Evaluation of Network Embedding framework combined with local nodes analysis

De Lorenzi Andrea, Fabris Daniele
Learning From Networks 2022/23
University of Padova - DEI

Introduction

Many real-world phenomena, including relationships, biological interactions, research collaborations, power systems, pandemics, and so on, can be abstracted into complex networks structures. Most of the time, these interconnections are bidirectional and have no load, resulting in undirected and unweighted networks. Identifying the most influential spreaders is crucial for understanding and controlling information propagation in many applications, such as disease control, market advertising, news promotion and information spreading. Different networks have different attributes that results in different top influential nodes identification processes.

Three of the most important classic centrality measures are the degree centrality, the betweenness centrality and the closeness centrality. The degree centrality makes use of the node's degree, as it can be guessed from the name, but it uses only the number of neighbors as measure and ignores all the other network characteristics. Therefore, degree centrality does not always perform well due to the lack of local network information. The betweennes centrality computes the number of shortest paths that pass through each vertex. The closeness centrality uses average shortest path distance between a node and all remaining ones in the network. Less the average distance, the more central it is and closer to all other nodes. Both of betweenness and closeness centralities use enough network information, but that makes them computationally heavy. For large networks, calculating them results unfeasible.

Many researchers have explored other techniques based on the network topology to identify a set of influential nodes that maximizes influence spreading. Recently k-shell or k-core decomposition technique has been developed. It is a method in which we can divide nodes in sets based on nodes' degree. Therefore k-shell places large number of nodes into a single set specifying that all nodes being part of a set have the same influence in the network. K-shell has a low computational complexity of $\mathcal{O}(m)$, where m is the number of edges in the network. However, the k-shell method assigns to many nodes the same index, which causes the monoticity problem. Thus, motivates researchers to search for improved heuristics that could alleviate this issue and node ranking becomes more close to the actual spreading influence.

In this study, the proposed method does not require the knowledge of network's specific properties for the identification of influential nodes but it is based on the analysis of topological properties, local information and network embedding. The main aspects covered in this paper are as follows.

- We suggest a node centrality index, which is able to identify the influence of nodes in the propagation. The algorithm makes use of NetMF network embedding and network local information, such as K-Shell decomposition and node's degree, in order to have low consumption in terms of time complexity and quite good accuracy.
- We use the NetMF network embedding method to map high-dimensional network's nodes into low-dimensional vectors in the Euclidean space. The computed Euclidean distances are used to replace the shortest path distances, but also contains local topology information, achieving better results.

Related works

Network embedding

Considering larger networks, the amount of data to be represented it is not negligible and therefore we need a way to represent this quantity efficiently. In the past years, researchers explored a low-dimensional and dense vector representation method in order to simplify the management of high-dimensional nodes in the network, which is called network embedding. Formally, the goal of network embedding is to map each vertex to a low-dimensional vector that capture its structural properties. The output representations can be used as input for mining and learning algorithms for a great variety of network science tasks.

DeepWalk considers vertex paths traversed by random walks over networks as the sentences and exploits the local structures in the network to learn latent vertex representations. With its appearance, many network embedding models have been developed, such as among the more common node2vec, LINE and PTE.

Instead of using them, we explored a different solution proposed in [1], which is called NetMF and offers significant improvements over other methods, unifying LINE, PTE, DeepWalk and Node2Vec in the framework of matrix factorization. NetMF explicitly factorized the closed-form matrices that DeepWalk and LINE aim to implicitly approximate and factorize.

Node centrality

Several methods to identify the influence of nodes have been proposed and, in general, higher is the ranking score associated with the node, the greater is its influence in the network. These methods are based on the node local features, local and/or global index-based centrality measures, the topology of the network. Some researchers proposed algorithms based on the law of gravitation extended from physics to networks studies, while other techniques focus on node aggregation, on density centrality, and so on. Global index-based centrality methods like closeness centrality and betweenness centrality, based on shortest path length computation, have high computational complexity and can't be applied in large-scale networks. Local index-based centrality algorithms have low computational complexity, since they consider only the neighborhood information, however they often have low accuracy and can't identify the nodes that are not core nodes but have high influential capability. There are also trade-off, like semi-local centrality-based methods that consider the 2-step neighborhood of nodes to balance cost and performance. On the contrary, the k-shell decomposition method is a fast node ordering method for large-scale networks, which iteratively removes the nodes on the basis of a certain degree value from 1 to k until there are no more nodes with that degree in the network; removed nodes are assigned a core value in terms of their degree, and higher is this index the more influential is the related node.

As we said before, k-shell has a low computational complexity, but assigns to many nodes the same index, which causes the monoticity problem.

A good solution has been proposed with mixed degree decomposition method which follows a similar approach to the k-shell except that it considers both residual nodes and exhausted nodes and resolved the monoticity problem. However, another issue rises where high degree nodes were given higher rankings, which is not always true in real-life scenario. In order to partition the network in a more refined way, weighted k-shell degree neighborhood method has been introduced with the degree, k-shell index and two free parameters. By taking into consideration its limitation, we propose a similar method with the absence of free parameters, which takes also into account the topology information thanks to network embedding.

Proposed method

In our study, we tried different alternatives in order to find the top influential nodes in different networks. Mainly, we followed the NLC algorithm proposed in [2], based on the definition of the local node centrality index:

$$NLC(i) = \sum_{j \in \Gamma(i)} Ks_i \times e^{-|\chi_i - \chi_j|^2} \tag{1}$$

where Ks_i represents the k-shell value of node i, χ_i, χ_j represents the vectors of the embedded nodes i, j, while $\Gamma(i)$ indicates the neighborhood of the node i. Despite the similar approach, our method differs in two main points: the embedding method ([1]) and the "weight" assigned to the couple of nodes in each iteration ([3]).

NetMF

NetMF is a matrix factorization framework that aims at improving the current formulation of other embedding methods, like DeepWalk and LINE. Such improvement was achieved by approximating, through a factorization process, the matrix extracted from the closed form equation of the DeepWalk algorithm (M), obtainable through the normalized graph Laplacian. As stated in [1], the decision of taking into account the closed form of DeepWalk is correlated to the method being more general, compared to LINE, and more computational efficient, compared to node2vec. The factorization process in [1] is performed for two cases, large window size and small window size. In the first one the matrix M and the logarithm of it have been approximated to reduce the overall time complexity associated with the use of a larger window size, while in the latter the factorization was directly applied to the two terms. Once the factorization of the terms is computed, the embedding can be extracted by applying SVD to the obtained results.

The main drawback associated to both processes is with the factorization of the $\log$M term, this factor has a direct influence over the complexity of the algorithm since the element-wise matrix logarithms require to be computed explicitly, reducing the overall gain over the other embeddings.

Combination of k-shell, degree and embedding

How a node spreads information does not just depend on its characteristics, but also on the local topology of the node. Generally, the influence between nodes decreases as the distances increases. Many node ranking heuristics regard all network edges with the same importance, but in this version of an edge weight based k-shell degree neighborhood method we assign a value to them, depending on the influence of the connecting nodes. In our algorithm 1, we propose a way to determine the influence between two nodes based on the computation of k-shell values and degrees, but also on network embedding. Numerically k-shell index and degree are not comparable, hence to compute a measure which consists of both of them we add k-shell indexes and degrees of the two nodes considered separately and only then, while adding them, introduce an equalizing factor that brings both the measures to a uniform scale. Generally, two free parameters are used in the formulation of edge weight. However, in our case, we use only a factor δ which is not a free parameter and is calculated as the ratio between the average of all k-shell values and the average of all degrees of the nodes in the network.

$$index(i,j) = (Ks_i + Ks_j) + \delta \cdot (K_i + K_j) \tag{2}$$

Here, Ks_i and Ks_j denote the k-shell values of nodes i and j, while K_i and K_j stand, respectively, for the degrees of node i and j. The equation 2 is not complete yet, because, as we said previously, there is the embedding to be taken into consideration. The influence of the nodes is not only related to the path distances, but also to the network structure. Therefore, the distances between nodes is replaced by the Euclidean distances between the low-dimensional vectors of the corresponding nodes, mapped with the NetMF embedding method. Since a spreader can influence not only its nearest neighbors but also neighbors' neighbors, we consider, principally, the third-level neighborhood, which is the one that gives the most satisfactory results compromising between accuracy and time complexity. In order to compute the influence 3 of a node i, we have to compute the index in 2 between the main node i and any other node in the set of neighborhood nodes $\Gamma(i)$ of i.

$$Influence(i) = \sum_{j \in \Gamma(i)} index(i, j) \cdot e^{-|\chi_i - \chi_j|^2} \quad (3)$$

Suppose the node i has two neighbors j and k, which Euclidean distances from i are respectively d_{ij} and d_{ik}, with $d_{ij} < d_{ik}$, indicating that the influence of node i on node j is greater than the influence of node i on node k. Therefore, when information spreads from node i to nodes j and k, node j decays more slowly. The exponential factor is quite important, because as we said before the index is dependent to the distance between the two nodes embeddings, greater is the distance, the lower the index associated to the edge will be. This way, in most cases, the higher the level-neighborhood of the "connected" node is, the lower the similarity between the two nodes embeddings will be, and therefore the closest nodes will be more influenced.

Algorithm 1 NLCKSD(G,N)

```
Require: graph G, integer k
Ensure: the top k nodes with greater influence
 1: NetMF: obtain the low-dimensional vectors of each
    node: $\chi_i = (x_{i,1}, x_{i,2}, ..., x_{i,r})$
 2: Compute the Euclidean distance between low-
    dimensional vectors of each pair of nodes in the
    network
 3: Compute the K-shell value $Ks_i$ for each node $i$ in the
    network
 4: Compute the degree $K_i$ for each node $i$ in the network
 5: for $i = 1$ to $N$ do
 6:     Get the third-level neighborhood set $\Gamma(i)$ of node $i$
 7:     for each node $j \in \Gamma(i)$ do
 8:         Compute $index(i, j)$ value of a pair of nodes using
            equation 2
 9:     end for
10:     Get the influence $Influence(i)$ of node $i$ using equa-
        tion 3
11: end for
```

Datasets and Hardware

Datasets

We've observed that NLC algorithm achieves good identification on top influential nodes in relatively small networks, like the ones presented previously. Therefore, we decide to consider larger datasets too and look for an algorithm capable of having a certain accuracy without being too expensive in both space and time complexity. The public datasets we have chosen are:

- dolphins [4] containing 62 nodes, 159 edges;
- lesmis [5] containing 77 nodes, 254 edges;
- polbooks containing 105 nodes, 441 edges;
- adjnoun [6] containing 112 nodes, 425 edges;

that can be found on http://www-personal.umich.edu/~mejn/netdata/.

- PPI [7] containing 3,890 nodes, 76,584 edges and 60 labels;
- wikipedia [8] containing 4,777 nodes, 184,812 edges and 40 labels;
- blogcatalog [8] containing 10,312 nodes, 333,983 edges and 39 labels;

that can be found on https://github.com/allenhaozhu/REFINE/tree/main/data. And

- Yeast containing 1,870 nodes and 2,277 edges;
- ego-facebook containing 2,888 nodes and 2,981 edges;

that can be found on https://konect.cc/networks/moreno_propro/ and https://http://konect.cc/networks/ego-facebook/ respectively.

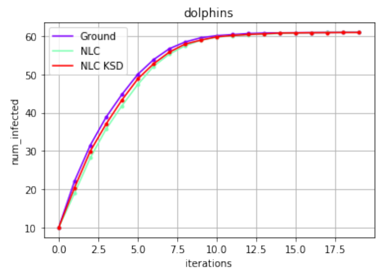
Hardware

The computer on which the tests are performed is equipped with: Processor AMD Ryzen 7 6800h with Radeon graphics , Installed RAM 2x16GB DIMM DDR5 4800MHz, System type 64-bit operating system, NVIDIA GeForce RTX 3070 8GB GDDR6, SSD 1TB M2.2 2280 PCIe Gen4 TLC.

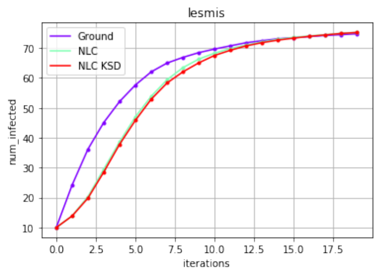
Experiments and analysis

Infection models

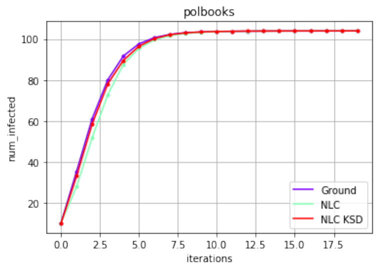
During the work of this paper, it was necessary to set a ground truth of the most influential nodes for the graphs used in the evaluation of the algorithms, in order to be able to compare their accuracy. Taking the proposed solution of the reference [2] to gain this essential information, the SIR and SI epidemic models were used to compute the influence of the nodes. In order to represent the infection state of the nodes the models label each node of the graphs into either Susceptible, Infected or Removed. The SI models use only the first two labels while the SIR model uses all three of them. The infection of each node is then modeled as a probability β representing the possibility of each Infected node to spread the infection to its adjacent Susceptible nodes, at each iteration of the modeled execution. The probability β is obtained from the product between a coefficient, associated to the used model, and β_h, a common threshold of both propagation models, which is computed as the ratio between the average degree of the nodes and the mean value of the square of the nodes degrees. In the case of SIR, the Removed state can be reached by Infected nodes with a probability μ, representing the probability of the nodes becoming immune.



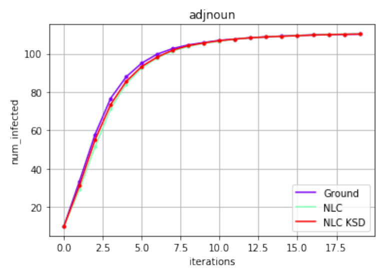
(a) dolphins



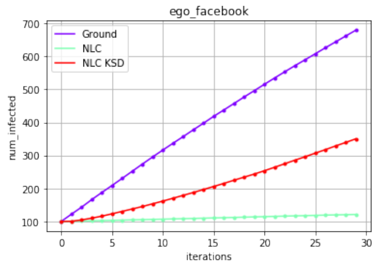
(b) lesmis



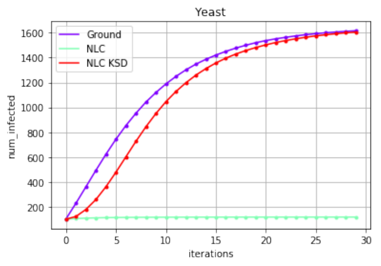
(c) polbooks



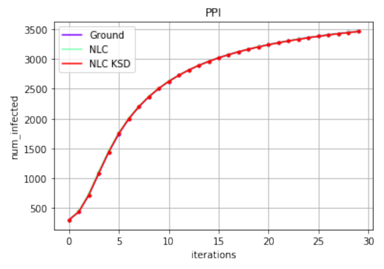
(d) adjnoun



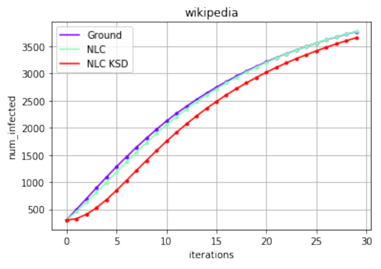
(e) ego_facebook



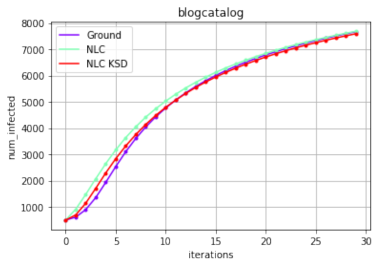
(f) Yeast



(g) PPI



(h) wikipedia



(i) blogcatalog

Figure 1: Graphical representation of the infections trend resulting from the SI model after setting as Infected nodes the most influential nodes of the ground truth and resulting from the NLC and NLC KSD algorithm. In order, (a), (b), (c) and (d) take the top 10 nodes in consideration, while (e) and (f) take the top 100 nodes in consideration. (g) and (h) consider the top 300 and (i) the top 500.

SIR

The true influence of nodes has been computed by instantiating the SIR model for each of them, setting the node as the only Infected node and the remaining ones as Susceptible. Together with this initial setting the probability of infection was set to $1.5\beta_h$ and the probability of the nodes to become Removed to 1. After the set up, iterations of the model where executed until no more nodes with label Infected where present in the graph. At this point the degree of influence is represented by the value F(t) consisting of the number of Infected and Removed nodes after t iterations of the model. The t value in this paper was set to ten for the smaller graphs, as in reference [2], while to either fifteen or twenty for the bigger ones since an earlier stop would not be able to correctly represent the influence of the nodes. Given the aleatory nature of the model the whole process was repeated multiple times in order to get as close as possible to the real values. In particular the process was repeated one thousand times for the smaller graphs and between twenty to fifty times for the bigger ones. This high difference in number of iterations is directly correlated to the complexity of performing the influential computation, which does not scale well with the increase in both nodes and edges number.

SI

The SI model was principally used to evaluate the actual infection of the nodes returned from the various algorithms in comparison between one another. In the set up of the model, the influential nodes were set to Infected and the remaining ones as Susceptible, while the probability of infection was set to $2\beta_h$. Differently from SIR, the Infected nodes do not lose their status so they keep on trying to infect their adjacent nodes at each iteration until all nodes have been infected. For the evaluation the number of infected nodes at each iteration

	dolphins	lesmis	polbooks	adjnoun	ego_facebook	Yeast	PPI	wikipedia	blogcatalog
NLC	60%	40%	60%	40%	0%	0%	58.33%	18.33%	23.4%
NLCKSD	80%	40%	60%	60%	6%	7%	42.66%	5.33%	24.8%

Table 1: Accuracy results of the influential nodes returned by the algorithms NLC and NLCKSD with respect to the ground truth on each of specified graphs.

for the various groups of influential nodes were compared and plotted.

Testing

Link to code: https://github.com/fabrisdnl/Project_LfN. The evaluation of the proposed algorithm was compared to the results of the NLC algorithm defined in [2] over the specified datasets. The overall accuracy results can be seen in 1 while the infection trend of the resulting most influential nodes for each dataset can be visualized in 1. Looking at the resulting trends, it is possible to state that for the first four plots, associated to the networks with smaller size, the trend of infection of the nodes resulting from the algorithms follows almost identically the ground truth one, with the only exception for the lesmis graph (b) where the initial trend of the algorithms results is lower with respect to the ground one. This can be associated, with the exception of the lesmis network, to both the accuracy obtained by the algorithms and their effectiveness in this networks in finding influent nodes that can compete with the ground truth ones.

Considering the results on the larger networks, it is possible to see that for the ego_facebook (e) and Yeast (f) networks the NLC algorithm fails to find the most influential nodes, as can be seen by the low accuracy result in 1, while the proposed algorithm performs a bit better in both accuracy and trend following. This obtained performances can be associated to the topology of the considered networks, which does not permit the analyzed algorithms to work as effectively as in other networks. Considering the last three networks instead the trend remains similar to the ground one. In particular, is practically equal for PPI (g) showing that the influential nodes reported by the algorithms are still highly influent even if the accuracy does not exceed the sixty percent mark. For the wikipedia (h) network the NLC algorithm has better results in both accuracy and trend with respect to the proposed algorithm, which follows from the network topology of wikipedia related networks that does not work well with the NetMF embedding, as stated in [1]. And lastly blogcatalog (i) trend comparison shows that the algorithms results achieve better infection level at the start then the ground truth, which is correlated to the high time requirement in computing the SIR model for the network resulting in a low number of iterations of the infection model.

Conclusion

In this study, we presented a node local centrality measure to identify the top influential nodes which is based on the potential edge weights related to the connecting nodes, estimated using k-shell values, degrees and network embedding. The algorithm achieved good identification results, comparable to the two algorithms from which we took inspiration, especially for small networks. Considering larger networks the proposed algorithm gets good result and thanks to the choice of the NetMF network embedding method, time complexity decreases substantially during the mapping into low-dimensional vectors.

References

[1] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. *Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec*. https://arxiv.org/pdf/1710.02971.pdf.

[2] Xu-Hua Yang, Zhen Xiong, Fangnan Ma, Xiaoze Chen, Zhongyuan Ruan, Peng Jiang, Xinli Xu. *Identifying influential spreaders in complex networks based on network embedding and node local centrality*. https://www.sciencedirect.com/science/article/pii/S0378437121002430.

[3] Giridhar Maji. *Influential spreaders identification in complex networks with potential edge weight based k-shell degree neighborhood method*. https://www.sciencedirect.com/science/article/pii/S1877750319304181.

[4] David Lusseau, Karsten Schneider, Oliver J. Boisseau, Patti Haase, Elisabeth Slooten, Steve M. Dawson. *The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations*. https://link.springer.com/article/10.1007/s00265-003-0651-y.

[5] Donald E. Knuth. *The Stanford GraphBase: A Platform for Combinatorial Computing*. https://mirror.gutenberg-asso.fr/tex.loria.fr/sgb/abstract.pdf.

[6] M. E. J. Newman. *Finding community structure in networks using the eigenvectors of matrices*. https://journals.aps.org/pre/abstract/10.1103/PhysRevE.74.036104.

[7] Chris Stark, Bobby-Joe Breitkreutz, Andrew Chatr-Aryamontri, Lorrie Boucher, Rose Oughtred, Michael S Livstone, Julie Nixon, Kimberly Van Auken, Xiaodong Wang, Xiaoqi Shi. *The BioGRID interaction database: 2011 update. Nucleic acids research 39, suppl*$_1(2010), D698–D704$. https://link.springer.com/article/10.1007/s10115-013-0644-8.

[8] Lei Tang, Huan Liu. *Relational learning via latent social dimensions. In KDD. ACM, 817–826*. https://github.com/allenhaozhu/REFINE/tree/main/data.

[9] Vahid Shirbisheh. *Local Graph Embeddings based on Neighbors Degree Frequency of Nodes*. https://arxiv.org/pdf/2208.00152.pdf.

[10] Mingqiang Zhou, Haijiang Jin, Quanwang Wu, Hong Xie, Qizhi Han. *Betweenness centrality-based community adaptive network representation for link prediction*. https://link.springer.com/content/pdf/10.1007/s10489-021-02633-7.pdf.

[11] Shay Deutsch, Stefano Soatto. *Graph Spectral Embedding using the Geodesic Betweeness Centrality*. https://arxiv.org/pdf/2205.03544.pdf.

[12] Xuan Yang, Fuyuan Xiao. *An improved gravity model to identify influential nodes in complex networks based on k-shell method*. https://www.sciencedirect.com/science/article/pii/S0950705121004603.

[13] S. Raamakirtinan, L. M. Jenila Livingston. *Identifying Infuential Spreaders in Complex Networks Based on Weighted Mixed Degree Decomposition Method*. https://link.springer.com/article/10.1007/s11277-021-08772-x.

[14] Fatiha Souam, Ali Aïtelhadj, Riadh Baba-Ali. *Dual modularity optimization for detecting overlapping communities in bipartite networks*. https://link.springer.com/article/10.1007/s10115-013-0644-8.

[15] Linda J.S. Allen. *Some discrete-time SI, SIR, and SIS epidemic models*. https://www.sciencedirect.com/science/article/abs/pii/0025556494900256.

Contribution

De Lorenzi Andrea

Initial study of the [2] for first components definition. Set up of the infection models to gather the required ground truths of the datasets and subsequent extraction of the ground truths. Analysis and representation of the infection trend of the influencial nodes resulted from the application of the algorithms. Work on the infection models in the Python code (influence_models.py) and utilities method set up. In the paper, work on infection models section and NetMF embedding basic description.

Fabris Daniele

Research work on node centrality, considering local topology information based on combination of k-shell, degrees and network embedding. Studies on major papers, like [2] and [3], and other minor ones, in order to deploy a well structured algorithm which could compete with those of reference. The path that led us to the proposed method required the study of different index-based node centrality heuristics, from simpler one like degree centrality, betweennes centrality, closeness centrality, to more complex ones like gravitational model, mixed degree decomposition and so on. Major work on the local node analysis in the Python code (algorithm.py), testing methods, while in the paper on introduction, related works, proposed method (not NetMF), conclusion.