Benchmarking - Struc2vec

COL868 - Graph Neural Networks

Shivam Singla

Mrigank Raman

Introduction

- Various real life networkings problems can be formulated as standard problems in the graphs and thus we can use the properties of graphs to solve those problems.

- We discuss some of the problems like link prediction, pair-wise node classification, multi-class node classification, etc.

- We try to analyse the technique Struc2vec[1], which is an unsupervised technique for undirected and unlabeled graphs to generate embeddings for each node, which can be used to perform graph related tasks.

Unsupervised techniques on graphs

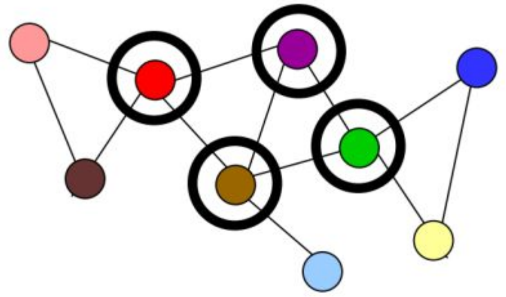
- In a way, graphs are unstructured themselves. Therefore, it is quite hard to do mainstream machine learning tasks on them.

- Recent approaches like node2vec[2], Deepwalk[3], struc2vec approach this problem by generating node embeddings for a given undirected and unlabeled graph.

- This gave a huge boost to new innovations in the area of Graph Neural Networks(GNNs).

Struc2vec model

- Structural identity of nodes
    - Identification of nodes based on network structure(no other attribute)
    - Often related to the role played by the node
- Automorphism: strong structural equivalence
    - Red, green: automorphism
    - Purple, brown : structural similarity



Source : class lecture slides

Step 1 : Structural similarity

- $R_k(u)$: the set of nodes at distance (hop count) exactly $k \geq 0$ from u in G.
- s(S) : ordered degree sequence of a set of nodes S.
- Hierarchical measure for structural similarity between two nodes used to generate a multi-layered graph, with each layer having all the nodes from G and edges between in the given layer denote the similarity between the two nodes at this particular level of hierarchy.

Step 1 : Structural similarity

$$f_k(u,v) = f_{k-1}(u,v) + g(s(R_k(u)), s(R_k(v))),$$
$$k \geq 0 \text{ and } |R_k(u)|, |R_k(v)| > 0$$

$g(D_1, D_2)$ here measures the distance between the ordered degree sequences D_1 and D_2, and f_{-1} here is initialised to be 0.

Dynamic Time warping : used to calculate $g(D_1, D_2)$

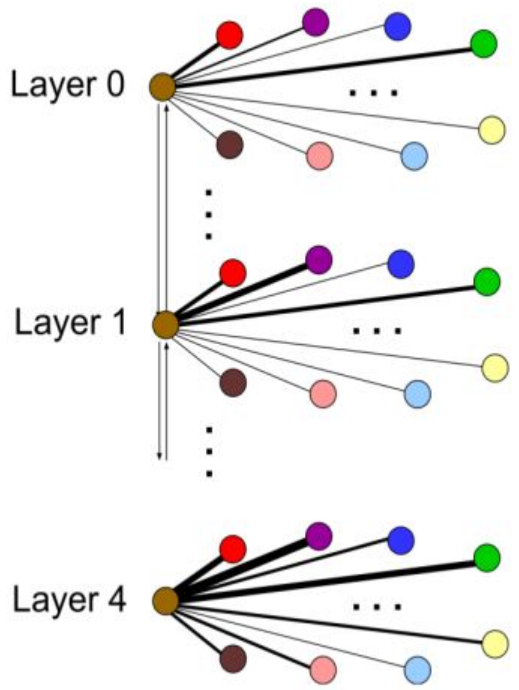
- Tries to find the optimal alignment between two given sequences
- Uses a distance function for pair-wise alignment : max(a,b)/min(a,b) -1

Step 2 : Multi-layered Graph

- Each layer is a weighted complete graph with edge weights measuring structural similarities between nodes at this level of hierarchy.

- Edge weights in layer k, $w_k(u,v) = \exp\{-f_k(u,v)\}$

- The edge weight between layer is proportional to the no. of similarly* structured nodes at that level.



Step 3: Context generation

- Context is generated by biased random walks
    - Walking on the multi-layered graph

- Probability of walking to a next node is proportional to the edge weight between them.

- Walk in the current layer with a fixed probability p.

Step 4: Learning embeddings

- Using the generated random walks for a given node, create a context set for it.

- Train a neural network to learn latent representations for node
    - Use skip-gram model(along with hierarchical softmax)

Optimizations

- *Reducing the length of degree sequences (OPT1)*
    - Reduce size by coding count of repeated node degrees instead
- *Reducing the number of pairwise similarity calculations (OPT2)*
    - Skipping calculations for node pairs with high difference in degrees
- *Reducing number of layers (OPT3)*

Important parameters

- Num-walks : the number of random walks per node to generate context. (Default is 10)

- Walk-length : the no. of nodes sampled per random walk. (Default is 80)

- No. of layers : the number of layers of the multi-layered graph used to make calculations. (Default is it calculates till the size of the diameter)

- Dimensions : Size of embeddings generated. (Default is 128)

- Window size : The context size for word2vec. (Default is 10)

Datasets

We perform our tasks majorly on the following three datasets :-

- Brightkite[4]
- Proteins[5]
- PPI[6]

Protein-Protein Interaction (PPI)

- The PPI dataset contains 24 different protein-protein interaction networks with nodes being different proteins and edges signifying their interaction among themselves.

- It contains total 56,944 nodes and 818,716 edges, with each vertex having an average degree of 28.8, along with a 50-dimensional feature vector. Nodes are divided into 121(multiclass) classes.

- We use this dataset for two tasks - Link prediction(Predicting whether two given nodes have a connecting edge) and Multi-class node classification (A supervised task to predict the class label of a node).

Experimental Setup

<u>Transductive learning</u> :

In the transductive learning setting, the model is trained and tested on a given graph with a fixed node ordering and has to be re-trained whenever the node ordering is changed or a new graph is given. We perform 5-fold cross validation and report the result with best validation accuracy.

Experiments : PPI

- We perform two experiments on PPI dataset :- *Link Prediction and Multi class node classification.*

- The hyperparameters for generating embeddings are as follows:-

No. of dimensions	128
Walk length	80
No. of walks per node	10
No. of layers of the graph	6
Window size	10

Experiments : PPI

- Multi class node classification

Model	**Precision**	**Recall**	**Micro-F1**
MLP	--	--	0.422
Node2vec[2]	0.419	0.609	0.479
DeepWalk[3]	0.363	0.594	0.431
Struc2vec[1]	0.380	0.551	0.428
GraphSage[7]	--	--	0.768
GAT[8]	--	--	**0.973**

Experiments : PPI

- Multi class node classification

Classification model	ROC AUC	Precision	Recall	Micro-F1
Logistic	0.587	0.38	0.551	0.428
Decision Trees	0.567	0.35	0.726	0.473
Extra Trees	0.589	0.363	0.774	0.494
Random Forest	0.588	0.362	0.778	0.494
MLP	0.567	0.35	0.726	0.473

Experiments : PPI

- Sampling Technique for Link Prediction:
  - For this task we first sample 40000 positive edges and 40000 negative edges and train our classifier using cross validation.
  - We then sample 10000 new positive edges not in the first set and 10000 negative sample not in the first set and use this set as our test set.
  - We shuffle the examples in both the sets.

Experiments : PPI

- Results using above sampling technique for link prediction:

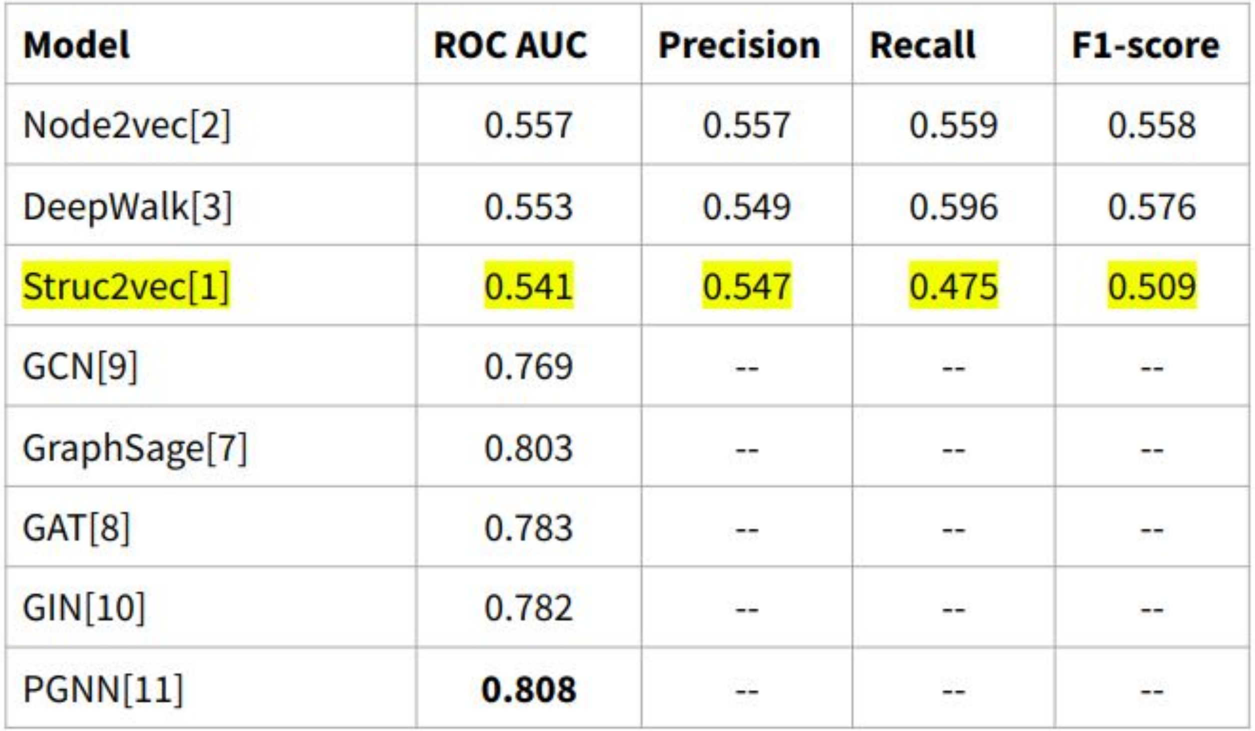
	ROC AUC	Precision	Recall	F1 Score
Random Forest	0.76	0.75	0.76	0.75

Experiments : PPI

- Sampling Technique for Link Prediction:
  - In this technique we use all the positive edges and sample around same number of negative edges and shuffle the set.
  - Then we train on 80% data and test on the remaining 20%.
  - This technique gives rise to a dataset approximately 16 times bigger than the other technique.
  - In the subsequent slides we provide results using this sampling technique.

Experiments : PPI

- Link Prediction



Model	ROC AUC	Precision	Recall	F1-score
Node2vec[2]	0.557	0.557	0.559	0.558
DeepWalk[3]	0.553	0.549	0.596	0.576
Struc2vec[1]	0.541	0.547	0.475	0.509
GCN[9]	0.769	--	--	--
GraphSage[7]	0.803	--	--	--
GAT[8]	0.783	--	--	--
GIN[10]	0.782	--	--	--
PGNN[11]	**0.808**	--	--	--

Experiments : PPI

- Link Prediction

Classification model	**ROC AUC**	**Precision**	**Recall**	**Micro-F1**
Logistic	0.541	0.547	0.475	0.509
Decision Trees	0.535	0.535	0.538	0.536
SVM	0.573	0.601	0.518	0.491
Random Forest	0.609	0.623	0.554	0.586
MLP	**0.633**	**0.639**	**0.613**	**0.625**

Experiments : PPI

- Link Prediction

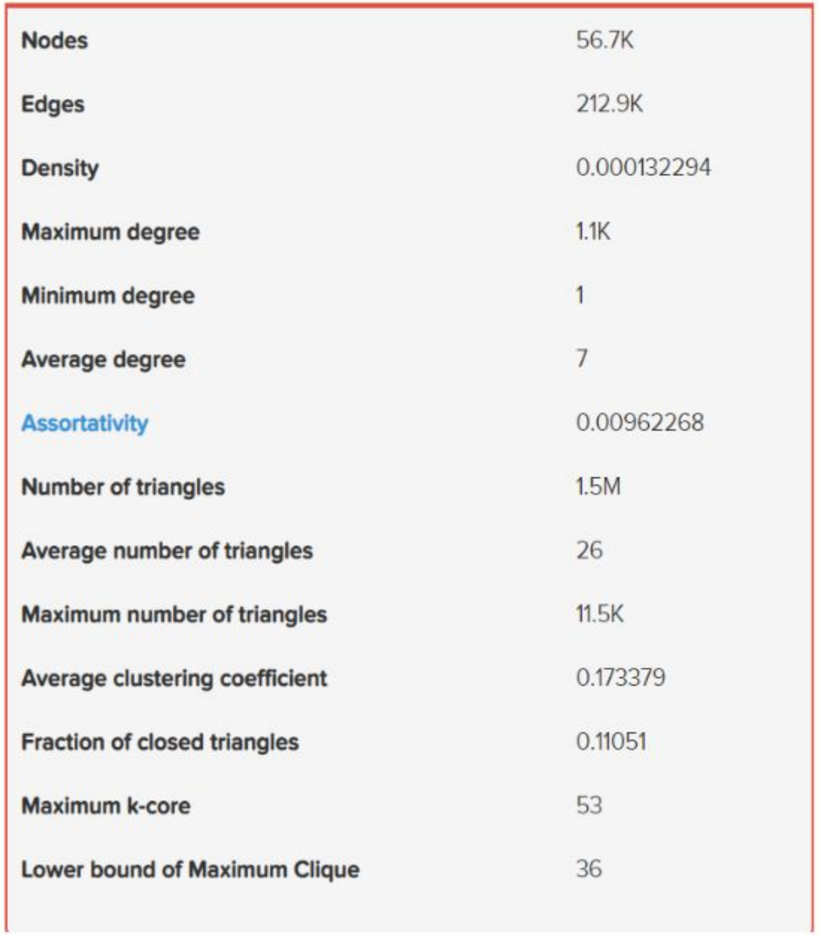
Aggregator function	**ROC AUC**	**Precision**	**Recall**	**Micro-F1**
Average	0.541	0.547	0.475	0.509
Dot Product	0.533	0.548	0.376	0.446
L1- distance	0.536	0.538	0.504	0.521
L2 - distance	0.529	0.541	0.384	0.449
Concatenation	**0.568**	**0.572**	**0.543**	**0.557**

Brightkite

- Brightkite [2] was once a location-based social networking service provider where users shared their locations by checking-in.

- Originally this was a directed network but the dataset we use is an undirected version where friendship happens both ways.

- The nodes in this graph are unlabelled but have a feature vector which will not be used by us as Struc2Vec only uses structural properties of a graph.



Nodes	56.7K
Edges	212.9K
Density	0.000132294
Maximum degree	1.1K
Minimum degree	1
Average degree	7
Assortativity	0.00962268
Number of triangles	1.5M
Average number of triangles	26
Maximum number of triangles	11.5K
Average clustering coefficient	0.173379
Fraction of closed triangles	0.11051
Maximum k-core	53
Lower bound of Maximum Clique	36

Experimental Setup

- We perform the link prediction task on the brightkite dataset.
- Since it takes around 9 hrs to train the model once it was infeasible to vary the hyperparameters.
- The hyperparameters that we select are:

No. of dimensions	128
Walk length	60
No. of walks per node	10
No. of layers of the graph	3
Window size	5

Experimental Setup

- Sampling Technique:
  - For this task we first sample 120000 positive edges and 120000 negative edges and train our classifier using cross validation.
  - We then sample 30000 new positive edges not in the first set and 30000 negative sample not in the first set and use this set as our test set.
  - We shuffle the examples in both the sets.

Experiments : Brightkite

- Link Prediction

	ROC AUC	Precision	Recall	F1 Score
Logistic	0.791	0.791	0.791	0.791
MLP	**0.821**	**0.821**	**0.821**	**0.821**
Random Forest	0.817	0.817	0.817	0.817

Modifications in Similarity Functions

- Betweenness Centrality :
  - Use BFS to find all the shortest paths
  - Very slight increase in link prediction ROC AUC
  - Takes more time
- Clustering Coefficient:
  - Sample a subgraph and apply naive algorithm
  - Very small increase in performance with huge increase in time taken

Results of Modification

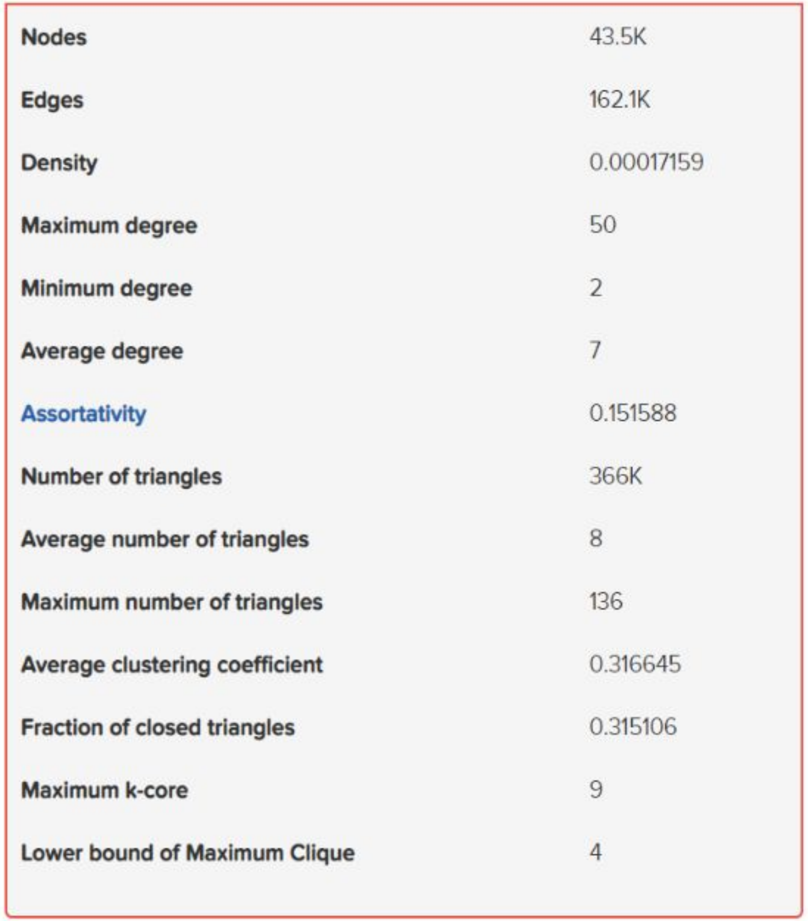
- We report results on Brightkite Link Prediction using an MLP and same sampling technique as used above in Brightkite.

	ROC AUC	Precision	Recall	F1 Score
Betweenness	0.824	0.824	0.824	0.824
Clustering	0.827	0.827	0.827	0.827

- Thus due to scalability issues and no marked increase in performance we submit code with degree similarity function.

Proteins

- This dataset contains the structure of different proteins in the form of a graph with each node representing an atom and links representing bonds between atoms.
- This graph is a labelled and undirected graph and as usual in Struc2Vec we don't use any node features.
- This dataset is used for pairwise node classification task in our benchmarking exercise.



Nodes	43.5K
Edges	162.1K
Density	0.00017159
Maximum degree	50
Minimum degree	2
Average degree	7
Assortativity	0.151588
Number of triangles	366K
Average number of triangles	8
Maximum number of triangles	136
Average clustering coefficient	0.316645
Fraction of closed triangles	0.315106
Maximum k-core	9
Lower bound of Maximum Clique	4

Experimental Setup

- We perform the pairwise node classification task on the proteins dataset.
- The above modifications took so much time that we did not have the time to try new hyperparameters.
- The hyperparameters that we choose are:

No. of dimensions	128
Walk length	60
No. of walks per node	10
No. of layers of the graph	3
Window size	5

Experimental Setup

- Sampling Technique:
  - For this task we randomly select 300000 node pairs and check the number of node pairs having same label.
  - If this number is around 130000 to 170000 we use this as our dataset or otherwise sample again. Everytime this condition was achieved.
  - We again trained on 80% of the dataset and tested on 20% on it and we shuffled the dataset as well.

Experiments : Proteins

- Pairwise Node Classification

	ROC AUC	Precision	Recall	F1 Score(Macro)
Random Forest	**0.519**	**0.520**	**0.530**	**0.510**
MLP	0.504	0.510	0.510	0.510

- We also used the above modifications to similarity functions but there was no increase in performance.

Intuition of the result

- As we can see in the image that Struc2Vec is unable to cluster the nodes on basis of labels as it does not use any label information.
- Thus as expected Struc2Vec performs badly in tasks concerning labels.
- Struc2Vec learns the structural properties of graphs which is important for link prediction thus it performs very good in this task.

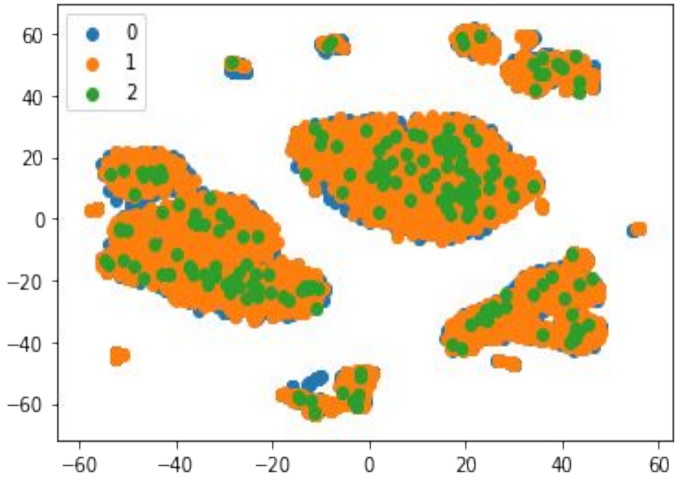


Fig: TSNE on Protein Nodes

References

[1] Ribeiro, Leonardo FR, Pedro HP Saverese, and Daniel R. Figueiredo. ”struc2vec: Learning node representations from structural identity.” Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2017

[2] Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016.

[3] Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014.

References

[4] E. Cho, S. A. Myers, J. Leskovec. Friendship and Mobility: Friendship and Mobility: User Movement in Location-Based Social Networks ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2011.

[5] K. M. Borgwardt, C. S. Ong, S. Schoenauer, S. V. N. Vishwanathan, A. J. Smola, and H. P. Kriegel. Protein function prediction via graph kernels. Bioinformatics, 21(Suppl 1):i47–i56, Jun 2005.

[6] Zitnik, M. and Leskovec, J. Predicting multicellular function through multi-layer tissue networks. Bioinformatics, 33(14):i190–i198, 2017

[7] Hamilton, Will, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs." Advances in neural information processing systems. 2017.

References

[8] Veličković, Petar, et al. "Graph attention networks." arXiv preprint arXiv:1710.10903 (2017).

[9] Kipf, Thomas N., and Max Welling. "Semi-supervised classification with graph convolutional networks." arXiv preprint arXiv:1609.02907 (2016).

[10] Wang, Hongwei, et al. "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems." Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2019.

[11] You, Jiaxuan, Rex Ying, and Jure Leskovec. "Position-aware graph neural networks." arXiv preprint arXiv:1906.04817 (2019).

Thank You