Network Embedding and Change Modeling in Dynamic Heterogeneous Networks

Ranran Bian
University of Auckland and Pingar
rbia002@aucklanduni.ac.nz

Yun Sing Koh
University of Auckland
ykoh@cs.auckland.ac.nz

Gillian Dobbie
University of Auckland
g.dobbie@auckland.ac.nz

Anna Divoli
Pingar
anna.divoli@pingar.com

ABSTRACT

Network embedding learns the vector representations of nodes. Most real world networks are heterogeneous and evolve over time. There are, however, no network embedding approaches designed for dynamic heterogeneous networks so far. Addressing this research gap is beneficial for analyzing and mining real world networks. We develop a novel representation learning method, change2vec, which considers a dynamic heterogeneous network as snapshots of networks with different time stamps. Instead of processing the whole network at each time stamp, change2vec models changes between two consecutive static networks by capturing newly-added and deleted nodes with their neighbour nodes as well as newly-formed or deleted edges that caused core structural changes known as triad closure or open processes. Change2vec leverages metapath based node embedding and change modeling to preserve both heterogeneous and dynamic features of a network. Experimental results show that change2vec outperforms two state-of-the-art methods in terms of clustering performance and efficiency.

CCS CONCEPTS

• **Computing methodologies** → **Learning latent representations**; • **Theory of computation** → **Dynamic graph algorithms**;

KEYWORDS

Dynamic Heterogeneous Networks; Vectors; Change Modeling

ACM Reference Format:
Ranran Bian, Yun Sing Koh, Gillian Dobbie, and Anna Divoli. 2019. Network Embedding and Change Modeling in Dynamic Heterogeneous Networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331273

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331273

1 INTRODUCTION

Most real world networks are heterogeneous and evolve constantly. Typical examples of these networks (graphs) are social networks and biological networks, where multiple types of objects (nodes) and relationships (edges) exist. Recently, many network embedding methods [2, 6, 10, 11] have been proposed to learn low-dimensional vector representation of a node. To the best of our knowledge, none of these approaches were tailored for dynamic heterogeneous networks. Unlike static homogeneous graph embedding, the techniques for dynamic heterogeneous graphs need to be scalable and incremental to deal with the dynamic changes efficiently. This makes most of the existing graph embedding methods, which suffer from single type oriented problems, unsuitable.

The intuition behind our proposed method, change2vec, is to impose triad (i.e., a set of three nodes) to capture the structural changes of dynamic heterogeneous networks. A triad is a fundamental block of a network [9] and can be a closed or open triad. In a closed triad, there exists a relationship between any two objects. Whereas in an open triad, there are only two edges, indicating that two nodes in the triad do not form a relationship. To capture formation and evolution of dynamic heterogeneous networks, we model how an open triad transitions into a closed triad through the *triad closure process* and how a closed triad becomes an open triad through the *triad open process* [4]. Figure 1 presents a dynamic heterogeneous social network where the circles represent users while squares denote topics that users subscribe to. The network contains two types of relationships: friendship between two users represented by a solid line and subscription between a user and a topic denoted by a dashed line. The transition from time step t to $t+1$ shows that the network introduces two new edges (edge e_{BC} connecting B and C and edge e_{BT} connecting B and T), which results in two triad closure processes. As the network evolves from time step $t+1$ to $t+2$, a triad open process occurs due to disappearance of the edge e_{AC} between A and C. To generate up-to-date latent vectors for all nodes in the network, traditional static network embedding methods such as Node2vec [3], DeepWalk [6] and Metapath2vec++ [2] need to process the whole network and approximate latent vectors for all nodes at each time step, which is intractable for large dynamic heterogeneous networks. Instead, change2vec captures network structural changes by only updating vector representations for nodes that are involved in triad closure or open processes during two consecutive time steps. In general, it is natural for networks to evolve smoothly over time, instead

of volatile restructures between each time step [10], which justifies the usefulness and efficiency of change2vec in this kind of networks. To illustrate, with the network in Figure 1, at time step $t+1$, change2vec only updates vector representations for B, C and T from time step t, at time step $t+2$, only vector representations for A and C from time step $t+1$ are updated.

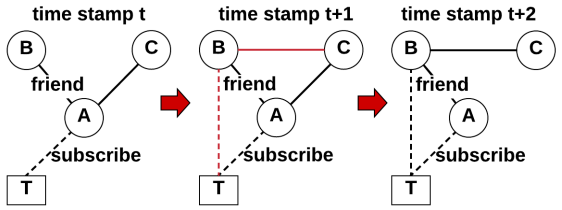

Figure 1: An illustrative example of dynamic social network

Metapath is defined in a heterogeneous network as a sequence of relationships between different object types, which describes a new composite relationship between its starting type and ending type [2]. The metapath scheme provides a useful change capture mechanism by incorporating the heterogeneous nature of the network. The dynamic social network in Figure 1 contains two types of objects: users (U) and topics (P), which can form meaningful metapath schemes such as "UUU" and "UPU" for modeling triad closure and open processes.

Our research contributions are as follows: (1) We propose the method, change2vec, which incorporates both dynamic and heterogeneous features of a given network for the representation learning task; (2) When applying learned embedding vectors of nodes, change2vec outperforms two state-of-the-art network embedding methods in terms of Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) scores; (3) Compared to a static heterogeneous network embedding method, change2vec is faster for networks that are constantly evolving.

2 RELATED WORK

In recent years, developing network embedding methods has attracted a considerable amount of research interest. These methods fall into three major categories based on the network type that the method was designed for. (1) Network representation learning algorithms that were designed for static homogeneous networks, such as DeepWalk [6] and node2vec [3]. The word2vec [5] framework uses the skip-gram architecture to learn the distributed representations of words in natural language. Both DeepWalk and node2vec were built on word2vec and leveraged the two-layer neural network structure in word2vec. (2) Representation learning models which were designed for static heterogeneous networks, e.g., metapath2vec++ [2] and HNE [1]. Similar to DeepWalk and node2vec, metapath2vec++ was also inspired by word2vec's skip-gram model to perform node embeddings. The algorithm captures heterogeneous structural correlations among multiple types of nodes from metapath-guided random walks. (3) Unlike the first two categories, where the network and the learned vector representations are fixed, there also exist network embedding methods for dynamic homogeneous networks [10, 11]. DynamicTriad [10] employs the triad closure process [4] to learn dynamic latent representations of nodes.

Overall, our work furthers this line of research by developing the change2vec framework to efficiently capture dynamic and heterogeneous features simultaneously. This cannot be achieved by previous methods, for example, DynamicTriad is incapable of distinguishing different types of objects, which could result in incorrect metapaths being adopted for modeling triad closure process, e.g., using "PPP" in the network in Figure 1. change2vec overcomes DynamicTriad's limitation by introducing heterogeneous metapath-guided representation learning and change modeling.

3 OUR APPROACH

Change2vec processes a dynamic heterogeneous network as snapshots of networks with different time stamps. Instead of reconstructing the network in each time step, we identify a set of changed nodes between two consecutive time steps. Our framework aims to learn the embedding vectors of nodes by modeling the dynamic and heterogeneous properties of a network.

Definition 3.1. **A Dynamic Heterogeneous Network** is an undirected graph $G = (V, E, T)$ in which each node v and each edge e is associated with their mapping functions $\phi(v) : V \rightarrow T_V$ and $\varphi(e) : E \rightarrow T_E$ respectively. T_V and T_E represent the sets of object and relationship types, where $|T_V| + |T_E| > 2$. In addition, between two consecutive time stamps t and $t+1$, $|V^t| \neq |V^{t+1}|$ or $|E^t| \neq |E^{t+1}|$, where $|V^t|$ and $|V^{t+1}|$ denote the number of objects at time stamps t and $t+1$ respectively, $|E^t|$ and $|E^{t+1}|$ represent the number of relationships at time stamps t and $t+1$ respectively. T_V and T_E remain unchanged across all time stamps t.

For example, the network in Figure 1 can be represented with users (U), topics (P) as nodes, wherein edges denote the friendship ($U-U$), subscription ($U-P$) relationships. As the network evolves, the set of nodes stays the same while the sets of edges change across different time steps. We now formalize the representation learning problem in dynamic heterogeneous networks as follows.

Problem 1. ***Dynamic Heterogeneous Network Representation Learning*** *Given a dynamic heterogeneous network G, which is represented as snapshots of a static heterogeneous network G^t at different time steps t ($t = 1...z$), the task is to learn the dynamic d-dimensional vector representations $X^t \in \mathbb{R}^{|V^t| \times d}$, $d \ll |V^t|$ that are able to capture the structural relationships among the objects in G^t at the time step t.*

The output of the problem is the low-dimensional matrix X^t, with the v^{th} row being a d-dimensional vector of node v at the time step t. Representations of different types of nodes in G are mapped into the same latent space. The learned embedding vectors of nodes can be used as feature inputs of various heterogeneous network mining tasks.

Figure 2 shows the design of change2vec. The input of the Initial Node Embedder component is G^1, which is the snapshot of a given dynamic heterogeneous network G at the initial time stamp $t = 1$. This component leverages the metapath2vec++ algorithm [2] to use metapath based random walks as input training data for learning vector representations of all nodes that exist in G^1.

When G evolves from time step t to $t+1$, change2vec models the differences between the network snapshots of the two consecutive time steps, G^t and G^{t+1}, by producing a set of changed nodes

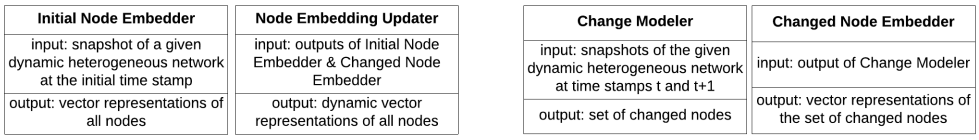

Figure 2: Framework of change2vec

V_{change}. Nodes that are involved in the following four scenarios are included in V_{change}:

(1) newly-added nodes and their one-hop neighbour nodes.

$$V_{change} = \{v, u : v, u \in V^{t+1}, v \notin V^t, (v,u) \in E^{t+1}\} \quad (1)$$

(2) deleted nodes and their one-hop neighbour nodes.

$$V_{change} = \{v, u : v, u \in V^{t}, v \notin V^{t+1}, (v,u) \in E^{t}\} \quad (2)$$

(3) newly-formed edges which caused triad closure processes.

$$V_{change} = \{v, u : (v,u) \notin E^t, \{(v,u) \in E^{t+1}, (v,w) \in E^{t+1}, (u,w) \in E^{t+1}\} \neq \emptyset\} \quad (3)$$

(4) deleted edges which caused triad open processes.

$$V_{change} = \{v, u : (v,u) \in E^t, (v,u) \notin E^{t+1}, \{(v,u) \in E^t, (v,w) \in E^t, (u,w) \in E^t\} \neq \emptyset, \{(v,u) \in E^{t+1}, (v,w) \in E^{t+1}, (u,w) \in E^{t+1}\} = \emptyset\} \quad (4)$$

Notice that the four scenarios describe the complete modeling scope of change2vec and if a node is involved in more than one of the scenarios, the node is included in V_{change} once only. After the Change Modeler component obtains a set of changed nodes, the set is used as input data by the Changed Node Embedder component. Similar to the Initial Node Embedder component, the Changed Node Embedder component leverages the metapath2vec++ algorithm to generate embedding vectors of nodes. The differences between the two components are the input and output data. Instead of processing all the nodes in the network, the Changed Node Embedder component only works on the changed nodes and produces vector representations of them. The main functionality of the Node Embedding Updater component is to generate up-to-date vector representations of all nodes at each time stamp $t+1$ from t ($t \geq 1$). Basically, X^t is updated to X^{t+1} by adding embedding vectors of the newly-added nodes ($\{v : v \in V^{t+1}, v \notin V^t\}$), removing vector representations of the deleted nodes ($\{v : v \in V^t, v \notin V^{t+1}\}$), replacing latent vectors of the existing but changed nodes ($\{v : v \in V^t, v \in V^{t+1}, X_v^t \neq X_v^{t+1}\}$).

Usually, the objective of network embedding methods is to maximize a given network probability in terms of neighbourhoods of nodes [2, 3, 5, 6]. The change2vec technique inherits the common objective and maximizes the probability of the network $G = (V, E, T)$ on each different time stamp t as follows:

$$\underset{\theta}{\text{argmax}} \sum_{v \in V^t} \sum_{tp \in T_V} \sum_{c_{tp} \in N_{tp}^t(v)} \log p(c_{tp}|v;\theta) \quad (5)$$

where $N_{tp}^t(v)$ represents node v's one-hop neighbourhood with the tp^{th} type of nodes at the time step t, $p(c_{tp}|v;\theta)$ denotes the conditional probability of having a context node c_{tp} given a node v and is commonly defined as a softmax function [2], that is:

$$p(c_{tp}|v;\theta) = \frac{e^{X_{c_{tp}}^t \cdot X_v^t}}{\sum_{u_{tp} \in V_{tp}^t} e^{X_{u_{tp}}^t \cdot X_v^t}} \quad (6)$$

where X_v^t is the v^{th} row of X^t, denoting the vector representation for node v at the time stamp t, V_{tp}^t is the node set of type tp in the network at t. For illustration, consider the social network in Figure 1, the neighbourhood of user node A can be structurally close to other users (e.g., B & C), topics (e.g., T). The Initial Node Embedder and Changed Node Embedder components leverage the same heterogeneous negative sampling [2, 5] and stochastic gradient descent algorithm used in metapath2vec++ [2] for optimizing the derived objective function.

Overall, change2vec preserves the heterogeneous nature of a network through metapath based schemes for representation learning as well as triad open and closure modeling. The Initial Node Embedder and Changed Node Embedder components use metapath guided random walks as input training data to generate vector representations of multiple types of nodes. The Change Modeler component follows specific metapaths for modeling triad open and closure processes, for example, in the network in Figure 1, triads formed by "UUU" and "UPU" can be specified for modeling.

4 EXPERIMENTAL EVALUATIONS

We apply change2vec on Digital Bibliographic Library Project (DBLP) data sets to evaluate its performance when compared to state-of-the-art methods. Metapath2vec++[2] and DynamicTriad [10] were selected as baselines because they have been empirically shown to outperform other techniques in their respective fields. All experiments were performed on virtual machines with 4 vCPUs and 16GB RAM. Our source code is available for download[1].

Table 1: Properties of DBLP data sets

Data Set Year	Sequence	Authors	Venues	Papers
2010	1st	140351	108	155117
2011	2nd	147024	110	163207
2013	3rd	185987	115	212953

For our experiments, we used three sequential time stamps of DBLP data sets extracted from AMiner[2]. Similar to Dong et al. [2], we processed each data set and have only kept entries that matched the top 20 conferences in 8 research categories[3] found on Google Scholar[4]. These research categories are used as ground

[1] https://github.com/Change2vec/change2vec.git

[2] https://aminer.org/citation

[3] 1. Computational Linguistics, 2. Computer Graphics, 3. Computer Networks & Wireless Communication, 4. Computer Vision & Pattern Recognition, 5. Computing Systems, 6. Databases & Information Systems, 7. Human Computer Interaction, and 8. Theoretical Computer Science.

[4] https://scholar.google.com/citations?view_op=top_venues&hl=en&vq=eng. Accessed on January, 2019.

truth information when we perform clustering tasks on venues (V) and authors (A). At each time step, we map an author to a particular research category (cluster) if over half of the papers they publish are from conferences of that research category. The properties for each processed data set is presented in Table 1.

Similar to metapath2vec++, change2vec adopts the metapath "AVA" to guide random walks. "AAA" and "AVA" are the metapaths specified in change2vec for modeling triad open and closure processes. For metapath2vec++ and change2vec, we used the optimal parameters as proposed by the original authors [2] with the number of walks set to 1000, length of walks to 100. For all three methods, we set the number of clusters to 8 and dimensions to 128. Due to the nondeterministic nature of random walk based schemes in metapath2vec++ and change2vec, we performed 30 runs of each evaluation and present the average and standard deviation results.

We performed clustering on venues and authors separately as proxy tasks to evaluate the accuracy of learned vectors. More accurate vectors should generate better clustering results. The embedding vectors of author and venue nodes produced by each method were used as feature inputs by the k-means clustering algorithm. From the resulting clusters, we then evaluated the performance of each heuristic with the NMI [7] and ARI [8] scores which show how close the resulting clusters are to the author and venue ground truths. Both scores are in the value range of [0,1] with 1 indicating that the results are identical to the ground truth clusters.

Table 2 presents the venue and author clustering NMI values and time taken for each method to process the data set with the specific time stamp. For example, change2vec obtained a high venue clustering NMI value of 0.81 and an author clustering value of 0.73 for the 2013 time stamp. The best result in each set of data is highlighted in bold. Since change2vec and metapath2vec++ both follow the same steps for the first time stamp, we can see that the values for them are identical (as underlined in the table). However, from 2011 onwards, we notice a decrease in run time as well as higher NMI values for venue clusters. DynamicTriad was considerably faster than change2vec and metapath2vec++. However, since it was designed for dynamic homogeneous networks and cannot differentiate between author and venue nodes as separate node types, the method consistently produced low NMI values.

Table 2: Run time and node clustering results (NMI) of each time stamp

Method	Year	Venue	σ(Venue)	Author	σ(Author)	Time(mins)
DynamicTriad	2010	0.18	N/A	0.04	N/A	**7**
Metapath2vec++	2010	**0.80**	0.05	**0.79**	0.04	20
Change2vec	2010	**0.80**	0.05	**0.79**	0.04	20
DynamicTriad	2011	0.31	N/A	0.07	N/A	**7**
Metapath2vec++	2011	0.68	0.05	**0.71**	0.03	20
Change2vec	2011	**0.71**	0.03	**0.71**	0.05	10
DynamicTriad	2013	0.33	N/A	0.14	N/A	**8**
Metapath2vec++	2013	0.73	0.04	**0.74**	0.05	25
Change2vec	2013	**0.81**	0.02	0.73	0.04	18

The Adjusted Rand Index [8] is another commonly adopted clustering evaluation metric. Table 3 highlights change2vec's effectiveness as seen in 2011 and 2013 data sets where it produced considerably higher ARI values for venue clusters. Similar to NMI, DynamicTriad was shown to produce the lowest cluster similarity scores due to the method being agnostic to node types.

Table 3: Node clustering results (ARI) of each time stamp

Method	Year	Venue	σ(Venue)	Author	σ(Author)
DynamicTriad	2010	0.10	N/A	0.03	N/A
Metapath2vec++	2010	**0.69**	0.04	**0.80**	0.02
Change2vec	2010	**0.69**	0.04	**0.80**	0.02
DynamicTriad	2011	0.03	N/A	0.05	N/A
Metapath2vec++	2011	0.46	0.02	0.64	0.04
Change2vec	2011	**0.56**	0.01	**0.67**	0.02
DynamicTriad	2013	0.24	N/A	0.02	N/A
Metapath2vec++	2013	0.57	0.04	0.74	0.03
Change2vec	2013	**0.81**	0.03	**0.76**	0.03

Due to space constraints, data sets and results of some other experiments are placed on our source code site[1]. Overall, experimental evaluations demonstrate that despite the usage of metapath2vec++, change2vec still managed to improve metapath2vec++'s embedding accuracy and efficiency by focusing only on the changed nodes between two consecutive time stamps.

5 CONCLUSION AND FUTURE WORK

We propose a framework, change2vec, to capture changes and heterogeneous characteristics in a dynamic heterogeneous network. The framework captures structural changes between two consecutive network snapshots by modeling the triad open and closure processes and describes such changes with up-to-date node embedding vectors. Experimental results show that change2vec outperforms two baselines in terms of clustering performance and efficiency. Future work includes automatically learning useful metapaths in diverse types of dynamic heterogeneous networks.

ACKNOWLEDGMENTS

We sincerely thank all reviewers for their comments, New Zealand Callaghan Innovation and Pingar for funding the R&D Student Fellowship Grant (PTERN1502), and Tobey Sheng-Yen Hung's assistance in the research.

REFERENCES

[1] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C. Aggarwal, and Thomas S. Huang. 2015. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21st ACM SIGKDD*. 119–128.
[2] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. 2017. Metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD*. 135–144.
[3] Aditya Grover and Jure Leskovec. 2016. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD*. 855–864.
[4] Hong Huang, Jie Tang, Lu Liu, JarDer Luo, and Xiaoming Fu. 2015. Triadic closure pattern analysis and prediction in social networks. *IEEE Transactions on Knowledge and Data Engineering* 27, 12 (2015), 3374–3389.
[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems* 26 (2013), 3111–3119.
[6] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD*. 701–710.
[7] Yizhou Sun, Jiawei Han, Peixiang Zhao, Zhijun Yin, Hong Cheng, and Tianyi Wu. 2009. RankClus: Integrating clustering with ranking for heterogeneous information network analysis. In *Proceedings of the 12th EDBT*. 565–576.
[8] Silke Wagner and Dorothea Wagner. 2007. *Comparing clusterings: an overview.*
[9] Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature* 393, 6684 (1998), 440.
[10] Lekui Zhou, Yang Yang, Xiang Ren, Fei Wu, and Yueting Zhuang. 2018. Dynamic network embedding by modeling triadic closure process. In *Proceedings of the 32nd AAAI*.
[11] Linhong Zhu, Dong Guo, Junming Yin, Greg Ver Steeg, and Aram Galstyan. 2016. Scalable temporal latent space inference for link prediction in dynamic social networks. *IEEE Transactions on Knowledge and Data Engineering* 28, 10 (2016), 2765 – 2777.