Motif-Aware Graph Embedding

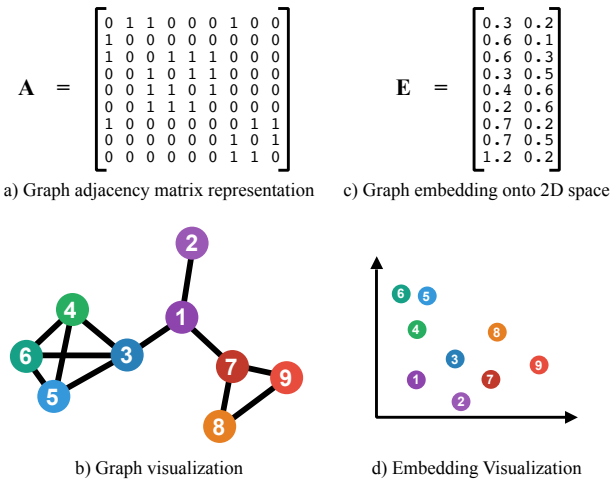
Anonymous Author(s)
Anonymous Institute
Anonymous Email Address(es)

Abstract

Given a large complex graph, how can we have lower dimension real-vector representations of vertices that preserve structural information? Recent advancements in graph embedding have adopted word embedding techniques and deep architectures to propose feasible solutions to this question. However, most of these former researches consider the notion of "neighborhood" by vertex adjacency only. In this paper, we propose a novel graph embedding algorithm that employs motif structures into the latent vector representation learning process. Our algorithm learns the graph latent representation by contrasting between different type of motif-biased random walk. We show that our algorithm yields more accurate embedding results compared with other existing algorithms through various graph mining benchmark tasks.

1 Introduction

The graph (or network) data model is a useful tool for a wide range of disciplines, and it is essential to have a low dimensionality representation of a complex graph. In its simplest form, a graph is an ordered set of vertices connected by edges. Being simple and expressive, the graph-theoric approach has been applied in many scientific fields for untangling complex discrete structures. For example, the study conducted by MP Van Den Heuvel suggested that the human brain functional network provides many new discoveries about our brain's organization (Van Den Heuvel and Pol 2010). The same approach for structural analysis can be found in other fields such as chemistry (Bader, Betel, and Hogue 2003), physics (Newman 2010), and sociology (Callon 1986). However, the graph analysis process often becomes intractable due to the complexity of the data. In such case, the graph usually contains several thousand to millions of vertices and edges. Therefore, it is desirable to have a compact latent representation of the graph while its statistical properties are retained. A *high-quality* latent representation of a graph can benefit machine learning algorithms in many ways. For instance, the result and runtime of machine algorithm will be improved thanks to the low dimensionality of the data. On the other hand, instead of relying only on graph mining algorithms to analyze some given data, researchers can also apply other machine learning algorithms on the learned latent representation to make predictions. Conventionally, the latent representation learning procedure on a graph is called *graph embedding*.



a) Graph adjacency matrix representation

c) Graph embedding onto 2D space

b) Graph visualization

d) Embedding Visualization

Figure 1: Graph dimension reduction.

Learning a high-quality latent representation is a challenging task. Traditionally, dimensionality reduction technique such as PCA (Jolliffe 2002), EigenMaps (Belkin and Niyogi 2001), and IsoMap (Tenenbaum, De Silva, and Langford 2000) are used. Although these aforementioned techniques have a profound theoretical background (Fodor 2002), they are impractical due to computational drawbacks. To address the dimensionality problem, (Perozzi, Al-Rfou, and Skiena 2014) proposed Deepwalk - a Skipgram-based algorithm for graph embedding (Perozzi, Al-Rfou, and Skiena 2014). Different from traditional linear algebra approach, Deepwalk learns the latent representation from the probabilistic point of view. By treating a random series of vertices as if it is a sentence of words, Deepwalk adopts the operation of Skipgram model (Mikolov and Dean 2013) to learn a vertex's latent representation. Following Deepwalk, other Skipgram-based graph embedding algorithms which aim to improve embedding quality were proposed (Cao, Lu, and Xu 2015;

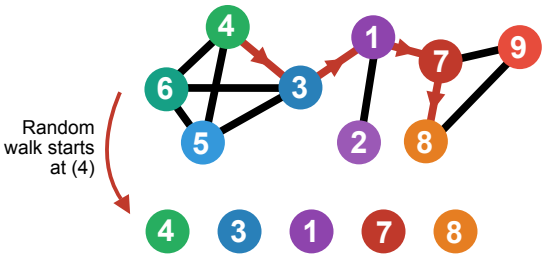


Figure 2: Random walk generates "sentence" from graph.

Tang et al. 2015; Yang, Cohen, and Salakhutdinov 2016; Grover and Leskovec 2016). However, these algorithms do not consider the intrinsic *motif structure* of a graph. Therefore, the performance of these algorithms depends heavily on heuristic hyper-parameter tuning.

In this work, we propose an algorithm which controls both graph context and negative samples generation. Our motif-aware context generation aims to emphasize the importance of local motif community, which is a strong indicator for true community in a graph. On the other hand, negative sampling is known to be the state of the art technique to estimate the normalization factor of a probabilistic model. Instead of sampling from a distorted unigram distribution as suggested by (Mikolov and Dean 2013), we propose a motif-aware method to generate negative samples from a graph. Generally, our algorithm, named Motif-Aware Graph Embedding (MAGE), has two following advantages:

- Positive samples are generated using a motif-biased walk. This motif-aware context generation procedure is backed by the hypothesis that vertices in the same motif are more likely to belong to the same community (Benson, Gleich, and Leskovec 2016). By selecting the appropriate motif for each graph, we have a sensible way to control the context generated for embedding.
- Negative samples are also generated using a motif-biased walk. However, by choosing the *opposing* motif to the characteristic motif of the graph, we have a concrete method to "escape" the motif community, which produces contrasting negative samples, hence better quality embedding.

Our algorithm is implemented on Keras (Chollet 2015) framework. The implementation and experimental results are available anonymously on Github[1].

The remaining of this paper is divided into 4 parts. Section 2 provides additional information about related work on graph embedding and graph motif. Section 3 presents our algorithm and experimental design. Section 4 and 5 discusses results and conclusion.

2 Related Works

2.1 Skipgram Model

Representation learning has been one of the keys to the success of machine learning algorithms (Bengio, Courville, and Vincent 2013). In the context of natural language processing (NPL), representation learning becomes even more important as the data has a discrete, but high-dimension nature. To address the dimensionality problem in NLP, (Mikolov and Dean 2013) have proposed the Skipgram model. Instead of maximizing the n-gram distribution as prior works, Skipgram maximizes the co-occurrence probability of context words given a target word. The softmax potential function for a context word given a target word is given by:

$$\Pr(v_c|v_t) = \frac{\exp\left(\langle \omega_{v_c}, \omega_{v_t} \rangle\right)}{\sum_{k \in V} \exp\left(\langle \omega_{v_k}, \omega_{v_t} \rangle\right)}, \tag{1}$$

where $\langle \cdot, \cdot \rangle$ is vector dot product; v_c and v_t are the tokens for the context word and the target word respectively; ω_{v_t} is the embedding vector selected from the *embedding matrix* by the token v_t; ω_{v_c} is the embedding vector selected from the *context matrix* by the token v_c.

Based on equation 1, the objective of Skipgram model is to maximize the following average log-likelihood:

$$\mathcal{O} = \max\left(\frac{1}{T}\sum_{t=1}^{T}\sum_{-c \le j \le c, j \ne 0} \log \Pr(v_{t+j}|v_t)\right) \tag{2}$$

The intrinsic intractable problem for softmax model is normalization factor computation. Therefore, the normalization factor in equation 1 needs to be estimated by approximation techniques such as Hierarchical Softmax (Morin and Bengio 2005) or Noise Contrastive estimation (Gutmann and Hyvärinen 2010). In their landmark paper, (Mikolov and Dean 2013) also proposed *Negative Sampling* - a simplified version of noise contrastive estimation. The log-likelihood of under negative sampling scheme is given by:

$$\begin{aligned}\log \Pr(v_c|v_t) &= \log \sigma(\langle \omega_{v_c}^{\text{nce}}, \omega_{v_t}^{\text{emb}} \rangle) \\ &+ \sum_{i=1}^{k} \mathbb{E}_{\omega_{v_i} \sim P_n(\omega)} \left[\log \sigma(-\langle \omega_{v_i}^{\text{nce}}, \omega_{v_t}^{\text{emb}} \rangle)\right],\end{aligned} \tag{3}$$

where σ is the sigmoid function; ω_{v_i} is sampled from the user-defined negative distribution $P_n(\omega)$; k is the number of the negative samples for each target v_t; $\omega_{v_c}^{\text{nce}}$ and $\omega_{v_t}^{\text{emb}}$ represent embedding vectors for words v_c and v_t respectively. Notice that the context embedding matrix (named "nce") and the target embedding matrix (named "emb") are different.

Similar to Noice Contrastive Estimation, Negative Sampling changes log-likelihood maximization objective to positive/negative samples classification task. As discussed in depth by (Gutmann and Hyvärinen 2010) and (Mikolov and Dean 2013), the choice of negative distribution $P_n(\omega)$ can greatly affect the quality of the embedding result. Therefore, an appropriate negative sampling setting will benefit the system in term of statistical performance and computational cost.

2.2 Graph Embedding

Inspired by the power-law similarity between the vertex frequency distribution in random walks and the word frequency

[1] https://github.com/anonsyuushi/mage

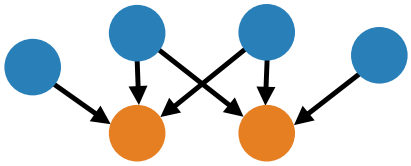


Figure 3: LINE generates context where blue vertices are together, whereas random walk cannot. It is trivial to see the bipartite structure in this figure.

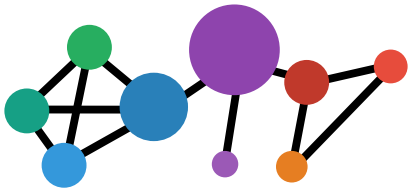
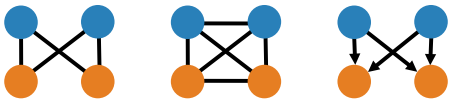


Figure 4: Node2vec emphasizes central "important" vertices.

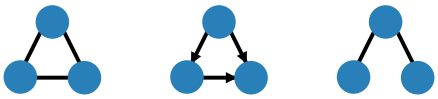
distribution of English text, (Perozzi, Al-Rfou, and Skiena 2014) proposed Deepwalk algorithm to *learn* the latent representations of vertices in a graph. The operation of Deepwalk is based on hierarchial-softmax Skipgram, the only different is artificial "sentences" are generated by performing random walks on the graph. Inheriting the advantages of the Skipgram model, Deepwalk has been successful in various graph-related machine learning tasks. However, the weakness of Deepwalk is in the fact that it ignores the graph's deep structures which cannot be discovered only by random walks.

In one of the researches subsequent to Deepwalk, (Tang et al. 2015) hinted the advantages of structure-aware graph context generation in their citation network experimental results. In their work, Tang et al. proposed LINE, which deploys the second-order proximity into the embedding process. By constructing half the embedding vector with Deepwalk, and the other half with vertices that have second-order proximity, LINE has improved embedding results compare to Deepwalk. However, in the most recent graph embedding research, (Grover and Leskovec 2016) reported some unexpected poor performance from LINE in friendship-based graphs. On the contrary, we noticed that LINE has exceptionally good performance in some graphs that have acyclic structure (e.g. citation networks). We also found the similarity between the definition of second-order proximity in (Tang et al. 2015) and the bipartite motif (Figure 5). From the observations above, we hypothesize that the graph embedding quality can be improved by taking advantage of the statistically significant motifs with the graph.

In 2016, Grover and Leskovec proposed *node2vec*, an algorithm based on biased random walks. This idea of using biased random walks to manipulate graph context generation is aligned perfectly with our idea. However, while node2vec aims to emphasize statistically important *vertices*, our algorithm aims to emphasize the local motif community structure. Another noteworthy graph embedding research is



a) Motifs of size 4: Undirected bipartite, undirected cluster, directed bipartite.



b) Motifs of size 3: Undirected triangle, directed triangle, undirected wedge.

Figure 5: Most common motifs of size 3 and 4.

Planetoid (Yang, Cohen, and Salakhutdinov 2016). While other graph embedding algorithms is unsupervised, Planetoid is a semi-supervised approach. In addition to random walk graph context generation, Planetoid generates another context using the training community labels. Due to the semi-supervised learning procedure of Planetoid, we consider it belongs to another category of algorithm. However, we will discuss the integration of Planetoid to our algorithm in later section.

2.3 Network Motifs

Network motifs are defined to be recurring and regulating patterns in a complex network (Alon 2007). First mentioned by (Milo et al. 2002), network motifs soon became a topic of attention in many research fields especially in biology and sociology (Masoudi-Nejad, Schreiber, and Kashani 2012). Recent researches suggested that network motifs withhold the structural information of the network (Benson, Gleich, and Leskovec 2016; Yanardag and Vishwanathan 2015). Due to the computational complexity of graph isomorphism, in most researches, the studied motifs often has a small size (number of vertices in a motif is less than 4) (Tran et al. 2014). Figure 5 presents some of the most noteworthy motifs.

Definition 1. ***Network motif*** *are patterns of interconnections that recur in many different parts of a network at frequencies much higher than those found in randomized networks (Milo et al. 2002).*

The sentence "a friend of a friend is a friend" makes sense to us in real life. Interestingly, from the graph-theoric point of view, social networks exhibit the similar rule of relationship. In the study conducted by Katherine Faust, the author compared the social interaction networks of animals using the triangle motif (Faust and Skvoretz 2002). Faust pointed out the representation power of various directed triangle motifs to each type of animal social interaction and also suggested that triangle motifs play a central role in a social network's structure. Moreover, studying the Stanford's SNAP network datasets archive (University 2016), we also observe that the social networks have higher triangle count compared to another type of network such as citation network. In conclusion, although there has not been a concrete proof for the importance of network motifs in a complex

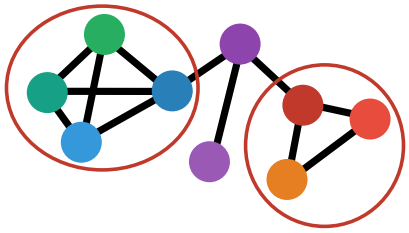


Figure 6: MAGE discovers local motif structure.

network, we cannot ignore the pragmatic representation capability of network motifs.

3 Method

In this section, we present our algorithm in detail. Although there are many different types of motif in a graph, within this paper, we only consider undirected triangle motif (figure 5), undirected wedge motif (figure 5), and undirected bipartite motif (figure 5) for the sake of simplicity. Other types of motif and directed graph extension is discussed in Section 5.

3.1 Motif-Aware Context Generation

Our algorithm is based on the Negative Sampling Skip-gram model (NEG-Skipgram) (Mikolov and Dean 2013). As mentioned in the previous section, Skipgram-based graph embedding *is* word embedding, the difference is the "sentences" are generated by a random process in graph embedding. We first discuss the graph context generation phase, also called the positive sample generation. In order to highlight the local motif structure given a target vertex, we define a motif-biased random walk (or *motif walk* for short). Our motif walk can be viewed as a variant of the Metropolis-Hastings algorithm (Hastings 1970). At each step along the motif walk, an adjacent vertex is chosen by the rejection sampling procedure. The rejection probability depends on the pre-defined states of the Markov Chain. Algorithm 1 describes the motif walk for an arbitrary motif.

The motif statistic of a graph is the key input to our algorithm. As mentioned by (Tran et al. 2014) in their research on motif detection algorithms, motif detection is a costly operation. The exact motif statistic is desirable but costly to obtain. However, we found that only one or two most characteristic motifs are enough to improve the graph embedding quality. Therefore, to keep the topic of the paper focused, we pose an assumption that we know in advance the most statistically significant motif for a given graph.

3.2 Negative Sampling

For each defined motif walk in the previous section, it is trivial to define the inverse version of it. For example, the inverse of the aforementioned triangle motif walk is the wedge motif walk (figure 5). In addition to the traditional unigram-distribution negative sampling, we further emphasize the importance of motif structure by sampling negative samples from the corresponding inverse version of the motif walk that generated graph context. Figure 8 illustrates our reason for motif-aware negative sampling.

Algorithm 1: Motif context sampling $P_m(\omega)$.

Data: Undirected Graph $G = (V, E)$;
Input: start vertex, walk length, motif $M = (V', E')$, bias β;
Result: Vertices in the local motif structure;
vertexList $\longleftarrow$ [start vertex]
current $\longleftarrow$ start vertex
initialize dequeue buffer Q of size $|V'|$
while *vertexList.length* $<$ *walk length* **do**
  candidate $\longleftarrow$ random(neighboor(current))
  Q.append(candidate)
  if *Isomorphic(Q, M)* **then**
    if random $< \beta$ **then**
      vertexList.append(candidate)
    else
      continue
  else
    if random $> \beta$ **then**
      vertexList.append(candidate)
    else
      continue

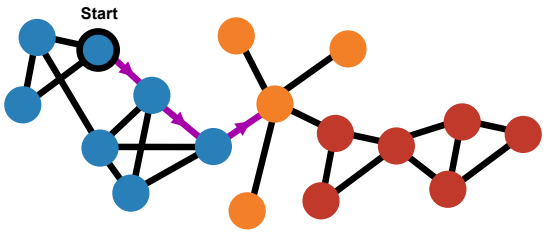


Figure 7: While the positive motif walk generates local context (blue and red), the inverse version walks toward other motif structures.

3.3 MAGE: Motif-Aware Graph Embedding

Figure **??** illustrates our MAGE algorithm. In summary, there are 4 context generation operations: two of them are positive sampling, and the other two are negative sampling. To control the number of samples for each type, we introduce a hyper-parameter α. For example, assume that MAGE generates 6 positive samples and 20 negative samples for each target vertex, with $\alpha = 0.5$, MAGE yields 3 positive samples from motif walk, 3 positive samples from random walk, 10 negative samples from vertex unigram distribution, and 10 negative samples from inverse motif walk. Finally, we define the objective function as follow:

$$\mathcal{L} = (1-\alpha)\mathcal{L}_{\text{random}} + \alpha\mathcal{L}_{\text{motif}}, \tag{4}$$

where $\mathcal{L}_{\text{random}}$ is given in equation 3; $\mathcal{L}_{\text{motif}}$ is defined as:

$$\begin{aligned}\log \Pr(v_c|v_t) = \mathbb{E}_{\omega_{v_c}\sim P_m(\omega)}\left[\log\sigma(\langle\omega_{v_c}^{\text{nce}}, \omega_{v_t}^{\text{emb}}\rangle)\right] \\ + \sum_{i=1}^{k}\mathbb{E}_{\omega_{v_i}\sim(1-P_m(\omega))}\left[\log\sigma(-\langle\omega_{v_i}^{\text{nce}}, \omega_{v_t}^{\text{emb}}\rangle)\right],\end{aligned} \tag{5}$$

3.4 Learning framework

Up to this point of the paper, we have presented the graph context generation procedures. In this section, we present the neural network implemented on Keras (Chollet 2015) (with Theano (Theano Development Team 2016) backend). One advantage of our implementation is that it can run on both CPU and GPU (when available) thanks to the Keras wrapper. Our neural network contains 3 modules. Firstly, there are two embedding matrices of the same shape (`num_vertices`, `emb_dim`). The matrix named "emb" contains embedding vectors for the target vertices, while the other matrix named "nce" contains embedding vectors for the context vertices. Secondly, the computation module computes the dot product of each sample in the mini-batch, applies the sigmoid function, then yields the final logit. The last modules compute the the cross-entropy (De Boer et al. 2005) of samples in the mini-batch and its positive/negative labels. Figure **??** present the block diagram of our learning framework. To optimize the binary cross entropy objective, we use Adam (Kingma and Ba 2014) as the mini-batch gradient optimizer.

4 Experiments and Result

4.1 Experiments design

In our experiments, we compare our approach (MAGE) with Deepwalk (Perozzi, Al-Rfou, and Skiena 2014), LINE (Tang et al. 2015), and node2vec (Grover and Leskovec 2016) by multilabel classification. Table 4.1 gives graph statistics and its corresponding machine learning task.

Blogcatalog (Tang and Liu 2009) is a social network of bloggers. The labels represent a blogger's interest. There are 39 labels in total, each blogger (vertex) has at least one label. Since Blogcatalog is a social network, we chose undirected triangle motif walk for this graph. On the other hand,

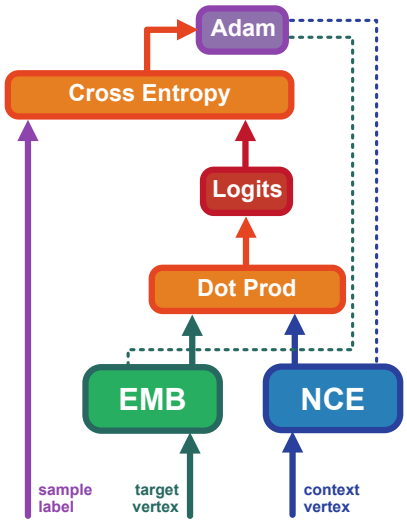


Figure 8: Neural network structure.

DATASET	#CLASSES	#VERTICES	#EDGES
BLOGCATALOG	39	10,312	333,983
CITESEER	6	3,327	4,732
YOUTUBE	47	1,138,499	2,990,443

Table 1: Dataset statistics

Citeseer (Yang, Cohen, and Salakhutdinov 2016) is a citation network. Therefore, we employ the bipartite motif for the context generation of Citeseer. The last graph we consider is Youtube, which is especially large with more than 1 million vertices. Since Youtube is also a friendship network, so we use triangle motif in context generation.

Conventionally, multilabel classification task is constructed as (Perozzi, Al-Rfou, and Skiena 2014) suggested in their paper. The embedding vector for each vertex will be treated as the feature vector for a One-versus-Rest logistic regression with L2 regularization. We split the data into training set and testing set. The result reported here is the weighted f1-macro and f1-micro score with split ratio of 0.5. One exception is with the Youtube network, the split ratio is 0.1 (training data : testing data) due to the size of the network.

In all of our experiments, we use the same hyper-parameters which gave the best results reported in previous researches. For our algorithm, the hyper-parameters are given in table 4.1.

4.2 Experimental Results

Experimental results are presented in table 4.2. For each algorithm presented here as a comparison to our algorithm, we try to use the provided implementation by its authors. The result we obtained here agrees with the result by (Perozzi, Al-Rfou, and Skiena 2014). We confirm the simplicity and effectiveness of Deepwalk. Generally, Deepwalk and Spectral clustering has the advantage of acceptable

HYPER PARAMETER	ABBR.	VALUE
Embedding dimenison	emb_dim	128
Skip window size	window_size	10
Walk length	walk_length	80
#Negative samples	neg_samp	30
#Positive samples	num_skip	5
Motif weight	α	0.2
Motif bias	β	0.9

Table 2: MAGE Hyper-parameters

METHOD	BLOGCATALOG	CITESEER	YOUTUBE
Spectral clustering	0.22	0.30	-
Deepwalk	0.20	0.28	0.29
LINE	0.18	0.35	-
Node2Vec	0.11	0.14	-
MAGE	**0.24**	**0.35**	**0.32**

Table 3: Weighted f1-macro score.

performance without having to many hyper-parameter tuning. Although in our experiment Node2Vec by (Grover and Leskovec 2016) performed poorly compare to other algorithms, it is not an unexpected result. Due to the fact that we did not *search* for the optimal p and q parameters of Node2Vec, we do not expect this algorithm to achieve give a good result. About the YouTube dataset, we can only run Deepwalk (MAGE with no motif) and our algorithm with triangle motif. It is not possible for us to do experiment with Spectral clustering and LINE on this large dataset due to the scalability of the algorithms. In all three datasets, our algorithm MAGE outperforms other prior algorithms. This result showed that by taking advantage of a network's motif structure we can improve the embedding quality.

5 Discussion

In practice, it is important to know the trade-off between the algorithm's complexity and its accuracy. For example, the result reported by (Grover and Leskovec 2016) promised a high-performance model, but it requires searching for the biased random walk hyper-parameters heuristically. In our work, we have proposed a graph embedding algorithm with a clear method to integrate the prior knowledge about the graph's structure. Our preliminary results proved the scalability and accuracy of our algorithm. It can be seen that comparing to Deepwalk or Spectral clustering, our algorithm has better performance with some computational cost and hyper-parameter tunning trade-off. However, the most sensitive hyper-parameter in our algorithm is the network motif choice itself - which can be solved by fast motif detection algorithms (Tran et al. 2014). In conclusion, we believe MAGE is a feasible solution for graph embedding in practice.

As mentioned in the previous sections, there are many ways to extend MAGE. First of all, our current algorithm only addresses graph embedding on undirected graphs. The immediate solution that we employed in this work is to convert any directed graph input to undirected. However, in many real-world networks, the information gain for embedding task on a directed graph may worth the computational cost. Secondly, MAGE can be integrated to the framework proposed by (Yang, Cohen, and Salakhutdinov 2016) to further improve the embedding quality. Thanks to the nature of our implementation, MAGE can be easily integrated into any deep architecture based on Theano framework. To summary, MAGE is a practical embedding algorithm that can be extended in a various ways.

References

Alon, U. 2007. Network motifs: theory and experimental approaches. *Nature Reviews Genetics* 8(6):450–461.

Bader, G. D.; Betel, D.; and Hogue, C. W. 2003. Bind: the biomolecular interaction network database. *Nucleic acids research* 31(1):248–250.

Belkin, M., and Niyogi, P. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, 585–591.

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35(8):1798–1828.

Benson, A. R.; Gleich, D. F.; and Leskovec, J. 2016. Higher-order organization of complex networks. *Science* 353(6295):163–166.

Callon, M. 1986. The sociology of an actor-network: The case of the electric vehicle. In *Mapping the dynamics of science and technology*. Springer. 19–34.

Cao, S.; Lu, W.; and Xu, Q. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 891–900. ACM.

Chollet, F. 2015. keras. `https://github.com/fchollet/keras`.

De Boer, P.-T.; Kroese, D. P.; Mannor, S.; and Rubinstein, R. Y. 2005. A tutorial on the cross-entropy method. *Annals of operations research* 134(1):19–67.

Faust, K., and Skvoretz, J. 2002. Comparing networks across space and time, size and species. *Sociological methodology* 32(1):267–299.

Fodor, I. K. 2002. A survey of dimension reduction techniques.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Gutmann, M., and Hyvärinen, A. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, volume 1, 6.

Hastings, W. K. 1970. Monte carlo sampling methods using markov chains and their applications. *Biometrika* 57(1):97–109.

Jolliffe, I. 2002. *Principal component analysis*. Wiley Online Library.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Masoudi-Nejad, A.; Schreiber, F.; and Kashani, Z. 2012. Building blocks of biological networks: a review on major network motif discovery algorithms. *IET systems biology* 6(5):164–174.

Mikolov, T., and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*.

Milo, R.; Shen-Orr, S.; Itzkovitz, S.; Kashtan, N.; Chklovskii, D.; and Alon, U. 2002. Network motifs: simple building blocks of complex networks. *Science* 298(5594):824–827.

Morin, F., and Bengio, Y. 2005. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, 246–252. Citeseer.

Newman, M. 2010. *Networks: an introduction*. Oxford university press.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710. ACM.

Tang, L., and Liu, H. 2009. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 817–826. ACM.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, 1067–1077. ACM.

Tenenbaum, J. B.; De Silva, V.; and Langford, J. C. 2000. A global geometric framework for nonlinear dimensionality reduction. *science* 290(5500):2319–2323.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688.

Tran, N. T. L.; Mohan, S.; Xu, Z.; and Huang, C.-H. 2014. Current innovations and future challenges of network motif detection. *Briefings in bioinformatics* bbu021.

University, S. 2016. Snap: Stanford network analysis project. `snap.stanford.edu`. Accessed: 2016-08-24.

Van Den Heuvel, M. P., and Pol, H. E. H. 2010. Exploring the brain network: a review on resting-state fmri functional connectivity. *European Neuropsychopharmacology* 20(8):519–534.

Yanardag, P., and Vishwanathan, S. 2015. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1365–1374. ACM.

Yang, Z.; Cohen, W. W.; and Salakhutdinov, R. 2016. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on Machine Learning*. ICML.