Computer Standards & Interfaces 83 (2023) 103655

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Intrinsic graph topological correlation for graph convolutional network propagation

Mustafa Coskun

Hakkari University, Hakkari, Turkey and Abdullah Gul University, Kayseri, Turkey, Kayseri, 38080, Kayseri, Turkey

ARTICLE INFO

Keywords:
Graph convolution network
Dimensionality reduction
Deep graph learning

ABSTRACT

Recently, Graph Convolutional Networks (GCNs) and their variants become popular to learn graph-related tasks. These tasks include link prediction, node classification, and node embedding, among many others. In the node classification problem, the input is a graph with some labeled nodes and the features associated with these nodes and the objective is to predict the unlabeled nodes. While the GCNs have been successfully applied to this problem, some caveats that are inherited from classical deep learning remain unsolved. One such inherited caveat is that, during classification, GCNs only consider the nodes that are a few neighbors away from the labeled nodes. However, considering only a few steps away nodes could not effectively exploit the underlying graph topological information. To remedy this problem, the state-of-the-art methods leverage the network diffusion approaches, such as personalized PageRank and its variants, to fully account for the graph topology. However, these approaches overlook the fact that the network diffusion methods favour high degree nodes in the graph, resulting in the propagation of the labels to the unlabeled,hub nodes. In order to overcome bias, in this paper, we propose to utilize a dimensionality reduction technique, which is conjugate with personalized PageRank. Testing on four real-world networks that are commonly used in benchmarking GCNs' performance for the node classification task, we systematically evaluate the performance of the proposed methodology and show that our approach outperforms existing methods for wide ranges of parameter values. Since our method requires only a few training *epochs*, it releases the heavy training burden of GCNs. The source code of the proposed method is freely available at https://github.com/mustafaCoskunAgu/ScNP/blob/master/TRJMain.m.

1. Introduction

Graph Convolutional Networks (GCNs) [1] are the variants of traditional Convolutional Neural Netwoks (CNNs) applied on graphs [2]. In general terms, GCNs utilize layers of learned filters, followed by a nonlinear activation function [2] to learn the graph representations. In recent years, GCNs have been successfully applied to a wide range of problems in data mining, such as node classification [1], recommendation systems [3], the prediction of combined side effects of drugs (polypharmacy side effects) [4], link prediction in biological networks [5] and natural language processing [6].

In the node classification problem, as an input the GCNs take a graph, which represents the relationship among vertices/nodes via edges connecting them. Also, the graph contains feature vectors associated with the nodes. In this problem, some of the nodes' labels are known beforehand. The objective of the GCNs is to predict the rest of the nodes' labels on the graph by using the features of the nodes and the graph topology [7]. In order to achieve this node classification goal, at each layer of GCNs, the convolution is performed by applying a first-order spectral filter to the feature matrix, followed by a nonlinear activation function [2]. This way, the features are smoothed across the graph at each layer of the neural network by using the graph's connectivity. This smoothing process is also known as the message passing phase of a GCN [1].

Despite the successful application of GCNs to a plethora of problems, one subtle issue remains unresolved; i.e, the message passing scheme of GCNs only utilizes a few hop neighborhoods of the labeled nodes [8]. There has been a few recent attempts to address this limited message passing capacity of GCNs by using attention mechanisms [9], random walk [10], and edge features [11]. However, all of these methods can only utilize the graph topology up to a very few neighbors for each node [8] since using many convolution layers in the message passing scheme of GCNs could potentially be detrimental to the original classification task and mix the predicted labels via over-smoothing of the features [7]. This over-smoothing is the primary reason to use two layered GCNs instead of many layered GCNs, which is a more intuitive approach than

E-mail address: mustafa.coskun@agu.edu.tr.

https://doi.org/10.1016/j.csi.2022.103655
Received 30 November 2021; Received in revised form 19 February 2022; Accepted 18 April 2022
Available online 19 April 2022

using two layers GCNs, to propagate the labels toward the far away nodes in the graph [1,7].

To circumvent the limited message passing problem of GCNs, Klicpera et al. [8] first observe the connection between the feature propagation of GCNs and well-known PageRank algorithm [12]. Then, they propose an algorithmic framework that utilizes the *personalized* version of PageRank to segregate the propagation scheme from neural networks [8]. While this approach has been shown to be effective in rendering the application of GCNs to instances with many layers of neural networks without over smoothing problem, it ignores an important problem that is associated with the application of the *personalized* PageRank based techniques. The problem in this type of proximity assessment is that the random walker in the *personalized* PageRank based assessment of network proximity assigns higher scores to the nodes with high connectivity and/or centrality [13,14], which biases highly connected nodes.

Inspired by our earlier work in the context of link prediction problem [13], here we propose an algorithmic framework that fairly assesses the similarity of the nodes in a graph. More specifically, our approach is based on the idea that nodes should be compared with their "vector-wise similarity" instead of their "entry-wise similarity" as used in the GCN literature [8]. In other words, as opposed to directly assessing the proximity of two nodes by relying on entries in the columns of full *personalized* PageRank matrix, we measure their similarity in terms of what they are close to. As we have shown previously in the context of link prediction [13], this approach drastically reduces degree bias in measuring closeness of the nodes in *personalized* PageRank matrix. While assessing the proximity based on aforementioned idea, sparseness of the underlying network might adversely affect the performance of this approach due to high dimensionality problem [13]. Hence, in this paper, we first reduce the high-dimensionality in the columns of the *personalized* PageRank matrix by just discarding the entries which are smaller than a given parameter, ε. Next, we measure the closeness of two nodes and account for the correlation of those reduced profile vectors. Finally, we assess the reduced topological profile similarity matrix to measure the closeness of nodes in the graph.

To test the performance of the proposed methodology, in improving the accuracy of GCN-based node classification task and reducing the train time via less training *epochs*,which refers to one cycle through the full training dataset, we perform systematical experiments on three citation and one co-author networks. Throughout these experiments, we observe that our proposed methodology improves the accuracy of the node classification, and reduces the train time on the training datasets. Our results show that the proposed approach using reduced topological profile similarity renders GCNs highly effective in node classification, and the resulting algorithmic framework outperforms the algorithm that uses *personalized* PageRank [8] by a large margin.

The rest of the paper is organized as follows: in Section 2, we describe the terminology, establish the background on GCNs and *personalized* PageRank [8], and describe our approach; in Section 4, we provide detailed experimental evaluation of our proposed approach and we draw conclusions and summarize avenues for the future research in Section 5.

2. Background

In this section, we first define graph convolutional networks (GCNs) in the context of node classification problem. We then present insights for the usage of *personalized* PageRank to separate the message passing scheme of GCNs from neural networks.

2.1. Graph convolutional networks

Here, we follow the notations by Kipf and Welling [1] to define GCNs in the context of the node classification problem. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph, where $\mathcal{V}$ denotes the set of n nodes and $\mathcal{E}$ represents the set of m edges in the graph. The nodes are associated with a feature matrix $\mathbf{X} = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{n \times f}$ such that $x_i \in \mathbb{R}^f$ is a feature vector for node $v_i \in \mathcal{V}$ and the label matrix is given as $\mathbf{Y} \in \mathbb{R}^{n \times c}$, where c denotes the number of classes. The adjacency matrix of $\mathcal{G}$ is given by $\mathbf{A} \in \mathbb{R}^{n \times n}$, where $\mathbf{A}$'s entries are the weights of the edges. Let $\widetilde{\mathbf{A}} : \mathbf{A} + \mathbf{I}_n$ be self-loop added adjacency matrix of $\mathcal{G}$. Then, we can give node classification problem as follows:

The node classification problem: We are given an undirected graph, the features of the nodes in this graph, and a set of labeled nodes, the node classification aims to predict the rest of the labels for all nodes in the graph.One simple yet very effective message passing algorithm that is used for the aforementioned problem is the GCN. Now, we can give the two message passing layers of GCNs as follows [1]

$$\mathbf{Z}_GCN = softmax\big(\widehat{\widetilde{\mathbf{A}}} ReLU\big(\widehat{\widetilde{\mathbf{A}}}\mathbf{X}\Theta^{(0)}\big)\Theta^{(1)}\big), \tag{1}$$

where $\mathbf{Z} \in \mathbb{R}^{n \times c}$ are the predicted labels, $\widehat{\widetilde{\mathbf{A}}} = \widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}$ is symmetrically normalized adjacency matrix of $\widetilde{A}$, $\widetilde{\mathbf{D}}$ is diagonal matrix with degree of $\widetilde{A}$, $\Theta^{(0)}$), and $\Theta^{(1)}$ are trainable weight matrices [1].

For the node classification problem, we employ Eq. (1) to predict the labels of all nodes that are not labeled by only considering two-hop away nodes. In essence, there are mainly two reasons why we cannot use a larger propagation steps in GCNs: first, using many layers are equivalent to Laplacian smoothing [7] and second increasing the depth of neural networks complicates the GCNs [8]. However, the depth of neural networks and the usage of many steps in propagation are two complementary aspects. The usage of the shallow layered networks leads to bad compromises [8].

Clearly, using many propagation steps is crucial to determine similarly labeled nodes which reside in different/distinct regions of the graph. However, we cannot employ many graph convolutional layers due to the over smoothing problem, as highlighted in [7]. To demystify this problem, Li et al. [7] have shown that for a k-layer GCN, the influence score of node x on node y exhibits random walk alike properties on a graph. More specifically, they observe that influence score of x on y can be obtained by solving the first eigenvalue problem as $\pi^* = \widehat{\widetilde{\mathbf{A}}}\pi^*$, where π^* denotes the converged first eigenvector [7]. Obviously, this propagation relies more on graph rather than the node where the random walker has started to walk. To alleviate this problem, Klicpera et al. [8] exploits Random Walk with Restarts (RWR) approach to have the random walker return to the starting node once in a while. In the following subsection, we give brief overview of the idea that is used by Klicpera et al. [8] to integrate RWR into the message passing framework.

2.2. Existing solution to message passing via personalized propagation of neural predictions

From message passing to RWR: Original PageRank algorithm has been used in numerous applications [12] and it is computed by only solving the first eigenvalue problem, $\pi_{pr} = \mathbf{A}_{rw}\pi_p r$, where $\mathbf{A}_{rw} = \mathbf{A}\mathbf{D}^{-1}$ denotes the row-normalized adjacency matrix. To account for the influence score of the root node, starting nodes and its neighborhood, Klicpera et al. [8] used the personalized variant of PageRank [12], i.e, Random Walk with Restarts, as follows:

$$\pi_{ppr}(i_x) = \alpha(\mathbf{I}_n - (1-\alpha)\widehat{\widetilde{\mathbf{A}}})^{-1} i_x \tag{2}$$

where $\alpha \in (0,1)$ is the teleport probability that determines how much a random walker should span the graph and i_x is the indicator vector which only contains 1 in its xth entries and 0s in its all other entries. Clearly, $\Pi_{ppr} = \alpha(\mathbf{I}_n - (1-\alpha)\widehat{\widetilde{\mathbf{A}}})^{-1}$ is an $n \times n$ fully personalized PageRank matrix that contains influence score of x on y with its $\Pi_{ppr}(x,y) = \Pi_{ppr}(y,x)$ entry.

Personalized propagation of neural predictions (PPNP): In a

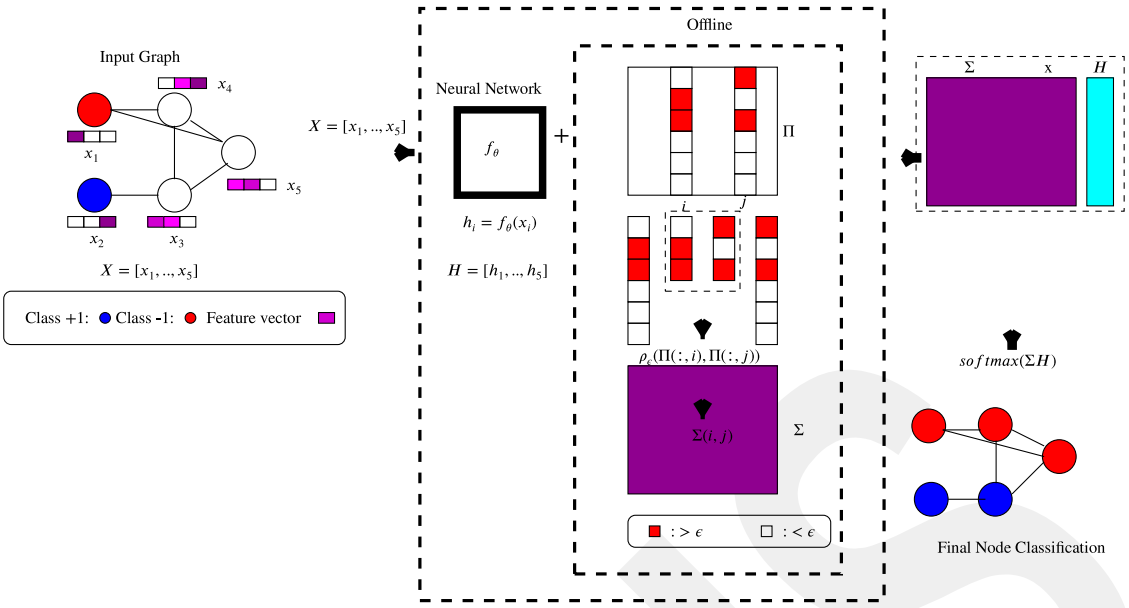

Fig. 1. Flowchart illustrating the proposed approach, for separating Graph Convolutional Network (GCN)'s Neural Network phase from its propagation phase. Given an undirected graph with features associated with all nodes and labels associated with some nodes (red and blue nodes in the first graph), we perform the following steps: (i) construction of $\mathbf{H} \in \mathbb{R}^{n \times c}$ matrix via training a Neural Network based on the feature information of the nodes, (ii) offline computation of Π matrix to obtain a low-dimensional topological profile (by eliminating the entries of topological profiles of node i and j if they both have a common white box (meaning entries less than given ϵ for both of them otherwise we keep the rows) for each node and construction of Σ, *Sparse Correlation Matrix*, (iii) further refinement of node the classification using Σ and $\mathbf{H}$.

Table 1
Descriptive Statics of Datasets.

Network	Type	Nodes	Edges	Classes	Features
Cora-ML	Citation	2810	7981	7	2879
CiteSeer	Citation	2110	3668	6	3703
Pubmed	Citation	19,717	44,324	3	500
MS-Academic	Co-Author	18,333	81,894	15	6805

nutshell, to utilize the fully personalized PageRank scores for node classification problem, Klicpera et al. [8] first generate classification prediction of each node based on its features and then propagate these predictions to gain more confidence about the final prediction via fully personalized PageRank matrix [8]. More formally, this formulation of node classification can be given as follows [8]:

$$\mathbf{Z}_{PPNP} = softmax\big(\Pi_{ppr}\mathbf{H}\big) \tag{3}$$

where $\mathbf{H} = f_\theta(\mathbf{X})$ and $\mathbf{X}$ denotes features matrix, f_θ is a neural network with parameter set θ generating the first predictions, $\mathbf{H} \in \mathbb{R}^{n \times c}$, before applying fully personalized PageRank and $softmax(x) = \frac{1}{\sum_{j=1}^{k} \exp(x_c)} exp(x)$ transforms predicted values into the probability density function [2].

As a result, PPNP [8] separates the message passing of GCNs from the neural networks that is used for generating prediction scores. Now, with the help of this formulation, the depth of neural networks are fully separated from the propagation phase of GCNs. Furthermore, Klicpera et al. [8] propose to compute Eq. (2) via classical power iteration to create a less accurate but more efficient variant of the propagation step as follows:

$$\begin{aligned} \mathbf{Z}^{(0)} &= \mathbf{H} = f_\theta(X), \\ \mathbf{Z}^{(k+1)} &= (1-\alpha)\widehat{\widehat{\mathbf{A}}}\mathbf{Z}^{(k)} + \alpha\mathbf{H}, \\ \mathbf{Z}^{(K)} &= softmax\big((1-\alpha)\widehat{\widehat{\mathbf{A}}}\mathbf{Z}^{(K-1)} + \alpha\mathbf{H}\big) \end{aligned} \tag{4}$$

where $k \in [0, K-2]$ denotes the dimension of power iteration.

3. Our approach

In this section we define our propose approach for training GCNs. Specifically, to balance the propagation towards intrinsic connectivity, we show that considering global correlation of nodes can be integrated into GCNs' massage passing framework.

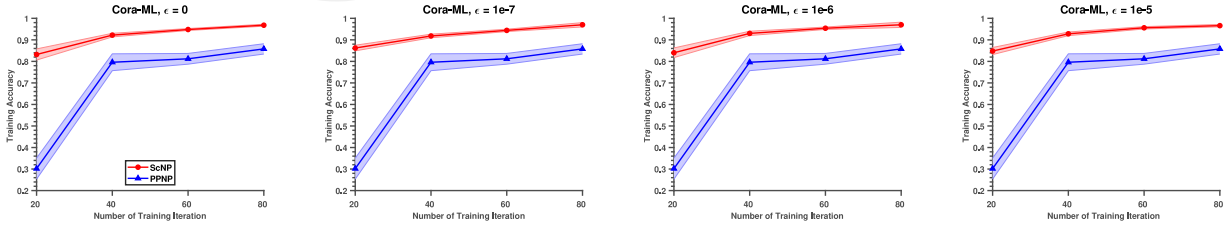

Fig. 2. Comparison of the performance of the proposed message passing method against that of the existing message passing algorithm on the Cora dataset. The plots show the mean and standard deviations of training accuracies of a GCN trained using ScNP and PPNP. The performance is shown as a function of training *epochs*.

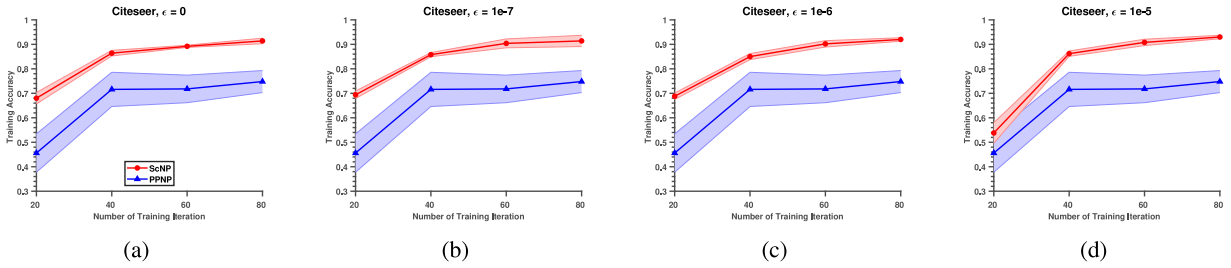

Fig. 3. Comparison of the performance of the proposed message passing method against that of the existing message passing algorithm on the Citeseer dataset. The plots show the mean and standard deviations of training accuracies of a GCN trained using ScNP and PPNP. The performance is shown as a function of training *epochs*.

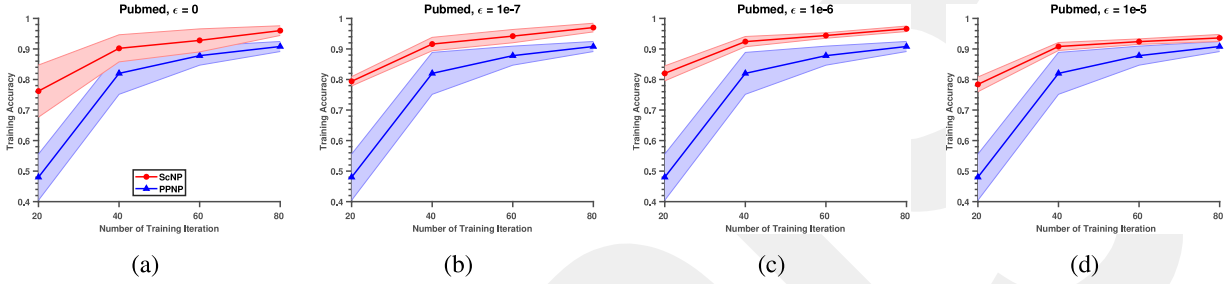

Fig. 4. Comparison of the performance of the proposed message passing method against that of the existing message passing algorithm on the Pubmed dataset. The plots show the mean and standard deviations of training accuracies of a GCN trained using ScNP and PPNP. The performance is shown as a function of training *epochs*.

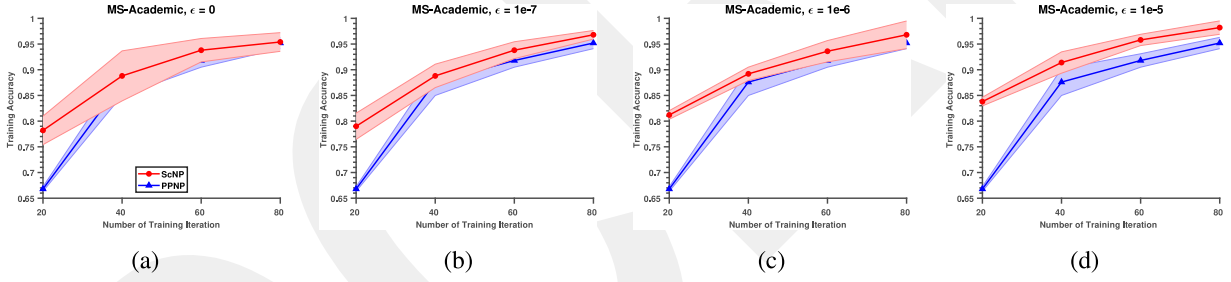

Fig. 5. Comparison of the performance of the proposed message passing method against that of the existing message passing algorithm on the MS-Academic dataset. The plots show the mean and standard deviations of training accuracies of a GCN trained using ScNP and PPNP. The performance is shown as a function of training *epochs*.

3.1. Proposed solution to the message passing scheme.

The main idea behind the proposed approach is that the proximity of nodes in a graph and the influence score of node x on y can be exploited more effectively by considering the *relative* position of nodes in the graph with respect to each other. In other words, if we consider the closeness of two nodes from perspective of similar to nodes, we might gain more insights in about the segregated propagation steps, i.e, the personalized PageRank. In the context of link prediction [13] and its application on the drug response prediction problem [15], we have shown that this approach is indeed more effective than the personalized PageRank like proximity measurements and generates much more accurate prediction results by drastically eliminating high dimensionality of PageRank vectors, such as, connectivity and/or centrality [16]. Elimination of such bias is particularly important for the node classification problem, since the final predictions, as discussed in [8] can be biased towards the high degree nodes and this could misguide the entire prediction process by favouring central unlabeled nodes.

Mathematically, it is clear from the Eq. (2) that the influence of a node x on all the other nodes, including itself, can be given with the vector $\pi_{ppr}(i_x)$. The influence of all nodes on all other nodes in the graph is given by the matrix, $\Pi_{ppr}I_n = \alpha(\mathbf{I}_n - (1-\alpha)\widehat{\widetilde{\mathbf{A}}})^{-1}I_n$. Klicpera et al. [8] propose their **PPNP** approach based on $\alpha(\mathbf{I}_n - (1-\alpha)\widehat{\widetilde{\mathbf{A}}})^{-1}$ matrix's entries distribution by considering individual entries' influences on themselves. Rather, here we propose to account for $\pi_{ppr}(i_x)$ vector as a whole to measure the influences of two nodes on each other. More formally, for a given $\Pi_{ppr} = \alpha(\mathbf{I}_n - (1-\alpha)\widehat{\widetilde{\mathbf{A}}})^{-1}$, fully computed personalized PageRank matrix, we rely on the proximity of correlations of two nodes x and y as $\rho(\Pi_{ppr}(:,x), \Pi_{ppr}(:,y))$, where ρ denotes the Pearson correlation of two column vectors corresponding profile vectors of nodes x and y, respectively. This way, we capture the global profile information of nodes from the perspective of all other nodes in the graph. Then, we take the closeness measure in terms of these vectors' entries all together, when we evaluate the proximity of two nodes, x and y. This approach has been shown to be much more informative than considering individual proximities separately in the context of disease gene prioritization [14,16], when one wants to classify nodes' belongings.

However, the high-dimensionality problem of vector $\Pi_{ppr}(:,x)$ for any node x in the graph may prevent us applying Pearson correlation to those profile vectors [13,15] due to sparsity of the vectors. Also, this

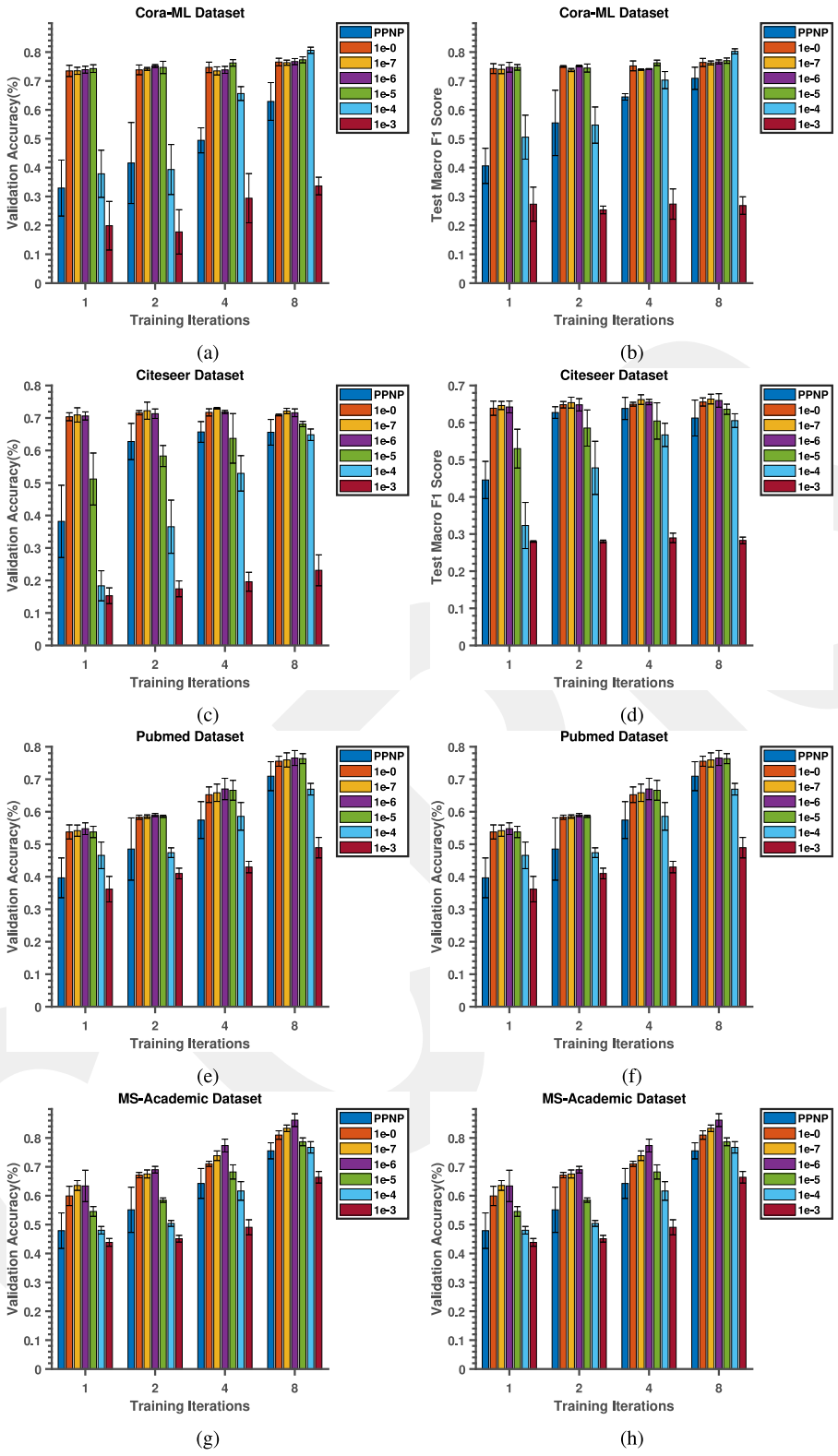

Fig. 6. The test accuracy and test Macro F1 scores of GCNs trained using our method and PPNP **with very limited training iterations.** In these experiments, we use various ϵ parameters across two datasets.

sparsity based approach can help to develop more efficient algorithms in terms of computational and storage cost, by suggesting not to compute the full Π_{ppr}.

To this end, we propose to reduce the high dimensionality of all the vectors, Π_{ppr}, by pruning the rows of any two vectors that are less than given parameter ϵ. That is, we delete the rows of $\Pi_{ppr}(:,x)$ and $\Pi_{ppr}(:,y)$, if they are smaller than the ϵ which helps us to prevent the small entries of both vectors acting as if they are correlated. This technique is named as *Sparse Correlation*, ρ_{ϵ} in [13]. In order to be consistent with the pioneering paper [13], we use the same terminology for the rest of this

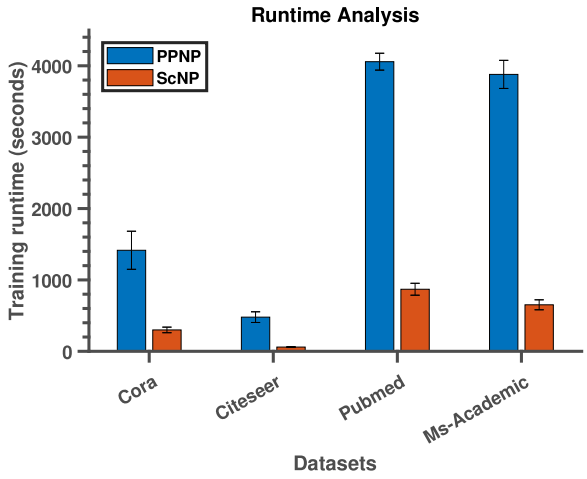

Fig. 7. Bar plot of training runtime of our proposed algorithm and the baseline algorithms as mean and standard deviation of 10 tuns.

Algorithm 1 Offline Construction of Σ

Compute $\Pi_{ppr} \in \mathbb{R}^{n \times n}$
for i=1:n **do**
  for j=1:i **do**
    $\Sigma(i,j) = \rho_\epsilon(\Pi_{ppr}(:,i), \Pi_{ppr}(:,j))$
$\Sigma = \Sigma + \Sigma^T$ ▷ $(.)^T$: Transpose of a matrix

Algorithm 1. Offline Construction of Σ.

paper.

Now, our objective should be clear that we aim at using sparsely correlated columns of Π_{ppr} in the context of node classification via GCN. More precisely, we construct a *Sparse Correlation Matrix*, Σ from $\Pi_{ppr}(:,x)$ and provide it to any GCNs to classify the nodes. The construction of this Σ matrix is given in the algorithm:

After computing the Σ matrix for the node classification problem, we define the following variant of GCN algorithm, Sparse correlation of Neural Predictions (ScNP):

$$\mathbf{Z}_{ScNP} = softmax(\Sigma\mathbf{H}) \tag{5}$$

In the final step of ScNP, we use *softmax* classifier to classify the nodes based on their learned features via neural networks and *Sparse Correlation Matrix*, Σ. With the proposed approach, we take the landmarks, which can be seen as the views of closely related nodes for the nodes that we want to classify, as oppose to a single entry view point as in [8]. The flowchart of our approach can be seen in Fig. 1.

4. Experiments

In this section, we systematically evaluate the performance of proposed algorithmic framework for the node classification problem. We start our discussion by describing the datasets and our experimental settings. Then, we analyze the performance of the algorithm as a function of the number of training iterations, *epochs*. We also compare the performance of our proposed methods against the best method presented as *full PPNP*, in [8]. We then investigate the performance of each algorithm as a function of very limited training iterations. Finally, we present training runtime of our algorithm and *full PPNP*, in [8].

4.1. Datasets and experimental setup

We test and compare the proposed method on four sets of real-world networks: `Cora`, `CiteSeer`, `Pubmed` and `MS-Academic` provided in [8,17]. Details of these four networks are given on Table 1. The datasets use a bag-of-words representation of the papers' abstracts as features, i.e., existence/non-existence of certain words are represented as 1/0 values in this feature vector [7].

For *PPNP*, we use the Python implementation provided by Klicpare et al. [8]. We implement our *ScNP* in Matlab which is only used for offline computations, and use Python for online computations. To invert the fully personalized PageRank matrix, Π_{ppr}, we use both Chopper [18] and I-Chopper [19].

We assess the performance of the algorithm as a function of training *epochs*. First, we fix the number of maximum training *epochs* to 80 and we present the training validation accuracy with standard deviation in 10 runs. Then, we evaluate the performance of the algorithm with very limited training *epochs*, namely for the values of $\{2^0, 2^1, 2^2, 2^3\}$ and present test validation accuracy and macro F-1 scores. Finally, we fixed the number of epoch to 2000 and evaluate the training runtime of algorithms, i.e., we report runtime of the algorithm when they are converged.

For the hyper-parameters of the GCNs, we follow Klicpare's [8] parameter settings. Namely, we use a two layered GCN with $h = 64$ hidden units. We apply L_2 regularization with $\lambda = 0.005$ on the weights of the first layer, use the dropout rate $d = 0.5$ on both layers, and use $\alpha = 0.1$ for RWR. Then, we report the mean accuracy of 10 runs for each dataset. Subsequently, for comparison, we use full *PPNP* method as the baseline method since it is the best method presented in [8] and it outperforms all its opponents methods (therein [8]). Finally, we report training time of both algorithms and it is observed that our approach requires much less training epochs than that of *PPNP* and thus our approach is at least 3-folds faster than *PPNP*. All of the experiments are performed on a Dell PowerEdge T5100 server with two 2.4 GHz Intel Xeon E5530 processors and 32 GB of memory.

4.2. Performance evaluation

We first compare the node classification performance of our method and full *PPNP* [8] method in terms of training accuracy, the number of correct prediction divided by the total number of prediction during the training phase of a GCN [7]. The results of this analysis are shown in Figs. 2–5 for four different datasets. As seen in the figures, the GCN that uses propagation based on our proposed algorithm delivers the best performance. To be more specific, the training accuracy of the GCN that uses Σ matrix instead of full personalized PageRank matrix Π in *PPNP* drastically outperforms its opponent, showing the validity of our method.

We then investigate the performance of our propagation algorithm as a function of very limited number of training *epochs* and evaluate the test accuracy and compare Macro F1 scores, which is the arithmetic mean of our per-class F1-scores, of the proposed method and its opponent. The result of this analysis is shown in Fig. 6. For varoius ϵ values, the left part of the Fig. 6 shows validation accuracy and the right side of the Fig. 6 depicts the Macro F1 scores across all four datasets. A seen in the figures, the test accuracy and Macro F1 scores obtained with our method surpass the test accuracy and Macro F1 scores obtained with *PPNP* method with very limited number of training *epochs*. It is very impressive that our method even renders a GCN to classify nodes with only 1 training *epoch* showing that in the propagation phase of a GCN, the use of ScPN instead of a PPNP is much more competent.

Finally, we investigate the training runtime of our proposed algorithms in Fig. 7. In this analysis, we fix the training epoch to 2000 and report running time of algorithms when they are converged. It can be clearly seen in the figure that our approach, *ScNP*, requires at least 3-folds less time that of *PPNP* [8]. We argue that this is because topological similarity based propagation matrix is well-clustered and we leave this direction of research as future work.

These results clearly demonstrate the effectiveness of topological similarity based propagation instead of the regular personalized PageRank approach, suggesting that this algorithm has great potential in

rendering GCNs message passing phase useful even when training with very limited number of *epochs*. These results demonstrate that our method presents a noel aspect for message passing phase of GCNs.

5. Conclusions

In this paper, we propose an algorithmic framework that utilizes both graph intrinsic topology and features to remedy the message passing phase of Graph Convolutional Networks. Testing our method on the real-world datasets, we demonstrate that our approach is highly effective in node classification problem and can be used for any graph convolution network tasks that require message passing, such as link prediction.

The work presented here has many feature directions including the use of different matrices in a GCN instead of a Laplacian matrix, the use of subgraph information instead of full graph in the message passing framework and the iterative computation of our method using effective Krylov based methods.

Declaration of Competing Interest

The authors whose names are listed immediately below certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakersbureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

References

[1] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks. International Conference on Learning Representations, 2017.
[2] F. Wu, T. Zhang, A.H.d. Souza Jr, C. Fifty, T. Yu, K.Q. Weinberger, Simplifying graph convolutional networks. International Conference on Machine Learning, 2019.
[3] R. Ying, R. He, K. Chen, P. Eksombatchai, W.L. Hamilton, J. Leskovec, Graph convolutional neural networks for web-scale recommender systems. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & 38; Data Mining, in: KDD'18, 2018.
[4] M. Zitnik, M. Agrawal, J. Leskovec, Modeling polypharmacy side effects with graph convolutional networks, Bioinformatics 34 (13) (2018) i457–i466.
[5] M. Coşkun, M. Koyutürk, Node similarity-based graph convolution for link prediction in biological networks, Bioinformatics 37 (23) (2021) 4501–4508.
[6] L. Yao, C. Mao, Y. Luo, Graph convolutional networks for text classification. Proceedings of the AAAI Conference on Artificial Intelligence vol. 33, 2019, pp. 7370–7377.
[7] Q. Li, Z. Han, X.-M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning. Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[8] J. Klicpera, A. Bojchevski, S. Gunnemann, Combining neural networks with personalized pagerank for classification on graphs. International Conference on Learning Representations, 2019.https://openreview.net/forum?id=H1gL-2A9Ym
[9] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs. Advances in Neural Information Processing Systems, 2017, pp. 1024–1034.
[10] S. Abu-El-Haija, B. Perozzi, R. Al-Rfou, A.A. Alemi, Watch your step: learning node embeddings via graph attention. Advances in Neural Information Processing Systems, 2018, pp. 9180–9190.
[11] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. vanden Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: A. Gangemi, R. Navigli, M.-E. Vidal, P. Hitzler, R. Troncy, L. Hollink, A. Tordai, M. Alam (Eds.), The Semantic Web, Springer International Publishing, Cham, 2018, pp. 593–607, https://doi.org/10.1007/978-3-319-93417-4_38.
[12] L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank Citation Ranking: Bringing Order to the Web. Technical Report, Stanford InfoLab, 1999.
[13] M. Coskun, M. Koyutürk, Link prediction in large networks by comparing the global view of nodes in the network. 2015 IEEE International Conference on Data Mining Workshop (ICDMW), 2015, pp. 485–492, https://doi.org/10.1109/ICDMW.2015.195.
[14] S. Erten, G. Bebek, R.M. Ewing, M. Koyutürk, Dada: degree-aware algorithms for network-based disease gene prioritization, BioData Min. 4 (1) (2011) 19, https://doi.org/10.1186/1756-0381-4-19.
[15] Z. Stanfield, M. Coşkun, M. Koyutürk, Drug response prediction as a link prediction problem, Sci. Rep. 7 (2017) 40321, https://doi.org/10.1038/srep40321.
[16] S. Erten, G. Bebek, M. Koyutürk, Vavien: an algorithm for prioritizing candidate disease genes based on topological similarity of proteins in interaction networks, J. Comput. Biol. 18 (11) (2011) 1561–1574, https://doi.org/10.1089/cmb.2011.0154.
[17] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, T. Eliassi-Rad, Collective classification in network data, AI magazine 29 (3) (2008) 93.
[18] M. Coskun, A. Grama, M. Koyuturk, Efficient processing of network proximity queries via Chebyshev acceleration. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, in: KDD'16, ACM, New York, NY, USA, 2016, pp. 1515–1524, https://doi.org/10.1145/2939672.2939828.
[19] M. Coşkun, A. Grama, M. Koyutürk, Indexed fast network proximity querying, Proc. VLDB Endow. 11 (8) (2018) 840–852, https://doi.org/10.14778/3204028.3204029.