Motif-Aware Graph Embedding

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

Abstract

Given a large complex graph, how can we have a 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 a feasible solution 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 showed that our algorithm yields more accurate embedding results compared to 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 (**?**). The same approach for structural analysis can be found in other fields such as chemistry (**?**), physics (**?**), and sociology (**?**). 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*.

Learning a high-quality latent representation is a challenging task. Traditionally, dimensionality reduction technique such as PCA (**?**), CCA (**?**), and IsoMap (**?**) are used. Although these aforementioned techniques have a profound theoretical background (**?**), they are impractical due to computational drawbacks. To address the dimensionality problem, Peperozi et al. proposed Deepwalk - a Skipgram-based algorithm for graph embedding (**?**). 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 (**?**) to learn a vertex’s latent representation. Following Deepwalk, other Skipgram-based graph embedding algorithms which aim to improve embedding quality were proposed (**?**; **?**; **?**; **?**). 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. The 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 et al. (**?**), 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 (**?**; **?**). 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 (?) framework. The implementation and experimental results are available on Github. (TODO: ADD FOOTNOTE).

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 (?). 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 et al. 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 (?) or Noise Contrastive estimation (?). In their landmark paper, Mikolov et al. also proposed *Negative Sampling* - a simplified version of noise contrastive estimation (?). The log-likelihood of under negative sampling scheme is given by:

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

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; $\omega^{\text{nce}}_{v_c}$ and $\omega^{\text{emb}}_{v_t}$ 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 et al. and Mikolov et al. , 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

Traditionally, dimension-reduction techniques can be adopted to produce latent representation for vertices in a graph. Linear algebra approaches such as Principal Component Analysis (PCA) (?) or Non-negative Matrix Factorization has a strong theoretical background but great computation drawbacks. (TODO: ASK NUKUI TO REWRITE)

Inspired by the power-law similarity between the vertex frequency distribution in random walks and the word frequency distribution of English text, Peperozi et al. 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 walk 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. hinted the advantages of structure-aware graph context generation in their citation network experimental results (?). In their work, Tang et al. proposed LINE, which consider 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 et al. 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 (?) and the bipartite motif (Figure ??). 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 et al. proposed *node2vec*, an algorithm based on biased random walks. This idea of using biased random walk 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 Planetoid (?) proposed in 2015 by Yang et al. 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 (?). First mentioned by Milo et al. (?), network motifs soon became a topic of attention in many research fields especially in biology and sociology (?). Recent researches suggested that network motifs withhold the structural information of the network (?; ?; ?). 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) (?). Figure **??** presents some of the most noteworthy motifs.

Definition 1. *Network motif is a small subgraph that is statistically significant. (TODO: FILL THIS)*

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 pointed out the representation power of different 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 (?), we also observe that the social networks have higher triangle count compared to other 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 network, we cannot ignore the pragmatic representation capability of network motifs.

3 Methods and Experiments

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 **??**), undirected wedge motif (figure **??**), and undirected bipartite motif (figure **??**) 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 Skipgram model (NEG-Skipgram) (?). 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 Monte Carlo Markov Chain (?), whose number of states equals the motif size. 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 arbitary motif.

Definition 2. *Monte Carlo Markov Chain is awesome. (TODO: FILL THIS)*

Algorithm 1: Motif walk based of MCMC (TODO: FILL THIS)

```
Data: Undirected Graph $G = (V, E)$;
Input: length, MCMC chain
Result: List containing vertices in the motif walk;
initialization
while Not full yet do
    Walk around
    if walk fail then
        walk again
    else
        continue to walk
```

The motif statistic of a graph is the key input to our algorithm. As mentioned by Tran et al. in their research on motif detection algorithms, motif detection is a costly operation. The exact motif statistic is desiable, 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 trival to define the inverse version of it. For example, the inverse of the aforementioned triangle motif walk is the wedge motif walk (figure **??**). 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 **??** illustrates our reason for motif-aware negative sampling.

3.3 MAGE: Motif-Aware Graph Embedding

Figure **??** and algorithm 2 illustrate 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 positve samples and 20 negative samples for each target vertex, with $\alpha = 0.5$, MAGE yeilds 3 postive samples from motif walk, 3 postive 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:

Algorithm 2: MAGE

```
Data: Undirected Graph $G = (V, E)$;
Input: length, MCMC chain
Result: List containing vertices in the motif walk;
initialization
while Not full yet do
    Walk around
    if walk fail then
        walk again
    else
        continue to walk
```

DATASET	#CLASSES	#VERTICES	#EDGES	TASK
BLOGCATALOG	39	10,312	333,983	MC
WIKIPEDIA	40	4,777	184,812	MC
CITESEER	6	3,327	4,732	MC
FACEBOOK	-	4,039	88,234	LP

Table 1: Datasets and machine learning task

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

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

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

3.4 Experiments design

In our experiments, we compare our approach (MAGE) with Deepwalk (**?**), LINE (**?**), and node2vec (**?**) in two machine learning task: Multilabel classification (abbr. MC) and link prediction (abbr. LP). Table 3.4 gives graph statistics and its corresponding machine learning task.

The Skipgram model proposed by Mikolov et al. (**?**) is a powerful model in natural language processing. Under the Distributional Hypothesis (**?**), the Skipgram model maximizes the occurent probability of context words given a target word.