✨ Member-only story

Random Walk in Node Embeddings (DeepWalk, node2vec, LINE, and GraphSAGE)

Graph Embeddings

Edward Ma · Follow

Published in Towards AI · 6 min read · Dec 24, 2019


和興無線電行
HUAWEI
紅蘋果

Photo by Steven Wei on Unsplash

> *Instead of using traditional machine learning classification tasks, we can consider using graph neural network (GNN) to perform node classification problems. By providing an explicit link of nodes, this classification problem is no longer classified as an independent problem but leveraging graph structures such as the degree of nodes. The usefulness of graph properties assumes that individual nodes are correlated with other similar nodes.*
>
> *Typically example is a social media network. Imagine how Facebook connects you and somebody else based on what post you like, where you check-in etc. A graph is capable to represent this kind of relationship and we can leverage it to train GNN. Detail use cases of GNN will be covered in later stories.*

We went through a knowledge graph embeddings in the last story. It covers a framework to train graph embeddings via TransE, and ComplEx on a large-scale. In this story, we would like to talk about graph structure and random walk-based models for learning graph embeddings. The following sections

cover DeepWalk (Perozzi et al., 2014), node2vec (Grover and Leskovec, 2016), LINE (Tang et al., 2015) and GraphSAGE (Hamilton et al., 2018).

Graph Structure

Before ego through those random walk-based model, we need to understand some basic graph structures.

- First-order Proximity: As known as Homophily. If two nodes are connected, they are homophily or having first-order proximity.
- Second-order Proximity: If two nodes share many connections, they have higher second-order proximity. i.e. node 5 and node 6 in the below figure 1. Although node 5 and node 6 are not connected directly, they share exactly the same neighbor nodes. It indicates they have a degree of similarity. On the other hand, it helps to convergence those fewer neighbor nodes.
- Structural Equivalence: It refers to similar structural roles. For example, two nodes are structural equivalence if two nodes are connected to three different nodes. i.e. node u and node s6 in the below figure 2.

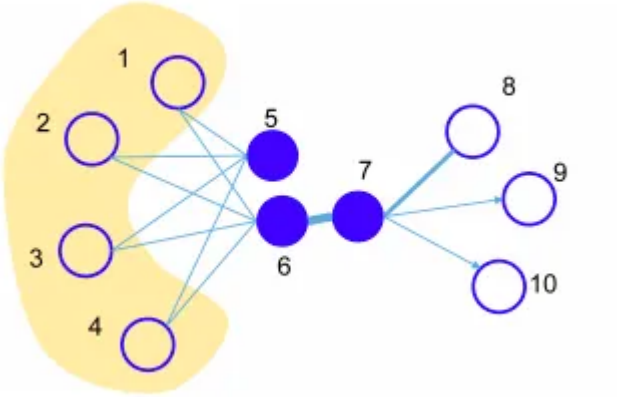


Figure 1 (Tang et al., 2015)

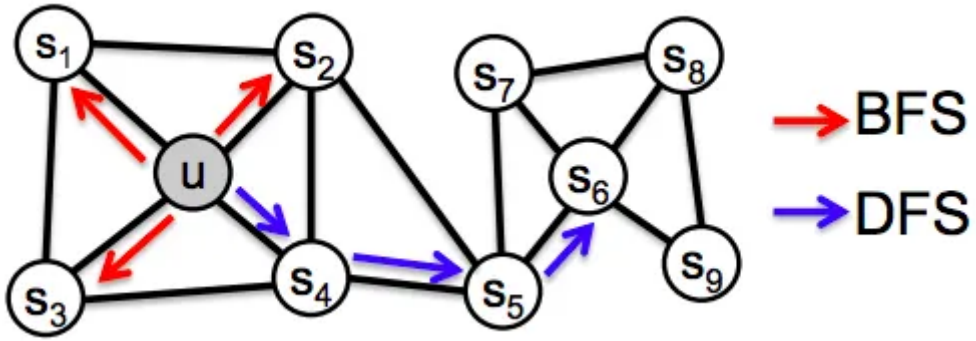


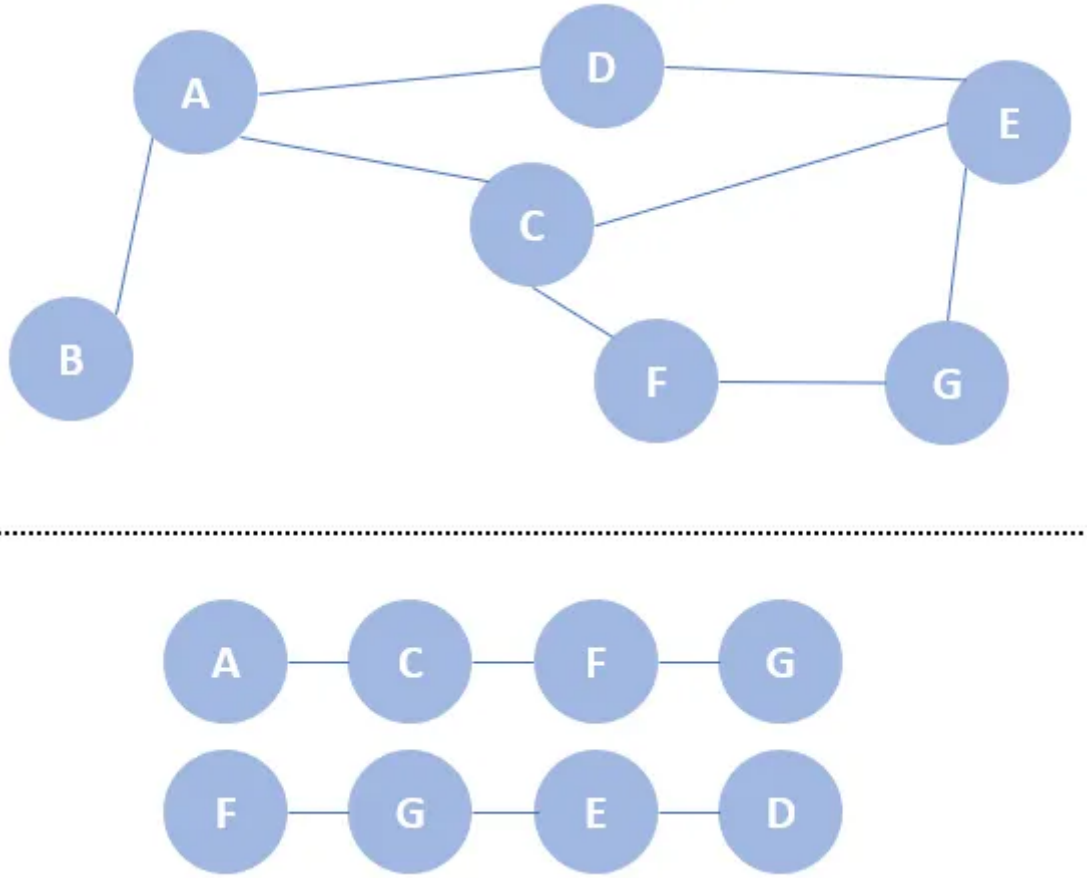
Figure 2 (Grover and Leskovec, 2016)

DeekWalk

DeepWalk (Perozzi et al., 2014) is introduced to learn node embeddings via a random walk and word2vec (Mikolo et al., 2013) word2vec algorithm. In natural language processing (NLP), we can apply the skip-gram model and feeding a sentence (a series of words) to train word embeddings. In short,

the training objective is using the center word to predict surrounding words. You may check this story for detail.

Go back to DeepWalk, it picks a node randomly and “walk” to a neighbor node randomly until it reaches maximum length (or some random length). Given the following knowledge graph (upper part), we pick “A” and “F” as starting points respectively. Later on, we move the node “A” to node “C” and so on.



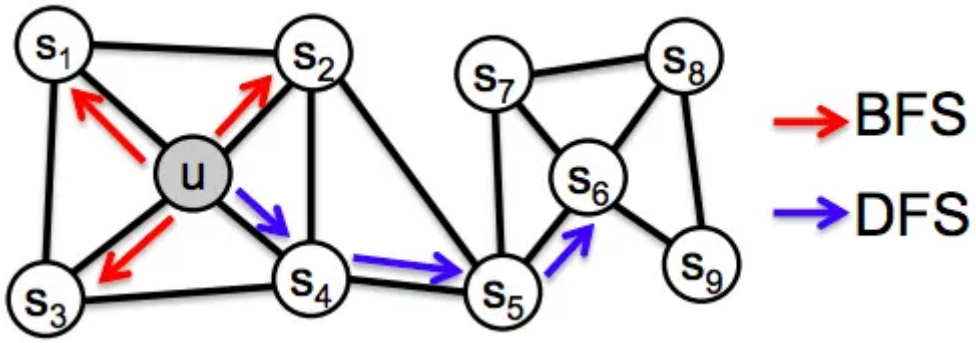
Above: Knowledge Graph. Below: Random Walk Sequence

In this case, we treat node and series of nodes as “word” and “sentence” respectively. Later on, we can reuse the skip-gram algorithm to convergence node embeddings.

node2vec

node2vec (Grover and Leskovec, 2015) is an advanced version of DeepWalk (Perozzi et al., 2014). One of the limitations of DeepWalk (Perozzi et al., 2014) is that you cannot control the path. node2vec (Grover and Leskovec, 2015) also use random behavior but having weight on it.

Instead of "walk" randomly, authors introduce Breadth-Fast-Sampling (BFS) and Depth-First-Sampling (DFS) to control the random behavior. BFS reaches immediate neighbors while DFS prefer node away from the source.



BFS: Breadth-First-Sampling. DFS: Depth-First-Sampling (Grover and Leskovec, 2016)

To provide this flexibility, random walk probability is no longer unweighed while it can achieve by the following setup. dtx means the distance from the source while p and q are parameters. A larger p ensures that it is less chance to sample a visited node. q allows us to control whether we want BFS or DFS. A larger q tends to make the random walk around the center.

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases}$$

LINE

LINE (Tang et al., 2015), aka Line Large-scale Information Network Embedding, proposes to use first-order proximity and second-order proximity to learn node embeddings.

To minimize the first-order proximity objective, connected nodes will convergence to each other in embeddings space. Note that it is only applicable for an undirected graph.

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

First-order proximity objective (Tang et a., 2015)

Same as above, we need to minimize the second-order proximity objective to convergence nodes who share similar neighbor nodes.

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

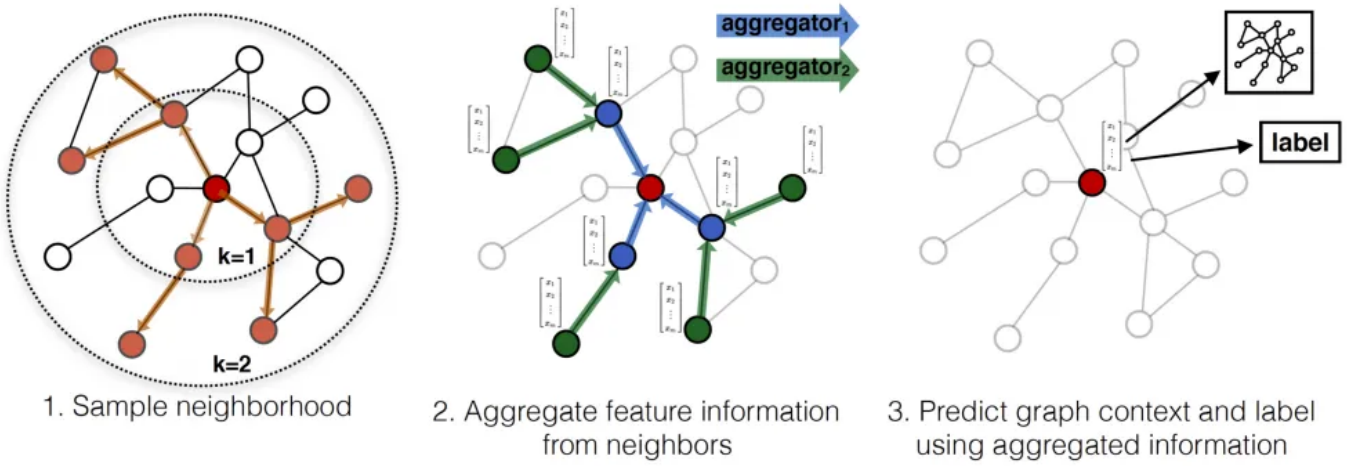
Second-order proximity objective (Tang et al., 2015)

GraphSAGE

GraphSAGE (Hamilton et al., 2018), aka Graph SAmple and aggreGatE, .is a model that generates node embeddings on the fly. Unlike other models, it does not train specific node embeddings but training an aggregator. Due to this nature, it can handle unseen nodes.

This approach including two major steps which are sampling neighbor nodes and aggregating it. The following figure illustrates the step:

1. Pick one node (i.e. center of this graph). Pick some neighbor nodes in each layer (i.e. k=1, k=2).
2. Aggregate nodes in each layer (blue layer and green layer)
3. Feed embeddings to the neural network and predict it



GraphSAGE sample and aggregate approach (Hamilton et al., 2018)

As mentioned before, it only samples some neighbor nodes for training in every mini-batch. The random walk approach is used and it is unweighted (same as DeepWalk).

But how do we aggregate it? Authors proposed 4 ways which are mean aggregator, LSTM aggregator, and pooling aggregator.

- Mean Aggregator: Taking the average of neighbor and concatenating with the target node.
- GCN Aggregator: Taking average after concatenating neighbor and nodes. The sequence is reversed when compared with a mean aggregator. The authors demonstrated it achieves a better result.

- LSTM Aggregator: Random arrange neighbor orders and feeding into the LSTM layer.
- Pooling Aggregator: Feeding to multi-layer perceptron (MLP) and performing the max-pooling.

Take Away

- The idea of DeepWalk is easy while it comes with some limitations. Since it is a purely random walk (unweighted), frequency or importance cannot influence embeddings and account for graph structure. Since it needs to "walk" to gain information from a neighbor, it is hard to support unseen data.
- node2vec address some of the limitations of DeepWalk such as unweighted problem. However, it can suffer from a higher computation requirement.
- DeepWalk and LINE use a depth-first search (DFS) strategy for all nodes and a breadth-first search (BFS) strategy on low-degree nodes respectively. On the other, node2vec mix both BFS and DFS to learn node embeddings.

Extension Reading

- DeepWalk repository
- node2vec repository
- GraphSAGE repository

About Me

I am a Data Scientist in the Bay Area. Focusing on the state-of-the-art in Data Science, Artificial Intelligence, especially in NLP and platform related. Feel free to connect with me on LinkedIn or follow me on Medium or Github.

Reference

- B. Perozzi, R. A. Rfou, and S. Skiena. DeepWalk: Online Learning of Social Representations. 2014
- J. Tang, M .Qu, M. Wang, M. Zhang, J. Yan and Q. Mei. LINE: Large-scale Information Network Embedding. 2015
- A. Grover and J. Leskovec. node2vec: Scalable Feature Learning for Networks. 2016
- W. L. Hamilton, R. Ying and J. Leskovec. Inductive Representation Learning on Large Graphs. 2018

Machine Learning | Data Science | Artificial Intelligence | Graph Neural Networks | Python

Written by Edward Ma

Follow

3.3K Followers · Writer for Towards AI

Focus in Natural Language Processing, Data Science Platform Architecture.
https://makcedward.github.io/

More from Edward Ma and Towards AI

Edward Ma in Towards Data Science

Data Augmentation in NLP

Introduction to Text Augmentation

5 min read · Apr 12, 2019

500 2

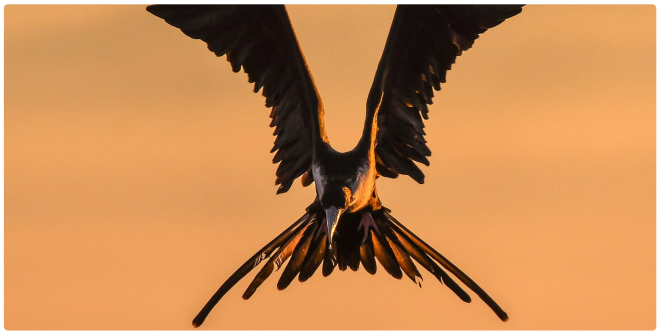
Youssef Hosni in Towards AI

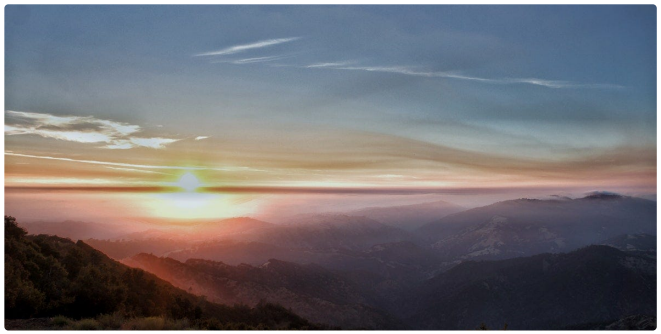
Mastering These 5 Statistics Concepts Will Boost Your Success...

Data science is a rapidly growing field that involves the extraction of valuable insights...

· 16 min read · May 9

567 2





Dr. Mandar Karhade, MD. PhD. in Towards AI

Edward Ma

Data Augmentation for Audio

Falcon-40B: A Fully OpenSourced Foundation LLM

Each Contributor hereby grants Grants to You a perpetual, worldwide, non-exclusive,...

✦ · 7 min read · May 28

431 1

Data Augmentation

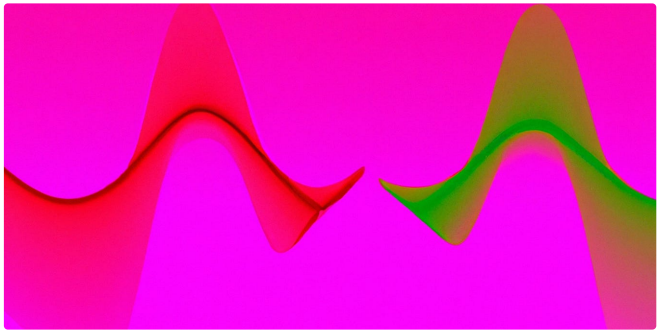
3 min read · Jun 1, 2019

227 3

See all from Edward Ma

See all from Towards AI

Recommended from Medium



Aparna Dhinakaran in Towards Data Science

Understanding KL Divergence

A guide to the math, intuition, and practical use of KL divergence—including how it is...

7 min read · Feb 2

42

AI TutorMas... in Artificial Intelligence in Plain Eng...

Graph Neural Networks—Introduction for Beginners

A Graph Neural Network (GNN) is a type of neural network that is designed to work with...

· 10 min read · Jan 14

423 1

Lists

What is ChatGPT?

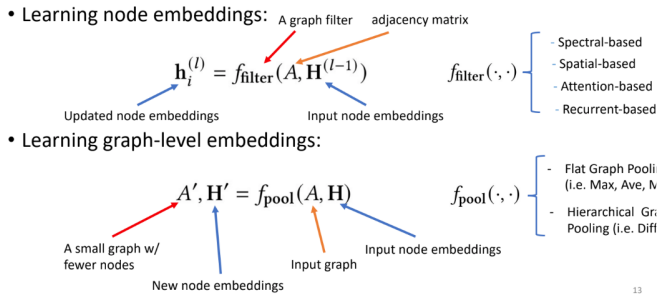
9 stories · 109 saves

Staff Picks

352 stories · 113 saves

Stories to Help You Level-Up at Work

19 stories · 103 saves



Jason Huang

Graph Encoder-Decoder Models for NLP

All of the contents in this article are excerpted from the tutorial video, slides, and website of...

5 min read · Dec 31, 2022

27 1

Tingsong Ou

PyTorch Geometric

Link Prediction on Heterogeneous Graphs with PyG

By Jan Eric Lenssen and Matthias Fey

10 min read · Dec 22, 2022

85 3

Rajanie Prab... in Stanford CS224W GraphML Tuto...

Variational AutoEncoder, and a bit KL Divergence, with PyTorch

I. Introduction

8 min read · Jan 1

74

Graph Analysis Made Easy with PyG Explainability

By: Anh Hoang Nguyen, Rajanie Prabha, Kevin Su as part of the Stanford CS224W course...

12 min read · May 14

9

See more recommendations