Assignment 3 **Akash Anil**

Introduction

Network embedding is an approach to generate low dimensional network features corresponding to nodes (node embedding), edges (edge embedding), sub-network (sub-graph embedding), etc. [1]. Recently network embedding using neural networks has been reported to be more scalable than the state-of-art embedding models exploiting matrix factorization (e.g. PCA, SVD).

Majority of the recently proposed network embedding methods [2, 4] exploit the skip-gram model (first used in word embedding, i.e. `word2vec` [3]) which is trained over a sentence corpus for a specified context window w capturing syntactic and semantic relationships between the input word and near-by words which are most likely to appear together. However, unlike word embedding, for network embedding, there is no corpus available beforehand. Therefore, one needs to generate a corpus for the given network which can capture structural relationships among nodes such that two near-by nodes are more similar than other farther nodes. To generate network corpus, a majority of the embedding methods use node sampling approach (e.g. BFS, DFS, Random walks, etc.) which is capable of sampling a small number of nodes representing structural similarity between the target node and other sampled nodes. However, different sampling approaches capture different characteristics of the network which may affect the performance of node embedding. Moreover, a single sampling approach may behave differently for different networks. Hence, it is hard to judge the best sampling approach [2].

Problem Statement

1. Study and understand three state-of-art node sampling techniques and prepare a summary report[1].

2. Select an unsupervised network embedding model exploiting node sampling [2] and generate node embedding for the above sampling approaches over the network explored by you in Assignment 2. You can use the available code for the chosen network embedding model. However, you need to modify and comment the codes for incorporating different node sampling approaches. Evaluate the performance of different node embeddings on link prediction task by training a Logistic Regression Classifier on the test edges used in Assignment 2 for evaluating the performance of the classifiers. Use AUC, F_1-Measure (Micro and Macro), and Accuracy as the evaluation metric.

3. Report a critical analysis of the performance of link prediction by embedding models utilizing different sampling approaches over all the networks used in Assignment 2.

4. Compare the link prediction performance achieved in Assignment 2 against Assignment 3. Report a critical analysis of any improvements/degradation.

[1] Get a list from Table 6 in column 2 of https://arxiv.org/pdf/1709.07604.pdf
[2] Get a list from Table 6 in https://arxiv.org/pdf/1709.07604.pdf

5. Can you find limitations of the node sampling techniques you have chosen and suggest possible improvements?

References

[1] Hongyun Cai, Vincent W Zheng, and Kevin Chang. "A comprehensive survey of graph embedding: problems, techniques and applications". In: *IEEE Transactions on Knowledge and Data Engineering* (2018).

[2] Aditya Grover and Jure Leskovec. "node2vec: Scalable feature learning for networks". In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2016, pp. 855–864.

[3] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[4] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 701–710.

Assignment 3 Problem

Loitongbam Gyanendro Singh

September 24, 2018

Exploration of Multi-View Network Representation Learning

1 Introduction

Recently, there is a growing interest in representing networks into low-dimensional space (a.k.a, network embedding), where each node is represented with a low-dimensional vector[6, 7, 3]. Such vector representations are able to preserve the proximities between nodes, which can be treated as features and benefit a variety of downstream applications, such as node classification, link prediction and node visualization. Most of the existing works consider homogeneous network with single type of node and single type of relation between nodes in a network. However real world information networks are often heterogeneous in nature, i.e. multiple types of nodes and relations may exist. Take the network between users in social media sites (e.g, Twitter), where multiple types of relations such as following-followee, reply, retweet, mention, etc. Each relation defines a view of a network, and multiple relations yield a network with multiple views. Each individual view is usually sparse and biased, and thus the node representations learned by existing approaches may not be so comprehensive. To learn more robust node representations, a natural solution could be leveraging the information from multiple views. In literature, various methods have been proposed for learning data representations from multiple views, such as multi-view clustering methods[1, 4, 8] and multi-view matrix factorization methods[2, 5].

2 Problem statement

We are interested to learn robust node representations by considering multiple views of a network. Perform node embedding on a given graph $\mathbf{G} = (\mathrm{V},\mathrm{E}_1,\mathrm{E}_2,...,\mathrm{E}_K)$, where V is the set of vertices in G and E_k s.t. $1 \leq \mathrm{k} \leq \mathrm{K}$ are the set of edges on a particular view in G. A view k can be defined as the type of relation from a node, v_i to another node, v_j in V. The required dataset for performing node embedding will be uploaded in Canvas. Some of the popularly used node embedding techniques are listed in `https://github.com/chihming/awesome-network-embedding`.

2.1 Tasks

1. Perform node embedding for each view i.e. K vectors for each nodes, using atleast one of the following techniques:

- Matrix factorization technique[1] of your choice
- Neural network based technique[2] of your choice

2. Perform node embedding by collaborating all the views i.e. a single vector for each nodes, using atleast one of the following techniques:
   - Multi-view clustering techniques[3]
   - Multi-view matrix factorization techniques[4]
   - Neural network based techniques for multi-view representation[5].
3. Apply the embedding obtained above for the Link Prediction task.

2.2 Results

We expect you to show the performance comparison of node representation using average of the K views and the robust representation using multi-view techniques using AUC score, Precision, Recall and F-Score.

References

[1] Kamalika Chaudhuri et al. "Multi-view clustering via canonical correlation analysis". In: *Proceedings of the 26th annual international conference on machine learning*. ACM. 2009, pp. 129–136.

[2] Derek Greene and Pádraig Cunningham. "A matrix factorization approach for integrating multiple data views". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2009, pp. 423–438.

[3] Aditya Grover and Jure Leskovec. "node2vec: Scalable feature learning for networks". In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2016, pp. 855–864.

[4] Abhishek Kumar, Piyush Rai, and Hal Daume. "Co-regularized multi-view spectral clustering". In: *Advances in neural information processing systems*. 2011, pp. 1413–1421.

[5] Jialu Liu et al. "Multi-view clustering via joint nonnegative matrix factorization". In: *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM. 2013, pp. 252–260.

[6] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 701–710.

[1] Hints: Use the important nodes as features
[2] https://github.com/aditya-grover/node2vec
[3] See https://github.com/amaunz/mvc for reference
[4] See https://github.com/yifeng-li/mvmf for reference
[5] See https://github.com/mnqu/MVE for reference

[7] Jian Tang et al. “Line: Large-scale information network embedding”. In: *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee. 2015, pp. 1067–1077.

[8] Tian Xia et al. “Multiview spectral embedding”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 40.6 (2010), pp. 1438–1446.

Assignment 3 Problem

Anasua Mitra

September 23, 2018

Exploration of Random-Walk based Network Representation Learning methods for graph clustering : DeepWalk[3] and Node2Vec[1] are two state-of-the-art methods for learning latent representation of vertices of a given network targeted at tasks like link prediction, vertex classification etc. The core idea is to employ popular SkipGram[2] algorithm from NLP domain on network data. We are interested in learning representations of graph data that help us in clustering task, i.e., learning clusterable representation of graph data. Although Node2Vec claims that the algorithm is able to capture the structural properties, i.e., communities, roles in underlying network, but it does not have any explicit criteria in its objective function to do so. *We are interested in a critical analysis of these two methods for node clustering task.*

A) Study & understand the official code-base of DeepWalk and Node2Vec on a given input graph. Upon getting learned vertex embeddings, run naive K-means algorithm for a varied range of K (the number of communities) using three different initialization techniques (random, K-means++, PCA based initialization) to see how they perform on clustering task. We will get non-overlapping clusters here. Use Purity, NMI (Normalized Mutual Information), Modularity metric to evaluate the same. Repeat the clustering task for five different runs and report the mean of respective metrics. Use a representative multi-class dataset (only one label/ type will be associated with each node) [1] For the sake of fair comparison, use same set of / best reported hyperparams for these two methods.

B) Run Fuzzy C-means clustering algorithm in a similar manner and report Overlapping NMI, Omega Index metrics to evaluate the performance of these two algorithms on forming overlapping clusters. Use a representative multi-label dataset (multiple labels/ types can be associated with each node) [2] Do you see any difference in the performance of DeepWalk and Node2Vec? Do evidence-based subjective analysis.

C) Generate t-SNE plots of the learned embeddings (Tips: while running t-SNE algortihm make sure you run for a varied range of perplexity values and iterations to get the best set of t-SNE plots, understand the role of perplexity to get better embedding visualization in t-SNE plots). How do you compare the quality of generated t-SNE plots, which one is better?

D) One important & interesting question to ask is - how do we get the best number of clusters for a given data? There are three popular methods - Elbow criterion, Gap statistics and Silhouette method to do so. Understand

[1] you are encouraged to use Cora dataset `https://relational.fit.cvut.cz/dataset/CORA`.

[2] you are encouraged to use PPI dataset `https://snap.stanford.edu/node2vec/Homo_sapiens.mat`.

and use these three methods to come up with the best possible cluster K for the suggested two networked data.

References

[1] Aditya Grover and Jure Leskovec. "node2vec: Scalable feature learning for networks". In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2016, pp. 855–864.

[2] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[3] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 701–710.

Assignment 3 Problem

Neelakshi

September 23, 2018

Application of PTE (Predictive Text Embedding through Large-scale Heterogeneous Text Networks) to Sentiment Analysis :
Most unsupervised word embedding models like Skip-gram [1] model the syntactic context of words. While such word embeddings have been found effective in different text analysis problems, they exhibit certain limitations when applied to specific tasks like Sentiment Analysis. In Sentiment Analysis, the objective is to determine the polarity (positive, negative, neutral etc.) of a given piece of text (word, sentence, document etc.). Unsupervised word embeddings map words with opposite sentiment polarity (e.g. good and bad) to neighbouring word vectors due to their semantic similarity. PTE [2] on the otherhand, makes use of supervised information to obtain task specific word embeddings. The objective of this assignment is to explore the application of PTE for sentiment analysis task and compare it with the unsupervised skip-gram based embedding methods (i) node2vec and (ii) DeepWalk.

At the end of the assignment, the following is expected:

- Explore and understand PTE [1].
- Study applicability of PTE for Sentiment Analysis.
  - You will be provided with a Twitter dataset [2].
  - Divide the annotated data into training set and test set.
  - Use the training set to provide label information to PTE. Experiment using different fractions of the training set to provide label information.
  - Using these word embeddings as features, use different classification frameworks like SVM, CNN for tweet sentiment classification.
  - Repeat the classification task with embeddings obtained using unsupervised word embedding methods
  - Compare the results obtained using PTE and that obtained using unsupervised word embeddings using Precision, Recall and F-measure. What difference do you see in the results obtained. Give some insights.
  - Also report a subjective evaluation of the word embeddings obtained:e.g.
    - Are semantically similar words closer in the embedded space.

[1] https://github.com/mnqu/PTE
[2] The dataset will be uploaded in Canvas

    - Are words with same polarity closer to each other than those with opposite polarity
    - Compare the word embeddings obtained using PTE and other unsuprvised methods over tasks like word similarity, word analogy, word clustering etc.

- Summarize your findings and observations. What are other methods that can be used for task specific embeddings from text? Do you have any suggestions or proposals?

References

[1] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[2] Jian Tang, Meng Qu, and Qiaozhu Mei. "Pte: Predictive text embedding through large-scale heterogeneous text networks". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2015, pp. 1165–1174.