Convolutional Networks and the basic ideas behind the relevant epidemic and diffusion network models, we will introduce our proposed framework, Epidemic Graph Convolutional Networks (EGCN). First, we analyze the relationship between the local and global sampling methods to classical network diffusion and epidemic models. Then, given this new found relationship, we discuss the challenges facing existing GCN sampling approaches, with emphasis in terms of diffusion and epidemic modeling. We then propose the use of a low-dimensionally approximated node similarity measure to aid in alleviating these found issues. Then we discuss how to integrate our similarity measure into enhancing both the local and global sampling methods. Finally we present EGCN in its entirety that combines the local and global sampling into one principled coherent framework.

4.1 Relation Between Local Sampling and ICM

Let us denote the set of initially active nodes in the ICM model as $\mathcal{A}_0$ at time $t = 0$. Then, for the first step of the ICM model, for every node $v_i \in \mathcal{A}_0$, they have the opportunity to activate their neighbors. For example, v_i would have this one chance to activate $v_j \in \mathcal{N}^1(v_i)$ according to the probability p_{v_i,v_j}, where $\mathcal{N}^1(v_i)$ denotes the set of first order neighbors (i.e., there exists an edge between v_i and v_j). Generally, we denote the set of activated nodes at time $t+1$ as $\mathcal{A}_{t+1} = \{v_j |$ where v_j was activated by $v_i \in \mathcal{A}_t\}$. For the comparison of ICM with the local sampling in GCNs, let the set of initially activates nodes equal to the minibatch set (denoted as $\mathcal{B}$) from the GCN (i.e., $\mathcal{A}_0 = \mathcal{B}$). Then, we can start to see the similarity in the local sampling for layer $L-l+1$ of the GCN to that of $\mathcal{A}_l$ for a GCN with L layers. More specifically, if the local sampling is to pick s neighbors uniformly (such as in GraphSAGE), then we can set the probabilities of activating their neighbor $v_j \in \mathcal{N}^1(v_i)$ as $p_{v_i,v_j} = \frac{s}{|\mathcal{N}^1(v_i)|}$. Notice then the expectation is that v_i will activate (i.e., sample) s of their neighbors. Thus, a local sampling of L layers in a method such as GraphSAGE [22], is similar to selecting the activates nodes from an ICM when the edge probabilities are set as defined above and activated for L steps.

4.2 Relation Between Global Sampling and SIS

Now that we have shown the relationship between the local sampling with ICM, we now seek to show the relationship of the global sampling performed in FastGCN to that of the SIS epidemic model. First, we denote the initially infected nodes in the epidemic as $\mathcal{X}$. Furthermore, we denote the probability of node v_i being susceptible and infected at time t as $\{s_i(t)\}$ and $\{x_i(t)\}$, respectively, and then the evolution of infected status can be formulated with the following differential equation:

$$\frac{dx_i(t)}{dt} = \beta s_i(t) \sum_{j=1}^{N} \mathbf{A}_{[i,j]} x_j(t) - \gamma x_i(t) \tag{1}$$

where $\mathbf{A}$ is the adjacent matrix and $\beta s_i \mathbf{A}_{[i,j]} x_j(t)$ represents the probability of node v_i catching a disease from node v_j. Note that the infection rate β and recovery rate γ are those introduced in Section 3.2. At the early stage, $s_i(0) = 1 - c/N \approx 1$ where the number of initial infected nodes c is small while the total number of nodes N is large. Therefore, the Eq. (1) can be approximated as:

$$\frac{dx_i(t)}{dt} \approx \beta \sum_{j} \mathbf{A}_{[i,j]} x_j(t) - \gamma x_i(t) \tag{2}$$

In a vector form, it is equivalent to:

$$\frac{d\mathbf{x}(t)}{dt} = \beta(\mathbf{A} - \frac{\gamma}{\beta}\mathbf{I})\mathbf{x}(t) \tag{3}$$

Let's write $\mathbf{x}(t)$ as a linear combination of the eigenvectors of $\mathbf{A}$:

$$\mathbf{x}(t) = \sum_{r=1}^{n} a_r(t)\mathbf{v}_r \tag{4}$$

Then, substituting back into Eq. (3), we have:

$$\sum_{r=1}^{N} \frac{da_r(t)}{dt}\mathbf{v}_r = \beta \sum_{r=1}^{N} (\kappa_r - \frac{\gamma}{\beta}) a_r(t)\mathbf{v}_r \tag{5}$$

where κ_r is the r^{th} eigenvalue. Then, comparing the term in $\mathbf{v}_r$, we have:

$$\frac{da_r(t)}{dt} = \beta(\kappa_r - \frac{\gamma}{\beta}) a_r(t) \tag{6}$$

which has the solution: $a_r(t) = a_r(0) e^{\beta(\kappa_r - \frac{\gamma}{\beta})t}$. The fastest growing term in the closed-form solution of Eq. (3) corresponds to the largest eigenvalue κ_1 and its eigenvector $\mathbf{v}_1$:

$$\mathbf{x}(t) \sim a_r(0) e^{\beta(\kappa_r - \frac{\gamma}{\beta})t} \mathbf{v}_1 \tag{7}$$

which is proportional to the eigenvector centrality [4, 5] $\mathbf{v}_1$ (which is a part of a popular class of centrality measures [20]). In other words, for nodes with higher centrality, they are more likely to be infected in the early stages of the SIS epidemic. Note that if γ (i.e., the recovery rate) is too large then the exponential term becomes negative and the epidemic will die out. Thus, the epidemic threshold occurs when $\beta(\kappa_r - \frac{\gamma}{\beta}) = 0$, i.e., when $\beta/\gamma = 1/\kappa_1$. Thus, β and γ can be chosen to allow for a steady epidemic according to κ_1. Furthermore, an analysis can be performed at late stages in the epidemic when $dx_i(t)/dt = 0$ to again show the infected nodes are proportional to the eigenvector centrality (with details in [36]).

Note that in FastGCN the set of globally sampled nodes ultimately ends up being from the below node importance distribution:

$$q(i) = ||\hat{\mathbf{A}}_{[:,i]}||^2 \; / \sum_{v_j \in \mathcal{V}} ||\hat{\mathbf{A}}_{[:,j]}||^2, \; v_i \in \mathcal{V} \tag{8}$$

where $\hat{\mathbf{A}}$ is a normalized version of the adjacency matrix $\mathbf{A}$ and $\hat{\mathbf{A}}_{[:,i]}$ denotes the i^{th} column of $\hat{\mathbf{A}}$. More specifically, $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{\frac{1}{2}}$, where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{[i,j]}$ and $\mathbf{I}$ is the identity matrix. Thus, both the infected nodes and globally sampled nodes in FastGCN, are being sampled according to their global importance in the network, although somewhat differently defined.

4.3 Challenges with Local and Global Sampling

In this subsection, we present a discussion on the challenges of designing a proper sampling technique for the GCN aggregation process by discussing the inherent downsides of the existing local and global methods. These views also incorporate some ideas/perspectives when thinking in relation to the newly discovered similarity to classical diffusion and epidemic models.

First, as previously mentioned, GraphSAGE [22] is a representative local sampling GCN model (as seen in Figure 1). In this method, a minibatch of nodes is first selected and then nodes are selected to aggregate from using their local neighborhood. In each layer of the GCN each of the nodes uniformly samples a set of s nodes. The first noticeable problem is that typically these models require the sampling of a large set of nodes (e.g., GraphSAGE recommends 25 nodes in the first layer and 10 in the second layer [22]). This results in an exponential expansion of the nodes that are required to aggregate from in terms of s (which is not necessarily the same in all layers). Another problem arises in the fact that this style of sampling uniformly at random is susceptible to both noisy connections and any potential adversarial links (which recently has become a popular topic of interest in the GCN domain [3, 12, 41, 48, 49]. In comparison with ICM, it is more likely that nodes are becoming activated in real network diffusion cascades from more similar neighbors as compared to uniformly at random. Also, in ICM, noisy links (i.e., two connected nodes) that do not interact much will result

in a probability between them being very low (even approaching zero). Thus, the local sampling methods are still open to potential improvements, specifically desiring a principled distribution to sample nodes from their local neighborhood. This could potentially allow the avoidance the challenges of using a local sampling such as, avoiding noisy connections, reduce the number of needed samples (i.e., smaller s), having a more informed aggregation beyond assuming all neighbors are equal (i.e., inferred weighted edges).

In comparison, for the global sampling, the problems/challenges are quite different. As mentioned in Section 3, a representative global sampling GCN model is FastGCN [9]. The premise of this model is to avoid the exponential expansion that happens within the local sampling method. More specifically, the global sampling method attempts to independently perform a global sampling for each layer of the GCN. However, one potential drawback of this methodology of sampling is that within larger networks problems could arise, since inherently many nodes in a minibatch might not connect to a given selected subset of globally important nodes. Furthermore, there could exist many real-world graphs that by construction have many nodes not connected to globally important nodes, and thus resulting in a biased training process. In other words, some nodes in minibatches (or those in other sampled layers) might not be able to find a node to aggregate from. Similar to in the local sampling case, it is desired to not only guarantee possible nodes to aggregate from, but furthermore to have a principled weighting according to the nodes local neighborhood and not only a global measure. Hence, to partially alleviate some of the problems discussed in both the local and global sampling techniques, we propose the use of a structural similarity measure during the sampling/aggregation process. We also note that for some graphs, such as social networks, if given information cascades (e.g., retweets pathways on Twitter), we could use that information in addition to structural node similarity, but we leave this as one future work.

4.4 Node Similarity to Alleviate Challenges

As previously mentioned, here we will introduce the usefulness of harnessing a similarity measure to improve the local and global sampling/aggregation methods.

For the local aggregation, we propose the use of this similarly to first perform a *top-s* sub-sampling of each node's neighborhood. In other words, for each node $v \in \mathcal{V}$, we use the similarity to select the at most s_{top} neighbors from $\mathcal{N}^1(v)$ to form the set $\mathcal{N}_s^1(v)$. This helps to alleviate noisy links and also allows us to lower the number of needed samples if we are focusing on aggregating from the most similar neighbors (in terms of network structure). Furthermore, the similarity (which we will later define) will be used to perform a weighted averaging over the set $\mathcal{N}_s^1(v)$ for a node v, again providing more informative and principled information.

For the global sampling, we will utilize the similarity to resolve the issue of having nodes in the minibatch either not having many connections to the globally important set of nodes, or lacking many connections. In other words, we can utilize the similarity to perform a full aggregation over all the sampled global nodes according to a weighted average based on the nodes similarity to each of the sampled nodes (even if a direct link between them does not exist). Furthermore, inherently a principled weighted average as compared to a uniform aggregation is preferred, which is provided through the use of the node similarity measure.

In choosing the node similarity measure, there are many such methods [32]. However, we have two requirements: 1) it needs to be a global similarity measure, such that in the global sampling we can still have a non-zero similarity for many nodes, and 2) it also needs to be both computationally feasible in both terms of time and space. Thus, we propose the use of a low-dimensional approximation to the Katz similarity measure[26], which has shown to perform well for the link prediction task [33]. The standard Katz similarity measure can be defined as follows:

$$\mathbf{S} = \sum_{k=1}^{\infty} \alpha^k \mathbf{A}^k + \delta \tag{9}$$

where α is used to exponentially decay the importance of paths of length k and δ can be used to provide a non-zero similarity to all nodes. However, the close form solution to this is intractable for larger graphs in both computational time and space (as we would need to store the entire dense $O(N^2)$ matrix). Thus, we can calculate a low-dimensional approximation of this similarity measurement instead. First, we note that in our setting of an undirected graph, the adjacency matrix $\mathbf{A}$ is symmetric and can be diagonalized as $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, where $\mathbf{U} \in \mathbb{R}^{N \times N}$ and $\mathbf{\Lambda} \in \mathbb{R}^{N \times N}$ are the eigenvector and diagonalized eigenvalues, respectively. However, rather than the full eigen-decomposition, we can instead perform it being truncated to only the largest c eigen-pairs. Thus, we then have the following:

$$\mathbf{A} \approx \bar{\mathbf{U}}\bar{\mathbf{\Lambda}}\bar{\mathbf{U}}^\top = \bar{\mathbf{A}} \tag{10}$$

where $\bar{\mathbf{A}}$ is the low-dimensional approximation of the adjacency matrix $\mathbf{A}$, $\bar{\mathbf{U}} \in \mathbb{R}^{N \times c}$ and $\bar{\mathbf{\Lambda}} \in \mathbb{R}^{c \times c}$. If we furthermore limit the similarity to only considering paths of up to length K we can obtain the low-dimensional approximated similarity measure between v_i and v_j as follows:

$$\bar{\mathbf{S}}_{[i,j]} = \sum_{k=1}^{K} \alpha^k \bar{\mathbf{U}}_i \bar{\mathbf{\Lambda}}^k \bar{\mathbf{U}}_j^\top + \delta \tag{11}$$

due to the fact that $\bar{\mathbf{A}}^k = (\bar{\mathbf{U}}\bar{\mathbf{\Lambda}}\bar{\mathbf{U}}^\top)^k = \bar{\mathbf{U}}\bar{\mathbf{\Lambda}}^k\bar{\mathbf{U}}^\top$ and thus providing a more efficient computational calculation for node similarity. Note that for our similarity calculations we instead use the normalized adjacency matrix $\hat{\mathbf{A}}$ as described in Section 4.2 instead of $\mathbf{A}$.

Furthermore, we note that in this way we are able to avoid the computation of the entire similarity matrix. Due to the fact we will use a fixed set of most similar nodes to aggregate from instead of performing a random local sampling for each minibatch, we can precompute these to find the *top-s* neighbors for each node v_i. In more detail, we compute the similarities $\bar{\mathbf{S}}_{[i,j]}\ \forall v_j \in \mathcal{N}^1(v_i)$ and select out the *top-s* most similar neighbors to form the sub-sampled neighborhood $\mathcal{N}_s^1(v_i)$. Note that for computational efficiency we restrict the search for the *top-s* most similar nodes to be among the first order neighborhood of v_i (i.e., in the set $\mathcal{N}^1(v_i)$). However, for the global sampling, we need to compute the similarity from each node in our minibatch to the nodes that are globally sampled, since they will be dynamically changing during the training (unlike the local fixed sampled set). Note that a cache could be used to further improve the efficiency by not having to recompute similarity scores by computing the similarity from each node to a small number of

globally important nodes (as they are the ones more likely to be sampled). Next we will discuss in detail how we perform the local and global sampling/aggregation stages of our EGCN framework.

4.5 The Proposed Local Aggregation Details

We describe how to perform the local sampling and aggregation process in this subsection. We adapt the sampling and aggregation operations in the GraphSAGE model to form the local sampling and aggregation components of our model. However, instead of uniformly sampling from the first order neighbors of a node v_i, a fixed number of most similar nodes (i.e., the set $\mathcal{N}_s^1(v_i)$) are selected according to the similarity measure. To aggregate the information from the sampled nodes $\mathcal{N}_s^1(v_i)$ for node v_i, we use the weighted average of representations of the sampled nodes as the new representation of the node. Mathematically, generating the representation for node v_i in the l-th layer of proposed network can be formulated as:

$$\mathbf{h}_i^l = \sigma\left(\left[\mathbf{h}_i^{l-1}, \frac{1}{\sum\limits_{v_j \in \mathcal{N}_s^1(v_i)} \bar{\mathbf{S}}_{[i,j]}} \sum_{v_j \in \mathcal{N}_s^1(v_i)} \bar{\mathbf{S}}_{[i,j]} \mathbf{h}_j^{l-1}\right] \mathbf{W}^l\right) \tag{12}$$

where $\mathbf{h}_i^l$ denotes the obtained d_l-dimensional representations for node v_i after the l-th layer, $[\cdot,\cdot]$ denotes concatenation, $\mathbf{W}^l \in \mathbb{R}^{2d_{l-1} \times d_l}$ is the weight matrix of the l-th local sampling and aggregation layer, and σ is a nonlinear activation function. Note that $\mathbf{h}_i^0$ is the associated feature vector of node v_i, which means that $\mathbf{h}_i^0 = \mathbf{x}_i$. A total of L layers of sampling and aggregation operations can be stacked. The output representation of the local sampling and aggregation component for node v_i is $\mathbf{h}_i^L$, which, for convenience, also denoted as the e-dimensional (with $e = d_L$) representation $\mathbf{z}_i^{ls}$.

4.6 The Proposed Global Aggregation Details

As previously mentioned, we propose to utilize an SIS epidemic model to perform the global sampling in EGCN. To do this, we initially select a random sample of nodes to e "infect" (denoted as $\mathcal{X}_0$) and then throughout the training, ever E updates (i.e., minibatches), we perform a step of the SIS model. This step will allow the infected nodes to infect each of their neighbors independently with probability β and then with probability γ change themselves to being susceptible again (instead of infected). As shown in Section 4.2, we can select hyperparameters for the SIS model (i.e., β and γ) to obtain a relatively stable infection (according to κ_1) where the small number of nodes infected at any given step will be more likely those having a high importance. Thus, for a given step t during the epidemic, we let the set of infected nodes at that time be denoted by $\mathcal{X}_t$.

Given we have a set of infected nodes $\mathcal{X}_t$ at time t, we can obtain the global sampling based e-dimensional representation $\mathbf{z}_i^{gs}$ from the global sample for each node v_i in minibatch $\mathcal{B}$ as follows:

$$\mathbf{z}_i^{gs} = \sigma\left(\left[\mathbf{x}_i\ ,\ \frac{1}{\sum\limits_{v_j \in \mathcal{X}_t} \bar{\mathbf{S}}_{[i,j]}} \sum_{v_j \in \mathcal{X}_t} \bar{\mathbf{S}}_{[i,j]} \mathbf{x}_j\right] \mathbf{W}^g\right) \tag{13}$$

where we use $[\cdot,\cdot]$ to denote concatenation, $\mathbf{W}^g \in \mathbb{R}^{2d \times e}$ is the weights of the global sampling single layer GCN, and σ is a nonlinear activation function.

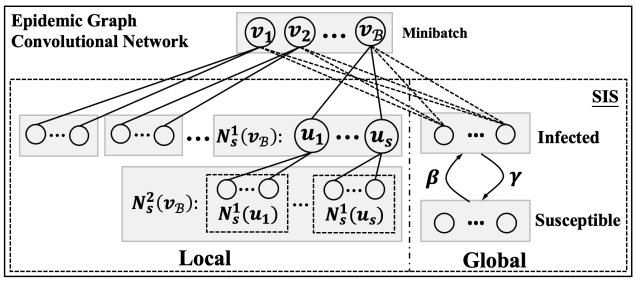


Figure 4: A visualization of the proposed EGCN Framework.

4.7 Overview of the EGCN Framework

Thus far we have shown how to obtain both the local and global sampling based representations $\mathbf{z}^{ls}$ and $\mathbf{z}^{gs}$, respectively. These can be seen in Figure 4 where on the left we have the local sampling and the right shows our global sampling according to the SIS model.

Now, we combine them into a unified representation that harnesses the power from both our enhanced local and global sampling/aggregation methods into a single representation $\mathbf{z}_i$ for a node v_i that can be trained and used for tasks such as node classification. More specifically, we denote the unified representation as follows:

$$\mathbf{z}_i = \pi \mathbf{z}_i^{ls} + (1-\pi)\mathbf{z}_i^{gs} \tag{14}$$

where here we linearly combine the local and global representations through the tunable hyperparameter π (which controls the balance between favoring the local/global sampling). Note that this choice also importantly is used to allow for a better understanding of the contribution of these two methods in EGCN (which is discussed in Section 5.5).

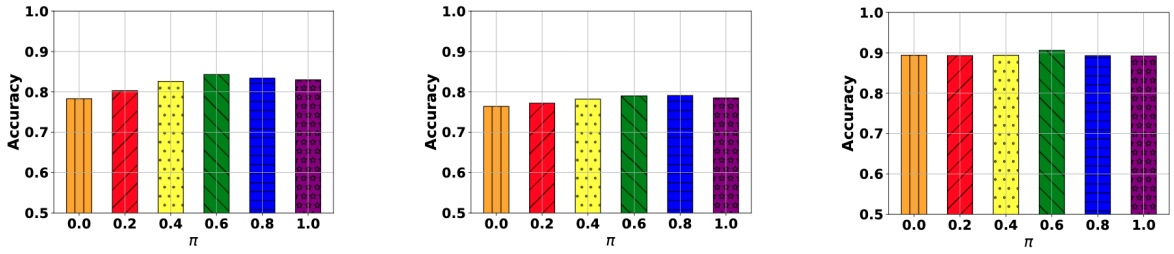
5 EXPERIMENTS

In this section, we perform an empirical evaluation to experimentally show the effectiveness of the proposed Epidemic Graph Convolutional Network (EGCN) towards learning inductive node representations. Through these experiments we seek to answer the following questions: (1) Can EGCN learn meaningful inductive node representations? and (2) Does using both locally and globally important sampled nodes in the node aggregation allow for learning better node representations (i.e., improved performance)?

We conduct experiments to answer the first question by performing the inductive node classification task by learning node representations using EGCN and comparing against baseline methods including state-of-the-art for this given task. Furthermore, we perform a parameter analysis to investigate the contribution of different components of the proposed Epidemic Graph Convolutional Network to answer the second question and better understand the model. These experiments are performed on a set of commonly used real-world graph datasets for the node classification task, where the size of the graph varies and also the number of node classes, as these two factors can greatly impact the performance.

5.1 Experimental Settings

Here we first introduce the datasets, then discuss the settings used in EGCN, followed by a description of the baseline methods we have compared against, and finally discussing the details for the metrics used for evaluation on the node classification task.



(a) Cora (b) Citeseer (c) Pubmed

Figure 5: Effect of varying the balance between local/global in EGCN Cora/Citeseer/Pubmed datasets with π along the x-axis.

Table 1: Basic statistics of the three real world datasets.

Network	# Nodes	Edges	Classes	Features	Training/Validation/ Testing
Cora	2,708	5,429	7	1,433	1,208/500/1,000
Citeseer	3,327	4,372	6	3,703	1,812/500/1,000
Pubmed	19,717	44,338	3	500	18,217/500/1,000

Table 2: Results on accuracy for node classification.

Embedding Method	Cora	Citeseer	Pubmed
NodeFeats	0.7380	0.7530	0.8750
GraphSAGE	0.8220	0.7140	0.8710
FastGCN	0.8500	0.7760	0.8800
EpidemicGCN	0.8430	0.7910	0.9060

5.2 Datasets

We evaluate our methods on several widely used benchmarks for node classification, which includes Citeseer, Pubmed and Cora [40]. In these datasets, nodes are documents and edges are the citation links between documents. Each of the document can be represented as a sparse bag-of-words feature vector, which can be viewed as the features associated to the nodes. Some of the basic statistics and the splitting of the dataset can be found in Table 1.

5.3 EGCN Hyperparameter Settings

For EGCN[1], we set the hyperparameters as follows. First, for the low-dimensional approximated similarity, we set the low rank dimension $c = 128$, $\alpha = 0.85$ that controls the contribution of longer order paths towards the similarity, δ =1e-05 to allow non-negative similarities, and the maximum approximated path-length to be $K = 3$. Then for the local sampling, we fix the $top\text{-}s = 5$ where we are aggregating form a fixed up to 5 neighbors for every node. We set the parameter $\gamma = 1$ in the global sampling SIS model (i.e., the recovery rate is 100%) to help promote diversity in the sampled (i.e., infected) sets, while β is chosen to maintain roughly 200 infected nodes (i.e., $|\mathcal{X}_t| \approx 200\ \forall t$). In the SIS model, we set the parameter E for the number of minibatches before updating to the next time step in SIS to be $E = 50$, to balance between obtaining new global sampled sets, the efficiency of running less epidemic time steps, and stability during training using the same sampled batch for a set of minibatches in a row. For Cora and Citeseer we use an initial set of infected random nodes $|\mathcal{X}_0| = 256$ and for Pubmed we use $|\mathcal{X}_0| = 512$. For the local, global, and final unified representations we use an embedding size of $e = 128$ for EGCN across all three datasets. For training we use Adam [28] to update the parameters in minibatch fashion with $|\mathcal{B}| = 256$ for Cora and Citeseer while using $|\mathcal{B}| = 1024$ for Pubmed. For the nonlinearity, we use σ to be ReLu [35]. The parameters are set using a grid search on for tuning according to the validation set, we have λ which is the regularization penalty on the L2-norm across all the learned weights of EGCN, learning rate η, and π which controls the contribution of the local versus global sampling/aggregation. During the tuning we vary these parameters as follows: $\lambda \in \{1e-02, 1e-03, 5e-04, 1e-04, 5e-05, 1e-05\}$, $\eta \in \{0.1, 0.05, 0.01, 0.005\}$ and $\pi \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$.

[1]https://github.com/DSE-MSU/EpidemicGCN

5.4 Performance Comparison

We compare the proposed EGCN with two samplings based GCN methods, GraphSAGE and FastGCN. To demonstrate the effectiveness of the network information, we also include a method denoted as NodeFeats, which only utilize the node features while ignoring the connections between nodes (i.e., NodeFeats directly uses the node features instead of using a GCN to learn representations). Note that we use the suggested parameters for both GraphSAGE and FastGCN as suggested by the authors in their respective works for these datasets. The results of node classification can be found in Table 2 where we observe that our method achieves comparable or even better performance than the baselines. EGCN especially outperforms the baselines with a large margin in the Citeseer and Pubmed datasets. Thus it appears that overall EGCN is able to learn meaningful inductive node representations; thus answering our first experimental question.

Based on our findings, we can observe that our local sampling neighborhood is smaller than that of GraphSAGE, but yet we are still able to outperform, which implies that the use of a fixed $top-k$ neighbbors according to similarities (while also performing a weighted average) helps EGCN to improve performance. Similarly, when compared against FastGCN, on the larger dataset, Pubmed, we observe that this is where we find the largest improvement, which is likely due to the global sampling challenges we had discussed previously when applying to the larger networks.

5.5 Parameter Analysis

The most important parameter to analyze here is π as that gives insight into the role the enhanced local and global sampling/aggregation mechanisms are to the combined power of EGCN. We first note that

the best performance among the tested π values was $\pi = 0.6$ for Cora and Pubmed and $\pi = 0.8$ for Citeseer. In Figure 5 we show the performance across the entire set of π values for our three datasets. Note that we kept all other parameters fixed when varying π for this analysis according to the best found for that given dataset during the grid search. Furthermore, we point out that using $\pi = 0$ is only considering global sampling with SIS and $\pi = 1.0$ is effectively only local sampling. Therefore, based upon both Figure 5 we can indeed see that both the local and global sampling/aggregation are meaningful and that together they achieve the best performance; thus answering our second proposed experimental question. However, the performance was quite stable for the Pubmed dataset.

As common with GCN models, we noticed that the L2-norm and learning rate influenced the performance, and the best settings found were the following. Learning rate on Cora and Citeseer was $\eta = 0.05$, while for Pubmed we found the best to be $\eta = 0.01$. As for the L2-norm, for Cora and Citeseer we observed that $\lambda = 5e-04$ performed the optimal in our examined range, and $\lambda = 1e-05$ for Pubmed.

6 RELATED WORK

In this section, we briefly review related work to GCNs and epidemic/diffusion models.

6.1 Graph Convolutional Neural Networks

Graphs are an essential representation for many data in various areas such as social networks, academic citation networks and transportation networks. Recently, many methods have been made to generalize deep neural networks to graph structured data. Among them, graph convolutional networks (GCNs) [13, 29], which adapted the classical convolution operation in classical Euclidean data to graph structured data, have gained extreme popularity. The key operation for GCNs is to aggregate features for each node from all its direct neighbors. This operation is natural and flexible regardless of the structure and size of the graphs. However, it could become prohibitively expensive when the size of graph gets large. To alleviate this issue, recent efforts have been made to reduce the number of nodes to aggregate from for each convolution operator. Other models such as PinSage [44] and Cluster-GCN [11] have recently been proposed that reduce the aggregation through limiting neighbors by either utilizing random walks similar to [18] (which in the limit approximates the Personalized PageRank [21, 24, 38]) or utilizes clustering boundaries, respectively. From the global sampling perspective, in [23] they had improved upon the sampling method of FastGCN [9] by adaptively sampling to ensure the better connectedness between aggregation layers. In [45], P-GNN was introduced that creates anchor-sets of nodes to be the commonly used aggregation sets for all nodes according to a similarity measure (instead of local neighborhoods); where our globally sampled sets of nodes can be compared to their anchor-sets.

6.2 Epidemic and Diffusion Modeling

The epidemic and diffusion models are used for modeling the spreading or diffusion through the networks across the connections between nodes. Some classical epidemic models [1, 2, 30, 36] includes Susceptible-Infection (SI) model, Susceptible-Infection-Susceptible (SIS) model, Susceptible-Infection-Recovery (SIR) model. Linear threshold model (LTM) and independent cascade model (ICM) [27] are two most popular diffusion models. As explained in Section 4, a local sampling of l layers in a method like GraphSAGE [22], is similar to selecting the activates nodes from an ICM while the global sampling methods in FastGCN [9] is similar to selecting infected nodes from a SIS model.

7 CONCLUSION

Recently, many works have witnessed the power of graph convolutional networks (GCNs) in lots of applications, but the lack of scalability hinders the potential applications for large-scale networks. This has led recent GCNs towards sampling techniques to still be applicable in training on large networks. However, the two primary sampling techniques, i.e., local and global, both have inherent shortcomings. Thus, in this paper, we first observe and formulate similarities between these two sampling methods and classical epidemic and diffusion models. Then, given this insight, we formulate our proposed Epidemic Graph Convolutional Network (EGCN) framework harnessing a low-dimensional approximation of the path-based Katz similarity.

On the one hand, from the local sampling perspective, EGCN now provides a principled mechanism for the local sampling according to the top-s most similar nodes based on this similarity. This also allows to reduce the exponential expansion in the local sampling by selecting a smaller s due to the improvement found in using the top-s neighbors as compared to a randomly selected s. On the other hand, from the global sampling perspective, EGCN is built upon the SIS epidemic process, which has been shown to infect/sample nodes proportional to their eigenvector centrality values, and also paired with the similarity to alleviate the issues of not having direct connections to the globally sampled nodes. Furthermore, in both sides of the sampling the efficient similarity allows for a weighted aggregation based on the approximated Katz structural similarity. Thus, the proposed EGCN overcomes the drawbacks of the previously proposed methods and experiments on three real-world datasets demonstrate its effectiveness compared with state-of-art baselines.

Our future work will first be to extend similar techniques to GCNs of more complex types of graphs, such as signed networks [14], where we would have to incorporate signed node similarities measures [15]. Thereafter, we will work towards the development of defensive robust GCN models [47, 50] that are able to withstand state-of-the-art GCN attack methodologies [34, 48, 49]; since recently adversarial attacks and defenses across all domains have become a critical area for investigation [43]. We believe the use of low-dimensionality will be beneficial and also the incorporation of similarity to avoid the noisy or any potential adversarially added links when constructing more robust GCN models.

ACKNOWLEDGMENTS

Tyler Derr, Yao Ma, Xiaorui Liu, and Jiliang Tang are supported by the National Science Foundation (NSF) under grant numbers IIS-1714741, IIS-1715940, IIS-1845081 and CNS-1815636, and a grant from Criteo Faculty Research Award.

REFERENCES

[1] Wen-Jie Bai, Tao Zhou, and Bing-Hong Wang. 2007. Immunization of susceptible–infected model on scale-free networks. *Physica A: Statistical Mechanics and its Applications* 384, 2 (2007), 656–662.

[2] Marian Boguná, Claudio Castellano, and Romualdo Pastor-Satorras. 2013. Nature of the epidemic threshold for the susceptible-infected-susceptible dynamics in networks. *Physical review letters* 111, 6 (2013), 068701.

[3] Aleksandar Bojcheski and Stephan Günnemann. 2018. Adversarial attacks on node embeddings. *arXiv preprint arXiv:1809.01093* (2018).

[4] Phillip Bonacich. 1972. Factoring and weighting approaches to status scores and clique identification. *Journal of mathematical sociology* 2, 1 (1972), 113–120.

[5] Phillip Bonacich. 2007. Some unique properties of eigenvector centrality. *Social networks* 29, 4 (2007), 555–564.

[6] Joan Bruna and X Li. 2017. Community detection with graph neural networks. *stat* 1050 (2017), 27.

[7] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203* (2013).

[8] Sandro Cavallari, Vincent W Zheng, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. 2017. Learning community embedding with community detection and node embedding on graphs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 377–386.

[9] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling. In *International Conference on Learning Representations*.

[10] Zhengdao Chen, Lisha Li, and Joan Bruna. 2019. Supervised Community Detection with Line Graph Neural Networks. In *International Conference on Learning Representations*.

[11] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks. *arXiv preprint arXiv:1905.07953* (2019).

[12] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. 2018. Adversarial Attack on Graph Structured Data. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Jennifer Dy and Andreas Krause (Eds.), Vol. 80. PMLR, Stockholmsmässan, Stockholm Sweden, 1115–1124. http://proceedings.mlr.press/v80/dai18b.html

[13] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*. 3844–3852.

[14] Tyler Derr, Yao Ma, and Jiliang Tang. 2018. Signed graph convolutional networks. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 929–934.

[15] Tyler Derr, Chenxing Wang, Suhang Wang, and Jiliang Tang. 2018. Relevance Measurements in Online Signed Social Networks. In *Proceedings of the 14th International Workshop on Mining and Learning with Graphs (MLG)*.

[16] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[17] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*. 2224–2232.

[18] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 1775–1784.

[19] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *The World Wide Web Conference*. ACM, 417–426.

[20] Noah E Friedkin. 1991. Theoretical foundations for centrality measures. *American journal of Sociology* 96, 6 (1991), 1478–1504.

[21] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. 2004. Combating web spam with trustrank. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 576–587.

[22] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[23] Wenbing Huang, Tong Zhang, Yu Rong, and Junzhou Huang. 2018. Adaptive sampling towards fast graph representation learning. In *Advances in Neural Information Processing Systems*. 4558–4567.

[24] Glen Jeh and Jennifer Widom. 2003. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web*. Acm, 271–279.

[25] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (01 Mar 1953), 39–43. https://doi.org/10.1007/BF02289026

[26] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (1953), 39–43.

[27] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence Through a Social Network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*. ACM, New York, NY, USA, 137–146. https://doi.org/10.1145/956750.956769

[28] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[29] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[30] Andrei Korobeinikov and Graeme C Wake. 2002. Lyapunov functions and global stability for SIR, SIRS, and SIS epidemiological models. *Applied Mathematics Letters* 15, 8 (2002), 955–960.

[31] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[32] Elizabeth A Leicht, Petter Holme, and Mark EJ Newman. 2006. Vertex similarity in networks. *Physical Review E* 73, 2 (2006), 026120.

[33] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58, 7 (2007), 1019–1031.

[34] Yao Ma, Suhang Wang, Tyler Derr, Lingfei Wu, and Jiliang Tang. 2019. Attacking Graph Convolutional Networks via Rewiring. *arXiv preprint arXiv:1906.03750* (2019).

[35] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *DLASLP at ICML*.

[36] Mark Newman. 2010. *Networks: An Introduction*. Oxford University Press, Inc., New York, NY, USA.

[37] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning convolutional neural networks for graphs. In *International conference on machine learning*. 2014–2023.

[38] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.

[39] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*. Springer, 593–607.

[40] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.

[41] Mingjie Sun, Jian Tang, Huichen Li, Bo Li, Chaowei Xiao, Yao Chen, and Dawn Song. 2018. Data Poisoning Attack against Unsupervised Node Embedding Methods. *CoRR* abs/1810.12881 (2018). arXiv:1810.12881 http://arxiv.org/abs/1810.12881

[42] Le Wu, Peijie Sun, Richang Hong, Yanjie Fu, Xiting Wang, and Meng Wang. 2018. SocialGCN: An Efficient Graph Convolutional Network based Model for Social Recommendation. *arXiv preprint arXiv:1811.02815* (2018).

[43] Han Xu, Yao Ma, Haochen Liu, Debayan Deb, Hui Liu, Jiliang Tang, and Anil Jain. 2019. Adversarial attacks and defenses in images, graphs and text: A review. *arXiv preprint arXiv:1909.08072* (2019).

[44] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 974–983.

[45] Jiaxuan You, Rex Ying, and Jure Leskovec. 2019. Position-aware Graph Neural Networks. In *International Conference on Machine Learning*. 7134–7143.

[46] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*. 5165–5175.

[47] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2019. Robust Graph Convolutional Networks Against Adversarial Attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. ACM, 1399–1407.

[48] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2847–2856.

[49] Daniel Zügner and Stephan Günnemann. 2019. Adversarial attacks on graph neural networks via meta learning. *arXiv preprint arXiv:1902.08412* (2019).

[50] Daniel Zügner and Stephan Günnemann. 2019. Certifiable Robustness and Robust Training for Graph Convolutional Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 246–256.