Deep Graph Neural Networks for Unsupervised Graph Learning

by Chun Wang

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of Prof. Chengqi Zhang, Dr. Guodong Long and Dr. Shirui Pan

University of Technology Sydney
Faculty of Engineering and Information Technology

December 2020

Certificate of Authorship/Originality

I, Chun Wang declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Production Note:
Signature removed
prior to publication.

2021/06/20

ABSTRACT

Deep Graph Neural Networks for Unsupervised Graph Learning

by

Chun Wang

Graphs are widely used to represent networked data, which contains complex relationships among individuals, and therefore cannot be well represented by traditional flat-table or vector format. Network applications, like social networks or citation networks, have been developing rapidly in recent years. Consequently, graph learning has also attracted much more attention.

Unsupervised graph learning is an important branch of the field since label information is usually not easily accessible. It is much more challenging as unsupervised graph learning aims to model the networked data without training supervision. Associated downstream tasks of unsupervised graph learning may include clustering, link prediction, visualization, etc., which are also very popular in modern graph analysing.

To perform unsupervised graph learning, previous methods may (1) consider only one aspect of the graph information; (2) use shallow approaches to capture simple or linear relationships among the data; (3) separate graph embedding learning and the downstream task as two steps and learn sub-optimal results since the learned embedding could not best fit the downstream method; (4) not able to manage corrupted data and perform robust learning, and (5) not able to make use of side information. These limitations have held back the development of unsupervised graph learning.

Therefore, we aim to address of the following challenges in our research: (1) How to characterize the individual properties of each node, and at the same time capture complex relationships in the graph; (2) How to learn deep and informative representation for graph data; (3) How to perform end-to-end learning for a certain graph data-based task; and (4) How to deal with different types of abnormal graph data information.

In this thesis, we aim to perform effective graph learning, with deep graph neural networks in an unsupervised manner. Firstly, we propose a special marginalized graph autoencoder, to integrate both node content and graph structure information into a unified framework. We add noise to the graph data, and employ a marginalized process for efficient computation. By further stacking multiple layers of such autoencoder, we learn deep and informative unsupervised graph embedding for graph clustering; Secondly, we combine graph autoencoder with a self-training model, to conduct a goal-directed training framework. In such a process, the clustering and embedding learning are performed simultaneously. Both of them can benefit from the other, thereby learn better graph embedding and clustering. Facing possible data corruption, especially structural corruption for graph data, we develop a dual-autoencoder interaction framework Cross-Graph, which takes advantage of the deep learning memorization effect that DNNs fit clean and easy data first. Two autoencoders filter out untrusted edges alternatively and learn robust embedding from graphs with redundant edges. Finally, to take advantage of possible side information in graph learning, we also propose a contrastive regularized graph autoencoder, that can improve the unsupervised graph learning ability using constraint information. All these frameworks are validated with unsupervised tasks like clustering in the experiments.

Dissertation directed by Dr. Guodong Long, Dr. Shirui Pan and Prof. Chengqi Zhang

Australian Artificial Intelligence Institute, Faculty of Engineering and Information Technology

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisors: Dr. Guodong Long, Dr. Shirui Pan, and Prof. Chengqi Zhang. They have continuously provided me support and guidance in the past four years. I could not go through my Ph.D. study without their patient guidance, immense knowledge and help on all sides.

I am also grateful to Dr. Bo Han and Dr. Fan Zhang, they are both friends and mentors who have also provided me a lot of guidance and help during my research. And also thanks to Dr. Jing Jiang, who has supported me along with my study.

I would also like to thank my school mates and research fellows, who had a positive influence on my Ph.D. study, including but not limited to: Ruiqi Hu, Tao Shen, Xiaolin Zhang, Zonghan Wu, Lu Liu, Han Zheng, Fengwen Chen, and Mengyao Li. They are the ones who share both my joyful and stressful times and help me from different aspects.

Finally, I would like to thank my family. No words could possibly express my gratitude for their endless love, encouragement and support.

Chun Wang

Sydney, Australia, 2020.

List of Publications

[1] Chun Wang, Shirui Pan, Guodong Long, Xingquan Zhu, Jing Jiang, “MGAE: Marginalized graph autoencoder for graph clustering”, CIKM 2017 (CORE rank A, citations: 106)

[2] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Chengqi Zhang, “Attributed graph clustering: A deep attentional embedding approach”, IJCAI 2019 (CORE rank A*, citations: 56)

[3] Chun Wang, Bo Han, Shirui Pan, Jing Jiang, Gang Niu, Guodong Long “Cross-Graph: Robust and Unsupervised Embedding for Attributed Graphs with Corrupted Structure”, ICDM 2020 (CORE rank A*)

[4] Chun Wang, Shirui Pan, Celina P Yu, Ruiqi Hu, Guodong Long, Chengqi Zhang, ”Deep Neighbor-aware Embedding for Node Clustering in Attributed Graphs”, Pattern Recognition (CORE rank A*, under the 3rd review process with minor revision decision)

[5] Chun Wang, Shirui Pan, Bo Han, Guodong Long, Chengqi Zhang, ”Constrained Node Clustering for Attributed Graphs with Regularized Autoencoder”, (It will be submitted to a CORE rank A journal in 2021)

Contents

Certificate ii

Abstract iii

Acknowledgments v

List of Publications vi

List of Figures xi

Abbreviation xvi

Notation xvii

1 Introduction 1

1.1 Background 1

1.2 Research Objectives 3

1.3 Thesis Organization 6

2 Literature Review 8

2.1 Graph Learning Overview 8

2.1.1 Deep Neural Networks for Graphs 8

2.1.2 Graph Embedding Models 9

2.1.3 Node Clustering in Graphs 10

2.2 Techniques Employed in Our Frameworks 11

2.2.1 Autoencoder 11

2.2.2 Deep Clustering Algorithms 12

2.2.3 Co-training based Methods 12

2.2.4 Outlier-Oriented Graph Models 13

2.2.5 Contrastive Learning 13

2.2.6 Constrained Clustering 14

2.3 Baseline Methods 14

3 Learning Using Two-aspects Information MGAE: Marginalized Graph Autoencoder for Graph Clustering 17

3.1 Background 17

3.2 Problem Definition 23

3.3 Proposed Method 23

3.3.1 Graph Convolutional Network 24

3.3.2 Marginalized Graph Autoencoder (MGAE) 26

3.3.3 Graph Clustering Algorithm 30

3.4 Experiments 32

3.4.1 Benchmark Datasets 32

3.4.2 Baseline Methods 33

3.4.3 Evaluation Metrics & Parameter Settings 34

3.4.4 Experiment Results 35

4 Learning with Goal-directed Framework Deep Neighbor-aware Embedding for Node Clustering in Attributed Graphs 41

4.1 Background 41

4.2 Problem Definition and Overall Framework 44

4.2.1 Overall Framework 45

4.3 Proposed Method 45

4.3.1 Graph Autoencoder 45

4.3.2 Self-optimizing Embedding 51

4.3.3 Joint Embedding and Clustering Optimization 53

4.4 Experiments 54

4.4.1 Benchmark Datasets 54

4.4.2 Baseline Methods 54

4.4.3 Evaluation Metrics & Parameter Settings 57

4.4.4 Experiment Results 58

5 Learning Corrupted Graph Data Cross-Graph: Robust and Unsupervised Embedding for Attributed Graphs with Corrupted Structure 69

5.1 Background 69

5.2 Problem Definition 73

5.3 Proposed Method 74

5.3.1 Graph Autoencoder 74

5.3.2 Cross-Graph Learning Framework 76

5.3.3 Algorithm Description and Deeper Insights 77

5.3.4 Time Complexity Analysis 80

5.4 Experiments 81

5.4.1 Datasets 81

5.4.2 Baselines 82

5.4.3 Node Clustering on Corrupted Data 82

5.4.4 Link Prediction on Corrupted Data 88

5.4.5 Experiments on Uncorrupted Data 91

5.4.6 Network Visualization on Corrupted Data 91

6 Learning from Side Information Constrained Graph Clustering with Contrastive Regularized Autoencoder 93

6.1 Background 93

6.2 Problem Definition 96

6.3 Proposed Method 96

6.3.1 Graph Autoencoder 97

6.3.2 Contrastive Regularizer 99

6.3.3 Joint Embedding Learning 100

6.3.4 Framework Description 100

6.4 Experiments 101

6.4.1 Benchmark Datasets 101

6.4.2 Baselines 101

6.4.3 Evaluation Metrics 103

6.4.4 Experimental Setup 103

6.4.5 Experimental Results 104

6.4.6 Parameter Study 105

6.4.7 Network Visualization 108

7 Conclusion 110

List of Figures

3.1 The effectiveness of using marginalization for graph clustering. Marginalization introduces a small amount of disturbance to the node content, resulting in a dynamic environment for node content and structures to interact. Because the optimization process is well informed in relation to data disturbance, marginalization will cancel out the disturbance and the underlying graph autoencoders can learn optimized outcomes. Results are based on the accuracy (ACC) and normalized mutual information (NMI) of the spectral clustering before and after marginalization. 21

3.2 Conceptual framework of Marginalized Graph Autoencoder (MGAE) for graph clustering. Given a graph $G = (V, E, X)$, MGAE firstly learns a graph convolutional network (GCN) by using a mapping function $f(\widetilde{X}, A)$ based on the adjacency matrix A and corrupted node content $\widetilde{X}$. By minimizing the error between the output of GCN $f(\widetilde{X}, A)$ and X, we will get a latent representation $Z^{(1)}$. By stacking multiple GCNs and performing layer-wise training, our algorithm can learn a deep representation $Z^{(\Gamma)}$. Finally, a spectral clustering algorithm is performed on the refined representation $Z^{(\Gamma)}$ of the last layer. 24

3.3 Parameters study on noise and number of layers. 36

3.4 Runtime comparisons of different methods. 39

3.5 2D visualization on representations learned from MGAE of various layers. 40

4.1 The difference between two-step embedding learning models and our model. 43

4.2 The conceptual framework of Deep Neighbor-aware Embedded Graph Clustering (DNEGC). Given a graph $G = (V, E, X)$, DNEGC learns a hidden representation Z through a graph autoencoder, and manipulates it with a self-training clustering module, which is optimized together with the autoencoder and perform clustering during training. The two variants share similar framework and differ as their autoencoder encode the inputs through different strategy. . . 46

4.3 Parameters study on clustering coefficient γ. 62

4.4 Parameters study on embedding size. 64

4.5 Parameters study on number of layers. 65

4.6 2D visualization of various methods using the t-SNE algorithm on the Cora and Citeseer dataset. 66

4.7 2D visualization of the DNEGC-Att algorithm using the t-SNE algorithm on the Cora and Citeseer dataset during training (the top line for the Cora dataset, and the bottom line for the Citeseer dataset). The first visualization of each line illustrates the embedding training with the graph autoencoder only, followed by visualizations showing subsequent equal epochs in which the self-training component is included, till the last one being the final embedding visualization. 67

5.1 We randomly add spurious edges to the Cora and Citeseer graph structure and then run Graph Autoencoder (GAE) on them for 10 times to record the average clustering performance evaluated by 3 clustering metrics. The X-axis shows the number of added edges represented as percentage to the number of original edges. It shows that spurious edges can easily ruin the performance of the graph. . . 71

5.2 Conceptual framework of Cross-Graph Autoencoder. Given a graph G with graph structure A and node content X, we maintain two autoencoders. Each autoencoder encodes A and X into a latent embedding Z, and then a decoder tries to reconstruct the structure A from Z and obtains A'. We regard A' partly as a reliability score of the edges in A and manipulate A according to it. Two updated A's are thereby formed and passed to the peer-autoencoder as the input for the next iteration. 74

5.3 The Cross-Graph Working Mechanism. The reconstruction level (the reconstruction's similarity to the input graph) could show its opinion of the possibility of each edge being spurious, represented by the sparsity degree of the dotted line. So, based on it and the input graph, our Cross-Graph constructs a new updated graph, which devalue those suspected edges. The edge value is positively correlated with the reconstruction level, represented by the gradation of the edge color. This updated graph is provided to the peer-autoencoder for the next iteration update. 79

5.4 The devalue of two types of edges (original edges and redundancy edges we added) along with the Cross-Graph training process. 87

5.5 The clustering performance under different percentage of redundancy edge corruption. 88

5.6 The clustering performance with different Cross-Graph coefficient γ. . 89

5.7 Box plots of the 20 times' clustering accuracy from the Cross-Graph Dual-autoencoders interactive process, compared with a single autoencoder framework. For each box, the five lines from top to bottom represents the maximum, the first quartiles, the sample median, the third quartiles and the minimum of the clustering results. 89

5.8 The devalue of inner-edges and inter-edges comparison along with the Cross-Graph training process on uncorrupted Cora dataset. . . . 90

5.9 Visualizations of the corrupted Cora dataset, based on the embedding learned from various algorithms. The dots represent nodes and the seven different colors represent the ground-truth clusters the nodes belongs to. 92

6.1 The difference between constrained node clustering and normal node clustering. Constraint information (the green arrow in the Figure) can define some pairs of nodes that should be in the same cluster, and therefore may help partition some equivocal marginal nodes to the right cluster. 95

6.2 Conceptual framework of the Contrastive Regularized Autoencoder for Constrained Graph Clustering. Given a graph G with graph structure matrix A, node content matrix X, and a list of node pairs that are constrained to the same cluster, we aim to learn effective clustering assignment. We train a graph autoencoder to integrate graph structure and node content information into a latent node embedding. When optimizing the autoencoder reconstruction loss, we together minimize a contrastive loss function that forces the constrained node pairs to learn similar embedding at the same time. The learned embedding is influenced by all three aspects of the information and used for clustering. 97

6.3 Average clustering performance evaluated by ACC and NMI, with different temperature parameter τ, from 0.01 to 0.1. 105

6.4 Average clustering performance with different number of constrained node pairs. The percentage is based on the total potential constraint links N_m. 106

6.5 Visualizations of the Cora and Citeseer datasets, based on the embedding learned from GAE and our CRA. The dots represent nodes and the seven different colors represent the ground-truth clusters the nodes belong to. 107

Abbreviation

NMF - Non-negative Matrix Factorization

GCN - Graph Convolutional Network

GAT - Graph Attention Network

DNN: Deep Neural Networks

AE - Autoencoder

GAE - Graph (Convolutional) Autoencoder

DA - Denoising Autoencoder

SDA - Stacked Denoising Autoencoder

DEC - Deep Embedded Clustering

KL - Kullback-Leibler

2D: Two-dimensional

SGD: Stochastic Gradient Descent

NMI: Normalized Mutual Information

AE: Average Entropy

ARI: Adjusted Rand Index

Nomenclature and Notation

Capital letters denote matrices.

Lower-case alphabets denote column vectors.

$(.)^T$ denotes the transpose operation.

I_n is the identity matrix of dimension $n \times n$.

$\mathbb{R}$, $\mathbb{R}^+$ denote the field of real numbers, and the set of positive reals, respectively.

n is the number of nodes in the graph.

m is the number of individual attributes of each node.

k is the number of clusters for the node clustering task.

X is a $n \times m$ matrix representing the node attribute information, each row x_n is a m-dimensional vector representing the attribute values of node n.

A is a $n \times n$ adjacency matrix representing the graph structure information, in which $A_{i,j} = 1$ representing there is an edge between node i and j, and $A_{i,j} = 0$ otherwise.

$\|.\|_F^2$ represents the squared Frobenius norm.

$\tilde{.}$ represents the corrupted or approximated version.

$tr(.)$ represents the trace of the matrix.

$Z^{(l)}$ is the learned node embedding in the l-th neural network layer.