GECS: Graph Embedding Using Connection Subgraphs

Saba A. Al-Sayouri
State University of New York at Binghamton
ssyouri1@binghamton.edu

Pravallika Devineni
University of California Riverside
pdevi002@ucr.edu

Sarah S. Lam
State University of New York at Binghamton
sarahlam@binghamton.edu

Evangelos E. Papalexakis
University of California Riverside
epapalex@cs.ucr.edu

Danai Koutra
University of Michigan
dkoutra@umich.edu

ABSTRACT

This paper studies the problem of learning large-scale graph representations (a.k.a. embeddings). Such representations encode the relations among distinct nodes on the continuous feature space. The learned representations generalize over various tasks, such as node classification, link prediction, and recommendation. Learning nodes representations aims to map proximate nodes close to one another in the low-dimension vector space. Thus, embedding algorithms pursue to preserve local and global network structure by identifying nodes neighborhood notions. However, the means proposed methods have been employed in order to identify nodes neighborhoods fail to precisely capture network structure. In this paper, we propose a novel scalable graph embedding algorithmic framework called GECS, which aims to learn graph representations using connection subgraphs, where analogy with electrical circuits has been employed. The connection subgraphs are created to address the proximity among each two non-adjacent nodes, which are abundant in real-world networks, by maximizing the amount of flow between them. Although a subgraph captures proximity between two non-adjacent nodes, the formation of the subgraph addresses the direct connections with immediate neighbors as well. Therefore, our algorithm better preserves the local and global structure of a network. Further, despite the fact that non-adjacent nodes are numerous in real-world networks, our algorithm can scale to large-scale graphs, because we do not deal with the graph as a whole, instead, with much more smaller extracted subgraphs.

Since our algorithm is not yet empirically examined, we here introduce a potential solution that can better learn graph representations comparing to existing embedding methods accompanied by rational reasoning.

KEYWORDS

information networks, network flow, learning graph representations, node embedding

ACM Reference format:
Saba A. Al-Sayouri, Pravallika Devineni, Sarah S. Lam, Evangelos E. Papalexakis, and Danai Koutra. 1997. GECS: Graph Embedding Using Connection Subgraphs. In *Proceedings of Mining and Learning with Graphs Workshop, Halifax, Nova Scotia Canada, August 2017 (MLG'17),* 5 pages.
DOI: 10.475/123_4

1 INTRODUCTION

Real world information networks (a.k.a. graphs), such as social networks, biological networks, co-authorship networks, and language networks are ubiquitous. Further, the large size of networks — millions of nodes and billions of edges — and the massive amount of information they convey [3], have led to a serious need to efficiently and effectively mine such networks. Conventional graph mining algorithms [6] have been designed to learn a set of hand-crafted features that best performs to conduct a specific downstream task; i.e., link prediction [10], node classification [2], and recommendation [17]. However, present research has steered the direction towards a more efficient mean to mine large-scale graphs; feature learning [1]. That is, a unified set of features that can effectively and efficiently generalize over distinct graph mining-related tasks is exploited. To this end, serious recent research efforts have been paid in order to design either unsupervised or semi-supervised algorithms to learn feature representations. Such efforts have been initiated in the domain of natural language processing (NLP) [9, 11, 12], where two word2vec [11] models have been proposed, namely continuous bag of words (CBOW) and Skipgram. Inspired by the recent advancements in NLP, and the analogy in the context, various algorithms have been developed to learn graph representations [7, 13, 15]. However, some recent proposed algorithms fail to clearly define and optimize an objective that is tailored for graph nature [13]. That is, the optimized objective was originally developed in the context of natural language, where the relations are more of a linear nature. Further, they employ totally random [13] or biased [7] random walks in order to attain nodes neighborhood notions, which still appear to be unsatisfactory in performing downstream processes; i.e., link prediction and node classification. This can be attributed to the fact that the underlying intuition of random walks cannot address the actual existing flow among graph entities, which eventually can impact the captured proximity among ubiquitous non-adjacent nodes in real-world graphs. Likewise, even algorithms, which adopted more deep models instead of random walks, to learn nodes representations failed to sufficiently perform graph mining-related processes.

It is worth mentioning that the quality of learned representations is heavily influenced by the preserved local and global structure of a network. In order to preserve the network structure, we have to precisely and carefully identify each node neighborhood notion. A node neighborhood notion captures its observed and unobserved connections represented by its local and global connectivity respectively. For that, and to the best of our knowledge, we are the first to develop a scalable graph embedding algorithm that can preserve connectivity patterns unique to undirected and (un)weighted graphs using the concept of network flow represented by connection subgraphs [4]. The connection subgraphs avail the analogy with electrical circuits, where a node is assumed to serve as a voltage source and an edge is assumed to be a resistor, where its conductance is considered as the weight of the edge. When forming the connection subgraph, we concurrently capture the observed and unobserved connections, and we account for the node degree imbalances by downweighing the importance of paths through high-degree nodes (hubs) and by taking into account both low- and high-weight edges. Further, using connection subgraphs confers us to account for metadata that is not well-addressed by existing methods. It is important to notice that our goal in forming connection subgraphs is to maximize the flow between pairs of non-adjacent nodes along with avoiding long paths, where generally information is lost, therefore, the formation process is distance and flow-driven. Although our proposed algorithm is not yet empirically examined, we expect that it will outperform the state-of-the-art approaches, as we more carefully and precisely preserve network local and global structure using a global-related proximity measure. A global measure that also accounts for local connections would better serve graph mining-related processes, such as link prediction and node classification. This can be attributed to two reasons: 1) One of the most common closure processes in social networks in particular, is triadic closure, which represents an increased probability that two users will be connected due to the presence of a mutual friend. Such a closure is hard to capture using a local or inefficient global proximity measure, and 2) A large percentage of connections in real-world networks are unobserved. To summarize, our contributions are as follows:

1. **Flow-based Formulation.** We propose a graph embedding approach that robustly-preserves network local and global structure using connection subgraph (GECS) algorithm to learn graph representations using the notion of network flow to produce approximate but high-quality connection subgraphs between pairs of non-adjacent nodes in undirected and (un)weighted large-scale graphs. We use the formed connection subgraphs to identify the nodes neighborhoods and not restrict ourselves just to one- or two-hop neighbors.
2. **Tailored Objective.** We optimize an objective function that is tailored to obey with network structure and account for node-degree distribution.

2 RELATED WORK

2.1 Conventional Graphical Representation Learning

The recent research developments in natural language have led to replacing feature engineering [6] by feature learning [1] and have influenced many fields, including graph mining. Work on graph embedding focuses on learning a set of node features or representations to perform various tasks, i.e., link prediction [10], node classification [2], and recommendation [17]. Therefore, deep learning [5] has come to the picture, where complex non-linear relationships are learned. It is important to notice that representation learning falls under the umbrella of unsupervised or semi-supervised learning [5], where the label information is not or is barely utilized. Since label information is scarce for real-world networks [3], representation learning has attracted the attention of the machine learning and data mining communities. The conventional unsupervised learning methods can be envisioned as a dimensionality reduction techniques [14, 16]. Such methods harness the spectral characteristics of graph's matrix representations, i.e., Laplacian and adjacency matrices. Therefore, they endure from high computational complexity and compromised statistical performance. In terms of computational complexity, they use the eigendecomposition of the data matrix, which makes it very inefficient and unscalable for large-scale data sets,unless it is restricted to the few top eigenvalues. Therefore, for the sake of being more efficient, approximations have been employed [7], which in turn compromise their generalization capabilities to be less useful for statistical models. Further, these methods optimize objective functions that are not tailored for networks. In other words, such objectives do not account for the unique connectivity patterns emerge in real-world networks, such as such as homophily and structural equivalence.

2.2 Graphical Representation Learning Recent Trends

Recent work in network representation learning has been largely motivated by new progress in natural language processing (NLP) domain [9, 11, 12], due to the existing analogy among the two fields, where a network is represented as a document. One of the NLP leading advancements is related to the word2vec models [11], namely, continuous bag of words (CBOW) and Skipgram. The Skipgram model has been more widely adopted by virtue of its efficiency in scaling to large-scale real-world networks. In a nutshell, it learns continuous distributed feature representations of words in a corpus by maximizing the probability of observing the context words given the representation of a target word. The underlying idea is that the words that usually appear in similar contexts tend to have similar meaning, thus have similar representation vectors, which in turn leads to embedding them close to one another in the low-dimensional continuous vector space. However, adopting the Skipgram model for graph learning representations imposes the need for some adjustments, for two reasons. First, more sophisticated highly-nonlinear relationships emerge in real-world networks comparing to mere simple linear regularities among words. Second, the objective that Skipgram optimizes is designated to satisfy the nature of corpora, where the probability of observing the surrounding words appear to the left and right of a given target word is maximized [11].

Recent proposed algorithms, which adopted Skipgram model appear to be incapable to satisfactorily and accurately capture the nodes neighborhood notions in a network [7, 13, 15], which is completely different from what corpora used to have. Specifically, DeepWalk [13], for instance, employs the random walk to attain the neighborhood of a node in a graph, which eventually introduces some noise in the search process. Further, LINE [15] proposes to preserve the network local and global structure using first and second-order proximities, respectively, along with using an edge-sampling algorithm to surpass the limitations of optimization using stochastic gradient descent. In this framework, an edge is sampled with a probability proportional to its weight, which ultimately enhanced the effectiveness and efficiency of the inference process. However, the sampling process ignores the strength of weak ties—i.e., edges with low weights. It is worth mentioning that generally, crucial information flows through weak ties do not flow through strong ties—i.e., edges with high weights [3]. A more recent approach, node2vec [7], proposes to preserve graph unique connectivity patterns—i.e., homophily and structural equivalence, using a more flexible, controllable, and carefully-designed search strategy to identify nodes neighborhoods, and specifically biased second-order random walks that combine the benefits of BFS and DFS. However, still relying on such biased walks to identify nodes neighborhoods appears to be insufficient due to the unsatisfactory prediction accuracy of link prediction and node classification-related tasks.

3 PROPOSED METHOD: GECS

In this section, we describe the main components of our algorithm as shown in Figure 1. In contrast to previous work [7, 13], which employs completely random or biased random walks in order to attain nodes neighborhood notions, we propose GECS, a method that identifies neighborhood notions using connection subgraphs and is capable of capturing both local and global—beyond two hops—connectivity patterns efficiently. Connection subgraphs [4] is an algorithm that was originally developed to capture proximity between a pair of two non-adjacent nodes in an undirected (un)weighted graph. Unlike random or biased walks, which identify node neighborhoods by starting a set of walks from that node without targeting any destination node, we propose to generate node neighborhoods based on proximities between pairs of nodes. Therefore, we overtly identify the node's neighborhood while maximizing its proximity to other non-adjacent nodes. In a nutshell, using connection subgraphs, we can: (1) better control the search space; (2) benefit from actual flow; (3) exploit the strength of weak ties; (4) avoid introducing randomness; (5) integrate two extreme search strategies, breadth-first search and depth-first search [18], where immediate neighbors and neighbors at increasing distances from source and destination nodes are considered, respectively; (6) address the issue of high-degree distribution, that is, a node with high-degree distribution has a low chance to be included in the connection subgraph, as a pair of nodes might be incidentally connected through a high-degree distribution node.

Our algorithm consists of two main steps, which we discuss in the next two subsections: (1) Neighborhood notion identification and (2) Representation vector update.

3.1 Step 1: Neighborhood Notion Identification

According to the analogy with electrical circuits, and by using the connection subgraph algorithm which maximizes the approximate flow between pairs of nodes, we identify the nodes' neighborhood notions. The use of a subgraph instead of a single critical path among two non-adjacent nodes is beneficial for multiple reasons: (1) Choosing the most important path using a fully automated mechanism is susceptible to errors; (2) Showing a subgraph elevates the probability of having the critical path, if present; (3) The connection subgraph is better at characterizing the random spread of information in real-world networks than a single path.

This step consists of three phases: (A) Candidate generation, (B) Display generation, and (C) Node neighborhood generation. In the first two phases, we implement subgraph connection algorithm [4]. Our contribution lies in the third phase, where we employ the generated *display* subgraphs to generate nodes neighborhood notions. We provide an overview of each phase as follows:

Phase A: Candidate Generation. This phase represents the first phase of connection subgraph algorithm, in which we quickly extract a much smaller subgraph, called *candidate* subgraph, that contains most prominent paths between pairs of non-adjacent nodes in the original graph. The generated *candidate* subgraph serves as an input to the next phase, i.e., the display generation.

At a high level, we form the *candidate* subgraph by gradually and neatly 'expanding' the neighborhoods of the two nodes until we have a 'significant' overlap. Specifically, we expand the neighborhoods on a distance basis using the function $\log(deg^2(u)/C(u,v)^2)$, which measures the distance between any two nodes u and v, where $deg(u)$ represents the summation of weights incident to node u, and $C(u,v)$ indicates the weight of the edge between node u and node v. That is, we include the node, which is closer to either the starting or the destination node, based on which can eventually reduce the information loss more. Since a pair of nodes is more probable to be incidentally connected together though a high-degree distribution node, the candidate generation algorithm employs a penalty term α for distance computation. Thus, a high-degree node appears to be farther from its root (starting or destination node) compared to a low-degree node.

Phase B: Display Generation. This phase removes any remaining spurious portions in each generated *candidate* subgraph. The display generation phase takes a *candidate* subgraph as an input and returns *display* subgraph as an output. The *display* subgraph is generated on network flow basis. Let s and t are the source and destination nodes in the generated *candidate* subgraph. Thus, in the display generation phase, we aim to add an end-to-end path at a time between the source and destination nodes that maximizes the delivered current (network flow) over all paths of its length. Typically, for a large-scale graph, the *display* subgraph is expected to have 20-30 nodes.

Phase C: Node Neighborhood Generation. After we implement the connection subgraph two phases, we here show our major contribution in detail. Our contribution lies in the way we generate nodes neighborhood notions from generated *display* subgraphs. It is important to notice that we avoid following previous work regime in generating neighborhood notions using a set of fixed-length random walks initiated from each node in a graph. Let v_i be any node in a graph G, and let D_{v_i} be the set of *display* subgraphs generated between node v_i and the rest of non-adjacent nodes in G. We identify v_i neighborhood notion as follows: 1) We

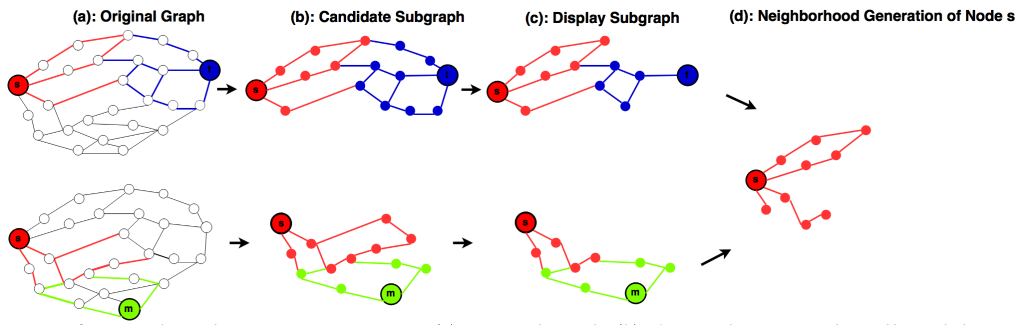

Figure 1: A description of GECS algorithm main components: (a) Original graph. (b) The quick generated small candidate subgraphs between nodes s and t first, and s and m second, (c) The display subgrahs with extra removal of spurious regions from previously generated candidate subgraphs, (d) The generated neighborhood notion of node s.

traceback each node shows in the set of generated *display* subgraphs to ascertain whether it has been expanded from v_i during candidate generation phase or not 2) We add end-to-end paths that comprise nodes originally expanded from v_i to represent its neighborhood notion. Our mean in identifying the neighborhood notions appears to be more concrete and rational for the following reasons: 1) We address the local and global structure of network by accounting for the immediate neighbors and neighbors at increasing distances of the source node to identify neighborhood notions; 2) We select the nodes on distance and network flow bases; 3) We address the issue of high-node degree distribution; and 4) We concurrently identify neighborhood notions while maximizing proximity among non-adjacent nodes, which are very abundant in real-world networks.

3.2 Step 2: Representation Vector Update

After identifying the nodes neighborhood notions in a graph, we now aim to learn representations. For that, and due to the analogy established by recent research with NLP modeling, where a graph is treated as a document, here, we employ an extended version of SkipGram model [11], for which a word neighborhood notion has been identified using a sliding window over consecutive words. Skipgram model maximizes the probability of observing the context word(s) given a target word to learn distributed word representations on the continuous feature space. In networks context, we pursue to learn and update nodes representations using and extended version of Skipgram model. We employ an extended version to account for the sophisticated connectivity patterns; non-linear relations, emerge in networks but not in corpora.

4 FUTURE EVALUATION PLAN

In order to demonstrate the superiority of our GECS, here we list the details of our evaluation plan:

1. **Proximity Measure.** Since our major claim is that proximity measures that have been employed by existing embedding algorithms are not capable enough to preserve network local and global structure, here we plan to compare GECS model that exploits the notion of network flow with other embedding algorithms that employ other proximity-based measures, such as random or biased walks and others.
2. **Path vs. Subgraph.** To emphasize the prominence of capturing proximity among two non-adjacent nodes in a network using a subgraph rather than a single good path. For that, we will employ single good path-related proximity measures, such as shortest path [8], which represents the length of the shortest path connecting two non-adjacent nodes together.
3. **Distance Function.** As a part of the candidate generation phase, we employ a distance function $\log(deg^2(u)/C(u,v)^2)$ in order to pick the next node to expand. However, the decision of the next node to expand can vary by altering the distance function. For that, we will employ other distance functions to explore the influence of such a variation on the identified neighborhood notions.

5 CONCLUSIONS

We propose GECS, a novel flow-based, algorithmic framework to learn undirected and (un)weighted graph representations. To identify the nodes' neighborhood notions beyond a small number of hops, we employ connection subgraphs and exploit the concept of network flow to capture proximity between pairs of non-adjacent nodes. Our algorithm is the first to harness network flow to attain neighborhood notions and effectively capture both local and global connectivity patterns. Furthermore, GECS leverages the strength of weak ties (which other methods have neglected) and computes proximities that are robust to the issue of high-degree nodes. In other words, in our framework, two nodes connected via a high-degree node will likely have lower proximity than two nodes with the same distance but intermediate connections to low-degree nodes.

In our future work, we will exploit networks from distinct domains, i.e., social networks, language networks, and biological networks, to show our proposed method's capabilities. Further, we will seek to more efficiently and effectively identify node neighborhoods from the generated display subgraphs. We will also address the issue of embedding update, especially for a recently-joined nodes that has no evident connections. This problem is very related to the "cold-start" problem in recommendation systems, where a new user joins the system and we seek external information for them, in order to properly compute their profile. Similarly, we propose to explore different forms of external context and meta-data for the recently-joined nodes which can help us address connection sparsity.

REFERENCES

[1] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

[2] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. 2011. Node classification in social networks. In *Social network data analytics*. Springer, 115–148.

[3] David Easley and Jon Kleinberg. 2010. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press.

[4] Christos Faloutsos, Kevin S McCurley, and Andrew Tomkins. 2004. Fast discovery of connection subgraphs. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 118–127.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. Deep learning (adaptive computation and machine learning series). *Adaptive Computation and Machine Learning series* (2016), 800.

[6] Palash Goyal and Emilio Ferrara. 2017. Graph Embedding Techniques, Applications, and Performance: A Survey. *arXiv preprint arXiv:1705.02801* (2017).

[7] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 855–864.

[8] Yehuda Koren, Stephen C North, and Chris Volinsky. 2007. Measuring and extracting proximity graphs in networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1, 3 (2007), 12.

[9] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 1188–1196.

[10] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology* 58, 7 (2007), 1019–1031.

[11] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[12] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

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

[14] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *science* 290, 5500 (2000), 2323–2326.

[15] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 1067–1077.

[16] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *science* 290, 5500 (2000), 2319–2323.

[17] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 283–292.

[18] Rong Zhou and Eric A Hansen. 2006. Breadth-first heuristic search. *Artificial Intelligence* 170, 4-5 (2006), 385–408.