Computationally efficient protein functionality prediction via node embeddings

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this paper we present a novel and computationally efficient method for predicting protein functionality in large-scale Protein-Protein Interaction (PPI) networks. The main idea is to exploit recent advances in graph embedding by means of random walks, specifically non-backtracking random walks, on the nodes of the graph followed by using the sample paths as inputs to a popular word embedding method namely word2vec. For a yeast PPI network, it is shown that our method is competitive in predicting protein functionality with the current state-of-art method while offering significant advantages in computational speed. This preliminary results open up new ways to fill in missing protein functionality by way of edge prediction via embeddings.

1 Introduction

Vector space embeddings of the nodes of a graph play an important role in dimensionality reduced representation of the data. Several methods based on stochastic flows, primarily random walks on the graphs, have been used for finding good embeddings [Belkin and Niyogi, 2001, Lawrence, 2012, Talmon et al., 2013, Singer and Wu, 2012]. However these approaches do not scale well with the number of nodes and with the number of edges in the graph due to severe computational requirements of computing a spectral factorization of the adjacency matrix or of the graph Laplacian. Motivated by advances in neural word embeddings for natural language data [Levy and Goldberg, 2014, Mikolov et al., 2013], several authors have proposed to use sample paths from stochastic processes on the nodes of the graph as sentences and subsequently using the word embedding algorithm for embedding the nodes [Perozzi et al., 2014, Grover and Leskovec, 2016, Hashimoto et al., 2015, Ding et al., 2016].

The choice of the stochastic process used to generate the sample path turns out to be a critical choice that dictates the quality of the embedding while also affecting the computational speed. In this work we employ a variation of random walks on graphs, namely the non-backtracking (NBT) random walks, to generate sample paths from the given graph. NBT walks mix faster compared to backtracking (BT) random walks [Alon et al., 2007] and one can reduce the length of the sample paths. Further, the NBT operator has better spectral properties for sparser graphs compared to the BT operator [Krzakala et al., 2013] and this is reflected in superior performance on clustering under the Stochastic Block Model (SBM) as shown in our numerical results section.

Besides showing the improvements on clustering under SBM, we show that this approach is very effective in predicting the protein functionality from incomplete protein-protein interaction networks. In particular we show that this approach has significant computational advantage over the recently proposed prediction method based on Diffusion State Distances (DSD) [Cao et al., 2013] computed from random walks over graphs.

2 Notation and background

Let $G = (V, E)$ be a graph with vertex set V and edge set E, and $|V| = n, |E| = m$. The *adjacency matrix* A of G is defined as the $n \times n$ matrix with $a_{u,v} = 1$ if and only if $(u, v) \in E$, and the *degree matrix* D of G is the $n \times n$ diagonal matrix indexed by v with each diagonal element equal to the degree of vertex v. A *random walk* on G is defined as a sequence of vertices $(v_0, v_1, ..., v_k)$, each connected by an edge in E, where at each step the next vertex is chosen randomly from those neighboring the current step with equal probability.

A *non-backtracking random walk* is defined as a random walk that chooses its next step from all neighbors except the one it visited in the previous step. For really sparse graphs with dangling trees and nodes a simple modification, namely that if the only choice of edge is the one visited previously, then one resorts to backtracking for that edge ensures faster mixing and avoid absorbing condition.

It is well-known that random walks converge [Lovász, 1993]. Also in [Kempton, 2016, Alon et al., 2007] it is shown that NBT also converges but at a faster rate compared to random walks.

3 Algorithm

The algorithm takes as input the graph and outputs the embedding. The embeddings are computed using the now widely used word embedding algorithm [Mikolov et al., 2013] applied to the sample paths generated from a variation of the NBT random walk.

Algorithm: VEC-NBT Embedding

Input : Graph G
Output: Embeddings U
for node $v \in V,\ t \in \{1 \dots r\}$ **do**
  $S_{v,t}$:= begrudgingly-backtracking random walk of length l starting at v
end
$\{U_i\}_{i=1}^{n}$:= embedding vector for each node generated via word2vec using S

We note that VEC-NBT Embedding is built upon the idea in [Ding et al., 2016] that is shown to have consistently performed better than standard community detection methods, such as spectral clustering and acyclic belief propagation, both in accuracy and robustness to random initialization of the graph [Ding et al., 2016]. However, accuracy is still lower than desirable for very sparse graphs. VEC-NBT uses the NBT-RW which, in addition to offering a faster mixing rate [Alon et al., 2007] has also shown to be useful for spectral clustering [Krzakala et al., 2013].

4 Numerical Results

4.1 Clustering using k-means on embedded vectors for Stochastic Block Models (SBM)

Compress this paraThe graphs used to measure the performance of VEC-NBT are synthesized using the Stochastic Block Model (SBM). SBM builds off of the classical Erdos-Renyi $G(n, p)$ random graph model, where each edge of a graph with n nodes is formed with probability p. In SBM, specifically the planted partition model, $G(n, p)$ is additionally given k clusters and each node is added to one of the clusters with equal probability. Edges are formed within the cluster with probability a while inter-cluster edges are formed with probability b, with b lower than a in order to form an assortative graph. We have elected to use graphs with constant scaling, $b = \lambda a$. In our model, the probability of an intra-cluster edge being formed is $Q_n(k, k) = a = \frac{c}{n}$ while that of an inter-cluster edge being formed is $Q_n(k, k') = b = \frac{c(1-\lambda)}{n}$. c is the average degree of nodes within the cluster and determines the sparsity of the graph. λ determines how connected the various clusters are: $\lambda = 1$ would imply completely disjoint clusters, while $\lambda = 0$ draws no distinction between clusters. Thus, our model can be represented as $G(n, k, c, \lambda)$, which fully determines the graph.

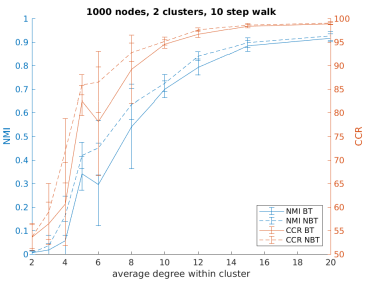
We compare the performance of VEC and VEC-NBT on SBM graphs generated using the parameters given through two metrics: Correct Classification Rate (CCR) and Normalized Mutual Information

(NMI). CCR is defined as the number of correctly classified points divided by the total number of nodes and NMI measures the statistical independence of the clusters, see [].

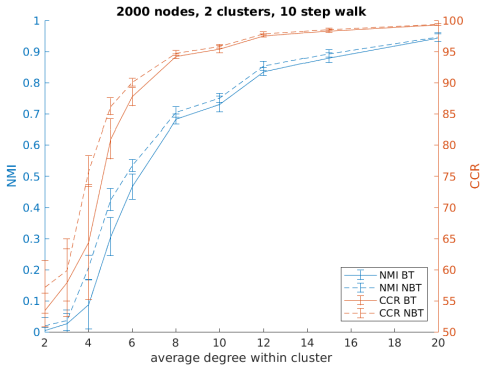
For random walks and embedding, VEC-NBT uses the same parameters used by VEC with the exception of the length of the random walk ($l = 5, 10$) and the window size ($w = 5$) and twice the number of random walks ($r = 20$). Here, we show empirically that VEC-NBT consistently achieves better accuracy than VEC for sparser graphs (low values of c) and comparable accuracy to VEC at higher sparsity levels.

Figures are shown with the original VEC algorithm ("BT") with solid lines and our new algorithm ("NBT") with dashed lines. CCR and NMI are shown for each algorithm on each plot. Note that red points correspond to CCR measurements, on a 50-100% scale, and blue points correspond to NMI measurements, between 0 and 1. NMI tends to be a more accurate indicator of performance.

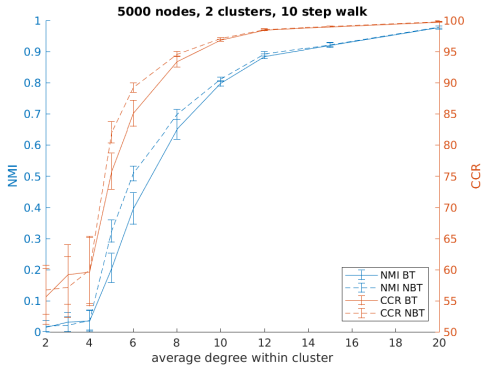
Unless otherwise specified, the x-axis is the sparsity of the graph, varying from 2 to 20; the number of clusters is 2; the graph has 10000 nodes; and the walks are 10 steps long.



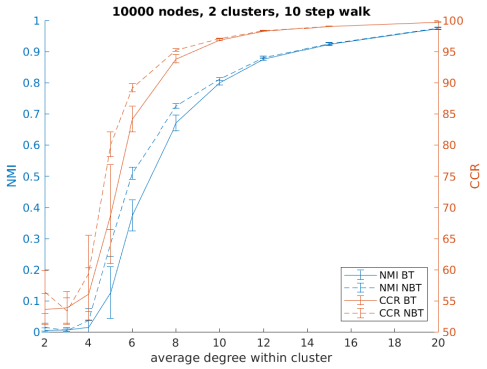
(a) 1000 nodes



(b) 2000 nodes



(c) 5000 nodes



(d) 10000 nodes

Figure 1: *Performance of both algorithms as a function of sparsity. We show performance for four differently-sized graphs. Note that measurement performance is noticeably better for VEC-NBT than for VEC.*

4.2 Performance on PPI networks

References

[Alon et al., 2007] Alon, N., Benjamini, I., Lbetzsky, E., and Sodin, S. (2007). Non-Backtracking Random Walks Mix Faster. *Communications in Contemporary Mathematics*, 09(04):585–603.

[Belkin and Niyogi, 2001] Belkin, M. and Niyogi, P. (2001). Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. *NIPS*.

[Cao et al., 2013] Cao, M., Zhang, H., Park, J., Daniels, N. M., Crovella, M. E., Cowen, L. J., and Hescott, B. (2013). Going the Distance for Protein Function Prediction: A New Distance Metric for Protein Interaction Networks. *PLoS One*, 8(10):e76339–.

[Ding et al., 2016] Ding, W., Lin, C., and Ishwar, P. (2016). Node Embedding via Word Embedding for Network Community Discovery. *CoRR*, pages 1–10.

[Grover and Leskovec, 2016] Grover, A. and Leskovec, J. (2016). node2vec. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 855–864, New York, New York, USA. ACM Press.

[Hashimoto et al., 2015] Hashimoto, T. B., Alvarez-Melis, D., and Jaakkola, T. S. (2015). Word, graph and manifold embedding from Markov processes. *arXiv.org*, page arXiv:1509.05808.

[Kempton, 2016] Kempton, M. (2016). Non-Backtracking Random Walks and a Weighted Ihara's Theorem. *Open Journal of Discrete Mathematics*, 6:207–226.

[Krzakala et al., 2013] Krzakala, F., Moore, C., Mossel, E., Neeman, J., Sly, A., Zdeborová, L., and Zhang, P. (2013). Spectral redemption: clustering sparse networks. pages 1–11.

[Lawrence, 2012] Lawrence, N. D. (2012). A Unifying Probabilistic Perspective for Spectral Dimensionality Reduction - Insights and New Models. *Journal of Machine Learning Research*.

[Levy and Goldberg, 2014] Levy, O. and Goldberg, Y. (2014). Neural Word Embedding as Implicit Matrix Factorization. *Advances in neural information processing ...*, pages 2177–2185.

[Lovász, 1993] Lovász, L. (1993). Random walks on graphs: A survey. *Combinatorics: Paul Erdos is Eighty*, 2(Volume 2):1–46.

[Mikolov et al., 2013] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *CoRR*, pages 1–9.

[Perozzi et al., 2014] Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). DeepWalk: Online Learning of Social Representations. *arXiv.org*, pages arXiv:1403.6652–710.

[Singer and Wu, 2012] Singer, A. and Wu, H. T. (2012). Vector diffusion maps and the connection Laplacian. *Communications on pure and applied ...*.

[Talmon et al., 2013] Talmon, R., Cohen, I., Gannot, S., and Coifman, R. R. (2013). Diffusion Maps for Signal Processing: A Deeper Look at Manifold-Learning Techniques Based on Kernels and Graphs. *IEEE Signal Processing Magazine*, 30(4):75–86.