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, followed by using the sample paths as inputs to a popular word embedding method namely word2vec. For the network of yeast (which one?) it is shown that our method is competitive with the state-of-art method based on computing Diffusion State Distance (DSD) between nodes (proteins) while offering significant advantages in computational speed. This preliminary results open up new ways to fill in missing protein functionality....

1 Introduction

Need a paragraph on PPI networks and literature, specifically on DSD. Why word2vec?

2 Method

2.1 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.

2.2 Algorithm

One para- 5-6 lines for the algo

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

VEC-NBT 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].

3 Numerical Results

3.1 Clustering on Stochastic Block Models (SBM)

Compress this paraThe graphs used to measure the performance of VEC-NBT are synthesized using the Stochastic Block Model (SBM), a canonical graph model used for community detection. 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). 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.

CCR is defined as the number of correctly classified points divided by the total number of nodes. To ensure the calculated clustering matches the ground truth, we use a linear sum assignment to match the cluster assignments to the original labels. Because of this, CCR is defined (at least for 2 clusters) only between 0.5 and 1, since a measured CCR of 0 would indicate that every 1 was labeled as a 2 and vice versa - which is in fact a perfect clustering. In our graphs we have plotted CCR as a percentage between $1/K$ and 1 where K is the number of clusters.

NMI is defined as the mutual information $I(X;Y)$ normalized by the square root of the entropies $H(X)$ and $H(Y)$:

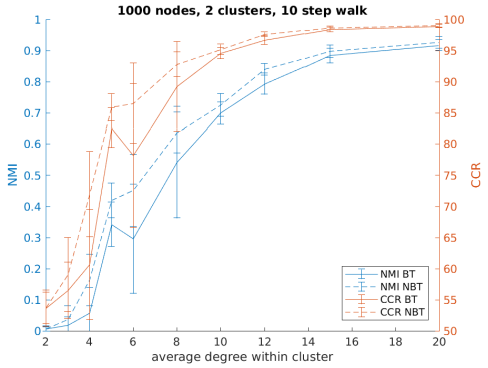
$$NMI(X,Y) := \frac{I(X;Y)}{\sqrt{H(X)H(Y)}} = \frac{H(X) + H(Y) - H(X,Y)}{\sqrt{H(X)H(Y)}},$$

a more technical metric measuring the information content of the resulting labels.

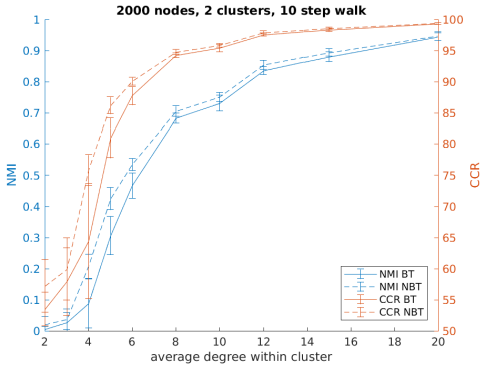
Figures are shown with the original VEC algorithm ("BT") with solid lines and our new algorithm ("NBT") with dashed lines. CCR and NMI are shown for each algorithm on each plot. Note that red

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.

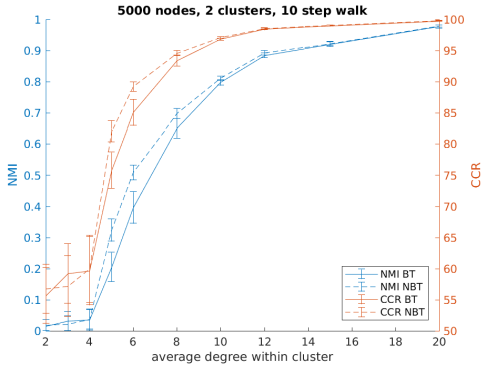
Unless otherwise specified, the x-axis is the sparsity of the graph, varying from 2 to 20; the number of clusters is 2; the graph has 10000 nodes; and the walks are 10 steps long.



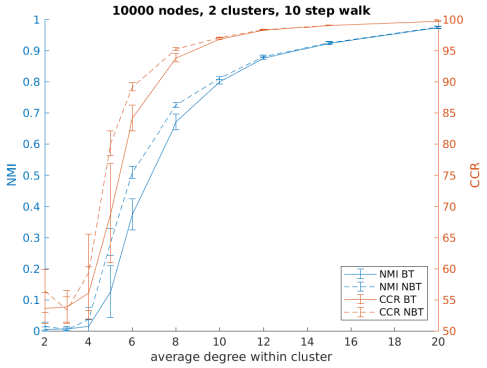
(a) 1000 nodes



(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.*

3.2 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.

[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.

[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.

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