IRE Major Project Report

Team 43, IIIT Hyderabad 14/04/2016

1 Problem Statement

Goal: **Learning representations for nodes in a social network**

Recently there has been an increasing attention to use Deep Learning(DL) techniques to analyze social graphs, such as Flickr, Youtube, Twitter and so on. The beauty of such solution is that once DL is applied, several network mining tasks such as node classification, link prediction, node visualization, node recommendation can be solved by conventional machine learning algorithms.

In this project, we will build a model that can capture the network information of a node in an efficient and scalable manner. These learned representations will be used to do nodes classification in our project.

2 Abstract

This project studies the problem of embedding very large information networks into low-dimensional vector spaces, which is useful in many tasks such as visualization, node classification, and link prediction. Most existing graph em- bedding methods do not scale for real world information networks which usually contain millions of nodes. In this paper, we implemented a network embedding method called the "LINE," which is suitable for arbitrary types of informa- tion networks: undirected, directed, and/or weighted. The method optimizes a carefully designed objective function that preserves both the local and global network structures. An edge-sampling algorithm is proposed that addresses the limitation of the classical stochastic gradient descent and improves both the effectiveness and the efficiency of the inference. Empirical experiments prove the effectiveness of the LINE on a variety of real-world information networks,including language networks, social networks, and citation networks. The algorithm is very efficient, which is able to learn the embedding of a network with millions of vertices and billions of edges in a few hours on a typical single machine.

3 Description

This Project aims to learn the representation of nodes in social network using embedding very large information networks into low-dimensional vector spaces, which is useful in many tasks such as visualization, node classification, and link prediction. Most existing graph embedding methods do not scale for real world information networks which usually contain millions of nodes. This project uses the approach called LINE .

3.1 Key Definitions

Definition 1. **Information Network** An information network is defined as G = (V, E), where V is the set of vertices, each representing a data object and E is the set of edges between the vertices, each representing a relationship between two data objects. Each edge e ∈ E is an ordered pair e = (u, v) and is associated with a weight w uv > 0, which indicates the strength of the relation. If G is undirected, we have (u, v) = (v, u) and w uv = w vu ; if G is directed, we have (u, v) ≠ (v, u) and $w_{uv} \neq w_{vu}$

Definition 2. **First-order Proximity** The first-order proximity in a network is the local pairwise proximity between two vertices. For each pair of vertices linked by an edge (u, v), the weight on that edge, w uv , indicates the first order proximity between u and v. If no edge is observed between u and v, their first-order proximity is 0.

Definition 3. **Second-order Proximity** The second order proximity between a pair of vertices (u, v) in a network is the similarity between their neighborhood network structures. Mathematically, let p u = (w u,1 , . . . , w u,|V |) denote the first-order proximity of u with all the other vertices, then the second-order proximity between u and v is determined by the similarity between p u and p v . If no vertex is linked from/to both u and v, the second-order proximity between u and v is 0.

Definition 4. **Large-scale Information Network Embedding** Given a large network G = (V, E), the problem of Large-scale Information Network Embedding aims to represent each vertex v ∈ V into a low-dimensional space R d , i.e., learning a function f G : V → R d , where d « |V |. In the space R d , both the first-order proximity and the second-order proximity between the vertices are preserved.

3.2 Line: LARGE-SCALE INFORMATION NETWORK EMBEDDING

It is a desirable embedding model for real world information networks must satisfy several requirements: first, it must be able to preserve both the first-order proximity and the second-order proximity between vertexes; second, it must scale for very large networks, say millions of vertexes and billions of edges; third, it can deal with networks with arbitrary type of edges: directed, undirected and / or weighted. In project we will be using LINE which satisfies all the three requirements.

The method optimizes :

1. a carefully designed objective function that preserves both the local and global network structures.
2. An edge-sampling algorithm is proposed that addresses the limitation of the classical stochastic gradient descent and improves both the effectiveness and the efficiency of the inference. Empirical experiments prove the effectiveness of the LINE on a variety of real-world information networks

The algorithm is very efficient, which is able to learn the embedding of a network with millions of vertexes and billions of edges in a few hours on a typical single machine.

3.3 Related Study

Various methods of graph embedding have been proposed in the machine learning literature. They generally perform well on smaller networks. The problem becomes much more challenging when a real world information network is concerned, which typically contains millions of nodes and billions of edges. For example, the Twitter followee-follower network contains 175 million active users and around twenty billion edges in 2012 . Most existing graph embedding algorithms do not scale for networks of this size. The time complexity of classical graph embedding algorithms such as MDS, IsoMap, Laplacian eigenmap are at least quadratic to the number of vertices, which is too expensive for networks with millions of nodes.

Among the most recent literature is a technique called graph factorization. It finds the low-dimensional embedding of a large graph through matrix factorization, which is optimized using stochastic gradient descent. This is possible because a graph can be represented as an affinity matrix. However, the objective of matrix factorization is not designed for networks, therefore does not necessarily preserve the global network structure.

Intuitively, graph factorization expects nodes with higher first-order proximity are represented closely. Instead, the LINE model uses an objective that is particularly designed for networks, which preserves both the first-order and the second-order proximities. Practically, the graph factorization method only applies to undirected graphs while the proposed model is applicable for both undirected and directed graphs.

4 Model Description

We describe the LINE model to preserve the first-order proximity and second-order proximity separately, and then introduce a simple way to combine the two proximity.

4.1 LINE with First-order Proximity

The first-order proximity refers to the local pairwise proximity between the vertexes in the network. To model the first-order proximity, for each undirected edge (i, j), we define the joint probability between vertex v_i and v_j as follows:

$$p_1(v_i, v_j) = \frac{1}{1+\exp\left(-u_i^T . u_j\right)}$$

To preserve the first-order proximity, a straightforward way is to minimize the distance between two distributions. We choose to minimize the KL-divergence of two probability distributions

$$O_1 = -\sum_{(i,j \in E)} w_{ij} \log\left(p_1(v_i, v_j)\right)$$

4.2 LINE with Second-order Proximity:

The second-order proximity assumes that vertices sharing many connections to other vertices are similar to each other. In this case, each vertex is also treated as a specific "context" and vertices with similar distributions over the "contexts" are assumed to be similar. Therefore, each vertex plays two roles: the vertex itself and a specific "context" of other vertices. For each directed edge (i, j), we first define the probability of "context" v_j generated by vertex v_i as:

$$p_2(v_j|v_i) = \frac{\exp(u_j^T.u_i)}{\sum_{k=1}^{|V|}\exp(u_k^T.u_i)}$$

To preserve the second proximity, a straightforward way is to minimize the distance between two distributions. We choose to minimize the KL-divergence of two probability distributions

$$O_2 = -\sum_{(i,j\in E)} w_{ij}\log(p_2(v_i|v_j))$$

Optimizing the above objective is computationally expensive, which requires the summation over the entire set of vertices when calculating the conditional probability $p_2(|v_i)$. To address this problem, we adopt the approach of negative sampling proposed in , which samples multiple negative edges according to some noisy distribution for each edge i, j). More specifically, it specifies the following objective function for each edge (i, j):

$$\log\sigma(u_j^T.u_i) + \sum_{i=1}^{K} E_{v_n\ P_n(v)}[\log\sigma(-u_n'^T.u_i)]$$

Using the above function as objective function, we try to minimize is using gradient descent.

4.3 Combining first-order and second-order proximities

To embed the networks by preserving both the first-order and second-order proximity, a simple and effective way we find in practice is to train the LINE model which preserves the first-order proximity and second-order proximity separately and then concatenate the embeddings trained by the two methods for each vertex. However we have trained both the objective function together. Which makes our objective function as:

$$O_3 = -\sum_{(i,j\in E)}(w_{ij}\log(p(v_i,v_j)) + w_{ij}\log(p(v_i|v_j)))$$

5 Experiments

We empirically evaluated the effectiveness and efficiency of the LINE. We applied it over the social networks.

5.1 Data Set

We have used BlogCatalog dataset for our experiment. It is a network of social relationships provided by blogger authors. It has two variables: network and groups. "network" is a symmetric sparse matrix

representing the interaction between users, and "groups" are the groups subscribed by users. We use that as the class labels in our work. The label represent topic categories provided by users.

Entity	**Count**
Nodes	10,312
Links	333,983
Categories	39

Table 1: BlogCatalog description

5.2 Parameter Settings

Below are the parameter values used in our experiments.

Parameter	**Value**
mini-batch size (b)	1
Learning rate (η)	0.025
Negative edge sample size ($bsample$)	5
Embedding dimensionality (d)	100

Table 2: Parameter values as used in experiments

5.3 Results

For our model we used a one-vs-rest logistic regression from scikit library for classification. In this experiment we used training ratios as 10%, 50% and 90%. We did experiments for LIne 1 and Line 2 (separate training) and our version of Line (joint training). For each model experiments were repeated three times. Mini F measure were calculated during these experiments. The Following table summarizes the results obtained.

iteration	**10%**	**50%**	**90%**
1	38	42.4	45.6
2	36	40.3	43.2
3	36.8	43.1	43.7
Mean	36.93	41.93	44.17

Table 3: Results from LINE1 and LINE2 (separately trained and concatinated) model

Iteration	10%	50%	90%
1	39.1	43.3	46.5
2	38.8	43.1	46.4
3	37.7	42.9	47.3
Mean	38.53	43.1	46.73

Table 4: Results from LINE (Jointly trained) model

It has been observed that joint training of objective function has resulted in slight increased in classification performance.

5.4 Conclusion

We implemented a novel network embedding model called the LINE, and also implemented joint training for it. This algorithm is easily scales up to networks with millions of vertexes and billions of edges. It has carefully designed objective function that preserves both the first-order and second-order proximities, which are complementary to each other. Experiment results on BlogCatalog data proves the efficiency of this method.

6 Applications

Better representation of nodes helps in solving various network mining tasks by conventional machine learning algorithms. It can be used for:

1. Node classification
2. Link prediction
3. Node visualization
4. Node recommendation

7 Challenges

Some challenges we will be facing would be:

1. Scalability issues while handling large graphs.
2. learning various deep learning models and tools in short time

8 References

1. DeepWalk: Online Learning of Social Representations
2. LINE: Large-scale Information Network Embedding
3. Understanding basics of Neural Network

4. Understanding the need of deep learning

5. Datasets: BlogCatalog Data, Flickr Data, Youtube Data