KTH Royal Institute of Technology

Machine Learning, Advanced Course

DD2434

Project – Large Network representation learning

Group 2
Jiaming Huang
Tianzhang Cai
Wenhao Luo
Zhengjie Ji

January 15, 2021

Abstract We reproduce the empirical study on unsupervised network representation learning (UNRL) approaches over graphs, considering 5 UNRL techniques and two common tasks – node classification and link prediction. We found that ?(Ex: We find that for non-attributed graphs there is no single method that is a clear winner and that the choice of a suitable method is dictated by certain properties of the embedding methods, task and structural properties of the underlying graph. In addition we also report the common pitfalls in evaluation of UNRL methods and come up with suggestions for experimental design and interpretation of results.)

1 Introduction

In this project we reproduce the results of the article "A Comparative Study for Unsupervised Network Representation Learning **a**" according to the requirements of the assignment. The original article conducted a large-scale study on multiple unsupervised network representation learning algorithms, and made a systematic comparison on their performance in different tasks. We implemented five methods mentioned in the article: DeepWalk, Node2vec, Line, NetMF and GraphSage, and apply them to two tasks: node classification and link prediction, evaluating on different datasets. Based on our results, we verify and further explore the research results of the article.

1.1 Study Background

To understand how various methods differ in definitions and treatment of similarity, we introduce the concept of context graph. A context graph is defined as a graph with the weight of edge (u,v) denote the similarity between u and v, where negative weight means dissimilarity between the two nodes. And the representation of a node is divided into "source" and "context". The source role of node v is obtained by the observable edges, and the context of v is its "one-hop neighbor" **a**. We would like to use different methods to learn and analyze the source and context representation of nodes.

1.2 Research Questions

In our experiment, we hope to answer the following research questions. These questions are adapted from that asked in the original paper **a**.

1. How does the choice of different methods affect the performance of assigned tasks?
2. How does the choice of different ways of exploiting the context affect the performance of network representation learning methods?
3. How does the choice of optimization method affect the performance? Do deep models always outperform the shallow models?
4. Do these methods have good generalization ability to new datasets?

2 Procedure

The procedure of our experiment are as follows, according to the requirements of the project assignment.

1. Implement five methods discussed in the paper: DeepWalk, Node2vec, Line-1, NetMF, and GraphSage.
2. Collect all the datasets presented in the paper and a few additional datasets.
3. Evaluate the different methods on the data we collected with respect to the tasks of link prediction and node classification.
4. Discuss our findings and give answer to the research questions we posted based on our results.
5. Compare our results to the results of the paper, and discuss the reproducibility of the study.
6. Discuss the original paper in terms of approach, methodology, and conclusions.

3 Methods

In this part, we introduced the principle of the algorithms that we re-implemented and our extensions of these methods.

3.1 Algorithm Description

In this part, we introduce the five unsupervised embedding methods analyzed in this project, DeepWalk, Node2vec, Line, NetMF and GraphSage.

method	Formulation of Context Graph	Learned Embeddings	Optimization Methods
DeepWalk	Random Walk	Source and Context	Hierarchical Softmax
Node2vec	Random Walk	Source and Context	Negative Sampling
LINE-1	Adjacency Based	Source	Negative Sampling
NetMF	Context Matrix	Source and Context	Context Matrix Factorization
GraphSage	Random Walk	Source	Neighborhood Aggregation and Negative Sampling

Table 1: Comparison of methods in terms of schemes of defining context, exploitation of context and optimization methods.

3.1.1 Random Walk Based Methods

DeepWalk utilizes random walk to sample a sequence of graph nodes where for each node, its successive node is randomly selected from its neighbors. After the random walk sequence is sampled, a window of a certain width slides through it and at each position, the node at the window center forms a *context edge* with each of the rest nodes in the window. For these context edges C, we want to find the graph embedding Φ and θ such that the loss

$$-\sum_{(u,v)\in C} \log P(v \mid u) = -\sum_{(u,v)\in C} \log \frac{\exp(\Phi_u \cdot \theta_v)}{\sum_{w\in V} \exp(\Phi_u \cdot \theta_w)},$$

is minimized, which is equivalent to maximizing the likelihood of the co-occurrence of nodes in the same context. However, note that in the denominator there is a sum over all nodes in the graph, which makes the objective function computationally very expensive. To deal with this, two methods have been proposed: hierarchical SoftMax and negative sampling. In the original paper, DeepWalk adopts hierarchical SoftMax, which constructs a Huffman binary tree in which each graph node is a leaf **b**. In this manner, the log probability is approximated by traversing the tree bottom up, thus reducing the time complexity to $O(\log V)$. After the loss is obtained, we use back-propagation and gradient descent to optimize the graph embedding.

Node2Vec The major difference between DeepWalk and Node2Vec is the probability distribution of the neighborhood selection. For DeepWalk, the distribution is uniform - each neighbor is equally likely to be selected. For Node2Vec, the distribution of neighbors is dependent on the list of nodes in the random walk - more specifically, dependent on the node preceding the current node in the walk, which is called the second-order neighborhood **f**. Intuitively, the probability distribution of the second-order neighborhood in Node2Vec is controlled by 2 parameters p and q - they control the probabilities of the next node coming back or walking away from the previous one respectively. Hence, Node2Vec is actually a generalization of DeepWalk. When $p = q = 1$, the neighborhood distribution in Node2Vec is unbiased and thus the same as that in DeepWalk.

3.1.2 Adjacency Based Methods

LINE d aims to preserve both local and global network structures in embedding. "First order proximity" is captured by observable links between vertic es. And "Second order proximity" is determined through the neighbors shared between vertices. LINE-1 optimizes the "First order proximity", which preserves the local network structure, and LINE-2 deal with the "Second order proximity" and global structure. The optimization is done by stochastic gradient descent. To improve the efficiency and to avoid a huge weight multiplying to the gradients, LINE samples the edges with probability proportional to their weights.

3.1.3 Matrix Factorization Based Methods

NetMF This method unified LINE, PTE, DeepWalk and Node2vec in the framewalk of matrix factorization. It directly factorizes the context matrix with (i, j)th element given by **e**:

$$c_{i,j} = log(\frac{vol(G)}{kT}(\frac{1}{d_j} \cdot (\sum_{r=1}^{T} P^r)_{i,j}))$$

where T is the window size, r is the walk length and k is the number of negative samples in DeepWalk. For small window size T, we first compute the DeepWalk matrix M, and then factorize (perform SVD) to get the network embedding. For large window size T, it is not feasible to directly compute the DeepWalk matrix M as the time complexity can be too high. A solution to this is to approximate the DeepWalk matrix M and show its low-rank nature by using the relationship of M with the normalized graph Laplacian.

3.1.4 Neural Network Based Methods

GraphSage is a framework of graph neural network that leverages neighborhood aggregation to inductively learn the topological structure of each node's neighborhood and incorporate them into the node embeddings. A variety of *aggregator functions* can be used in the GraphSage framework. The simplest one is the *mean* function, which aggregates the neighborhood information by taking the mean of the embedding of each neighbor. Other more advanced aggregators include LSTM (long-short term memory) and MaxPooling (which utilizes another single-layer neural network). For each node, the neighborhood aggregation starts from its Kth-order neighborhood and propagates layer by layer until the information is aggregated from its immediate neighbors. Thus K is the number of layers in the neural network. After that the embedding of the node is updated as the aggregated information, and then the loss of the embeddings of nodes in a minibatch is calculated by using random walk and negative sampling. After optimizing node embeddings, they should properly represent the topological structure of the graph.

3.2 Task Description

In this part, we describe the two tasks that we used for comparing the Unsupervised Network Representation Learning (UNRL) methods.

3.2.1 Link Prediction (LP)

The first task is to predict missing edges of a network. In the original paper describing the experiments of Link Prediction **g**, the author work with the subgraphs of a social network G with potentially different time-stamps. In this article **a**, we instead evaluate a network with a fraction of removed edges (no vertex is isolated due to the removal). The residual network is used for training, and the removed edges are served as the test split. Also, we sample negative edges to balance the test split. In the training process, we measure the distance (similarity) between pair of vertices using four different methods: common neighbors, Jaccard's coefficient, Adamic/Adar and preferential attachment. Then we rank the node-to-node similarity and predict links according to the rank. To obtain the accuracy of the prediction, we test our prediction on the test split and calculate the ROC-AUC (Area Under the Receiver Operating Characteristic Curve) scores.

3.2.2 Multilabel Node Classification (NC)

The second task is to do node classification on a graph where each node has one or more labels. We do the multilabel classification using one-vs-rest logistic regression. Then, we test the accuracy using 5-fold cross validation, and report the mean Micro-F1 and Macro-F1 scores.

4 Implementation

4.1 Datasets

In this part, we introduce the datasets used in our experiment for different tasks. For the Reddit, Youtube and Twitter, due to the limitation of computation resources, it is difficult for us to conduct experiment on these extreme large datasets, so we are unable to present results on them. For node classification, we test the different algorithms on BlogCatalog, Cora, Pubmed, Cocit, Flickr. For link prediction, Blogcatalog, Flikcr, Cora, DBLP-Ci and Epinion were used. The detail of these datasets are summarized in table 2.

	BlogCatalog	Cora	Pubmed	Cocit	DBLP-Ci	Epinion
Category	SN	Citation	Citation	Citation	Citation	SN
Type	undir	dir	dir	dir	dir	dir
$\|V\|$	10k	2.7k	19k	44k	12.5k	75k
$\|E\|$	333k	5.4k	44k	195k	49k	508k

Table 2: Summary of datasets used in our experiment. SN denotes social networks.

4.2 Parameter settings

Most of the parameters are the same as the parameters in **1**. Part of them are modified due to some special reasons.

Parameter	Value	Parameter	Value
Embedding dimension d	128	Walk length t	40
Number of Walk r	80	Window size w	10
Negative Sample Size ns	5	LINE iteration	300000
Number of Eigenpairs h	256	NODE2VEC p,q	0.25,4

Table 3: Parameters we used and the corresponding values.

The iteration number for LINE is reduced from 10 billion to 300000, mainly because it is too time-consuming and we cannot get a result in time by our hardware.

5 Results

In this part, we demonstrate the results that we obtained applying each method to the two tasks discussed above.

5.1 Link Prediction (LP)

Here we demonstrate the result for link prediction. We evaluate the five methods mentioned above on four datasets: BlogCatalog, Cora, Lesmis (additional) and Facebook (additional).

method	BlogCatalog	Cora	PubMed	Lesmis	Facebook
DeepWalk	0.433	0.000	0.000	0.000	0.000
Node2vec	0.615	0.000	0.000	0.000	0.000
LINE-1	0.000	0.000	0.000	0.000	0.000
NetMF	0.000	0.000	0.000	0.000	0.000
GraphSage	0.000	0.000	0.000	0.000	0.000

Table 4: Link Prediction results using 50% edges as training data, balanced with negative edges. Blue indicates that our results are better than the original paper, and vice versa.

5.2 Multilabel Node Classification (NC)

Here we demonstrate the result for Multilabel Node Classification. We evaluate the five methods mentioned above on three datasets: BlogCatalog, Cora, PubMed and Chameleon (additional). In most cases we achieved lower scores than that of the original paper. blablabla

method	BlogCatalog		Cora		PubMed		Chameleon	
	mic.	mac.	mic.	mac.	mic.	mac.	mic.	mac.
DeepWalk	**0.336**	**0.195**	0.000	0.000	0.645	0.548	0.000	0.000
Node2vec	0.192	0.088	0.000	0.000	0.603	0.558	0.000	0.000
LINE-1	0.161	0.064	0.000	0.000	0.628	0.540	0.000	0.000
NetMF	0.191	0.091	0.824	0.813	**0.808**	**0.794**	0.571	0.562
GraphSage	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table 5: Multilabel Node Classification results in terms of Micro-F1 and Macro-F1. All results are mean of 5-fold cross validations. Blue indicates that our results are better than the original paper, and vice versa.

6 Discussion

6.1 Discussion of Our Findings

In this part, we discuss our findings and give answer to the research questions in the Introduction section based on our results.

Inspect into our result, we can see that blabla

Now, we give answer to the research questions.

1. How does the choice of different methods affect the performance of assigned tasks?
   Answer: blabla

2. How does the choice of different ways of exploiting the context affect the performance of network representation learning methods?
   Answer: blabla

3. How does the choice of optimization method affect the performance? Do deep models always outperform the shallow models?
   Answer: blabla

4. Do these methods have good generalization ability to new datasets?
   Answer: blabla

6.2 Comparison of Results

6.3 Discussion about the Original Paper

7 Conclusions

Here we write something.