A Comparative Study of Representation Learning Techniques for Dynamic Networks

Carlos Ortega Vázquez[1](✉), Sandra Mitrović[1], Jochen De Weerdt[1], and Seppe vanden Broucke[1,2]

[1] Research Center for Information Systems Engineering (LIRIS), KU Leuven, Leuven, Belgium
carloseduardo.ortegavazquez@kuleuven.be

[2] Department of Business Informatics and Operations Management, Ghent University, Ghent, Belgium

Abstract. Representation Learning in dynamic networks has gained increasingly more attention due to its promising applicability. In the literature, we can find two popular approaches that have been adapted to dynamic networks: random-walk based techniques and graph-autoencoders. Despite the popularity, no work has compared them in well-know datasets. We fill this gap by using two link prediction settings that evaluate the techniques. We find standard node2vec, a random-walk method, outperforms the graph-autoencoders.

Keywords: Dynamic networks · Representation learning

1 Introduction

Network analysis has increasingly gained attention both in academia and industry because it offers a framework that analyzes interrelationships within natural structures: we find applications in churn prediction [7,20], crime detection [26,27], recommendation systems [14]. However, network analysis traditionally requires extensive preprocessing: data analysts have relied on handmade feature engineering based on expert knowledge or summary statistics (e.g. clustering coefficients) [17]. Despite the popularity of ad-hoc feature engineering, it lacks flexibility and requires extensive domain knowledge [12,14,20].

One response to traditional feature engineering is representation learning (RL); it is sometimes referred as feature learning. RL aims at finding a low-dimensional representation or embedding of the data so further downstream tasks become more automatic [3]. However, most early techniques in RL can only handle static networks [12,16,22,28]. In contrast, a real-world network displays dynamic processes that changes its topological structure [25]. Recent techniques for RL in dynamic graphs have relied on random walks [8,21,24], autoencoder

[10,11], or matrix factorization in a lesser extent [18]. However, to the best of our knowledge, no work has compared graph autoencoder and random-walk techniques. Furthermore, no work has compared RL techniques for link prediction both interpolation (i.e. finding missing links) and extrapolation (i.e. predict future networks) settings.

Our contribution is twofold for RL in dynamic networks. First, we propose an experimental setup that evaluates RL techniques in how effective they recover missing links (i.e. interpolation setting) and predict future version of graphs (i.e. extrapolation setting). Second, we use a bayesian approach of word2vec [1] for representation learning in graphs. A bayesian approach to RL provides insights due to its probabilistic nature [2]. We evaluated the RL techniques in two well-known datasets in the literature of dynamic networks: Facebook forum and Enron.

2 Related Work

The literature on RL for graphs have diversified into several lines of research [6,29]: for network transductive tasks, dynamic RL exploits the topological evolution [10,20,24] while inductive RL leverages extra information for unseen nodes [13,27]. We can categorize RL techniques in dynamic networks regarding time granularity [9]: some methods handle discrete time and others, with continuous time evolution [21]. We focus on the former as more works have been developed in that line of research.

We consider two main types of RL techniques in dynamic networks that are relevant for this study: random walk and graph-autoencoder approaches. On the one hand, the random walk techniques, related to shallow embedding methods [13], learn the network embedding based on the nodes that co-occur on random walks. One strong merit of random-walk techniques is the use of a stochastic similarity measure (e.g. co-occurrence in random walks) which leads to lower complexity compared to deep learning approaches. However, these techniques require fine-tuning of the random walks. On the other hand, graph-autoencoder techniques leverage the adjacency matrix that captures non-linear relationship in the node neighbourhood [28]: similar neighbourhood leads to similar embedding. Compared to the random-walk approach, graph-autoencoders can reconstruct the whole graph since they learn from its adjacency representation; dependence on the adjacency matrix also constrains its applicability in large real-world networks [20].

Both approaches originally handle static networks [12,28] so recent works developed extensions for dynamic networks. Most of the techniques depends either on reusing parameters in each time-step [10,11,19] or aligning the static embeddings [8,24]: However, these efforts lack theoretical foundation. The Bayesian framework considers a prior probability distribution that can allow a smoother drift of the network embedding across time. Bayesian word embeddings have been explored in NLP [1,2,5]: Bayesian word embeddings offer noise robustness by time slices and uncertainty measurement via density. Despite its theoretical benefits, no work in RL for networks has explored this direction. We implement the approach taken in [1] as a first step towards filling the gap for Bayesian RL techniques in dynamic networks.

3 Methodology and Experimental Setup

Current literature considers both interpolation and extrapolation settings, although not at the same extent. Most of the literature focuses on the interpolation setting, also known as the completion problem [9]. The completion problem for the missing links in a graph is posed as: given $\mathcal{G} = \{G_1, .., G_T\}$, we want to predict links among the graphs. The RL techniques for static graphs mostly rely on the single time-step of the interpolation setup [12]; however, it is also possible to use interpolation for dynamic graphs [8]. In contrast, extrapolation requires predicting links from further time-steps beyond G_T; prediction of a graph as a set of links can be considered as extrapolation setting. Few works address the extrapolation setup [10,11,30] since it can be more challenging. For this work, we consider both interpolation and extrapolation setting.

The literature lacks of comparison of RL techniques in dynamic networks that are based on orthogonally different approaches. For this study, we compare two approaches for RL techniques: random-walk and graph-autoencoder. We use the standard node2vec [12], and a network adaptation of the dynamic Bayesian word embeddings [1]. From the graph-autoencoder techniques, we consider DynGEM [11] and dyngraph2vecAE (dynae) [10]. Both techniques generate embeddings from the bottleneck layer in the autoencoder. The techniques optimize the embedding in order to preserve the local and global structure: the relevant hyperparameters that represent the trade-off between preserving local and global structure are α and β. The key difference between these graph-autoencoder is that DynGEM exploits the adjacency matrix from a previous time-step while dynae can leverage more previous time-steps.

Our adaptation for Bayesian embeddings (dynbae) requires random walks, instead of text, as input so we must sample the network as in node2vec: thus, the same sampling parameters are present (e.g. return parameter p and in-out parameter q). This dynamic approach is based on the previous Bayesian Skip-gram model [2], but a Kalman Filter [15] is added to allow dynamic processes. The Bayesian skip-gram model uses a Gaussian prior to formulate the posterior probability of the embedding. The posterior is computed through Variational Bayes [4] which differs from the standard Skip-gram optimization. The result is that the technique can map nodes into probability densities instead of point estimates.

Two network datasets are used for the experimental setup; all of them are available on Network Repository (http://networkrepository.com) [23] where open access is given to different dynamic networks. Table 1 describes some descriptive statistics for each of the datasets. The Facebook Forum dataset display high density, considered as a dense network, which can be reflected in the high average degree of 15.68. The other datasets can be considered as sparse

Table 1. Descriptive statistics

	Facebook forum	Enron
Number of nodes	899	135
Number of edges	7046	135
Average degree	15.68	10.96
Average clustering coef.	0.0637	0.4889
Number of connected components	1	3
Degree assortativity coef.	−0.1083	−0.1490
Density	0.01746	0.0818
Number of snapshots	6	8
Size interval of snapshots	Month	2 Months

networks given their low density. The Enron dataset is also a dense network but it contains substantially less nodes and edges. However, the dataset shows a higher average clustering coefficient.

First, we need to split the datasets into different snapshots. The choice of the time frame size follows as in [8]. For each snapshot, we have an edge list that represents a network. The test snapshot in both interpolation and extrapolation settings derive from the last time-step G_T. For the interpolation approach, we randomly divide into two sets for the subsequent downstream task: 70% of the edges are used for training and 30%, for test. For the extrapolation approach, we use the whole G_T for evaluation. Additionally, we sample non-edges as many as the edges so the datasets are balanced in all snapshots. We only use previously seen training nodes so we extract a subgraph from G_T that complies with this requirement. All training snapshots share information of the training nodes even if they are not active in a particular snapshot (i.e. a node without any link to others). We follow the approach in [12] to get the edge features from the node embeddings: the Hadamard operator is used to combine a couple of node embeddings into one vector for representing edges. At the end, we obtain a matrix of edge features: for the extrapolation setting, embeddings for each snapshot are stacked vertically while for the interpolation setting, embeddings are stacked horizontally. Subsequently, a classifier can learn from the edge features for predicting in the corresponding test set if the two nodes hold an edge. Three well-known classifiers in link prediction are used: Logistic Regression, Random Forest, and Gradient Boosting.

We perform a hyperparameter tuning on the training snapshots for the RL techniques based on the link prediction performance. For the random-walk techniques, the grid search is as follows: $p \in \{0.25, 0.5, 0.75, 1\}$, $q \in \{0.1, 0.5, 1, 2, 5, 10, 100\}$. For the graph-autoencoder, the grid search is as follows: $\alpha \in \{10^{-6}, 10^{-5}\}$, and $\beta \in \{2, 5\}$. All experiments, including the data, can be found in https://github.com/CarlosOrtegaV/dyn-bae.

4 Results

Table 2 and 3 contain, for each classifier and RL technique, the highest Area Under the Receiver Operating Curve (AUC) scores across hyperparameter combinations with its corresponding Average Precision (AP). We can observe that the extrapolation setting poses a more challenging task since RL techniques have a lower AUC score than in the interpolation setting; Facebook forum dataset also obtained lower AUC scores because of the higher number of nodes and edges. Node2vec consistently outperforms all other RL techniques in both interpolation and extrapolation setting. Despite its simpler structure compared to dyngraph2vecAE (dynae), dynGEM reaches the second place among the RL techniques. The dynamic Bayesian node2vec (dynbae) scores low compared to the standard node2vec. Figure 1 and 2 display the variability of the AUC scores across hyperparameters for two classifiers in the Facebook forum dataset. Interestingly, the graph-autoencoder techniques have higher variability in the extrapolation setting compared to the interpolation counterpart.

Table 2. The AUC score & average precision in interpolation setting

Dataset	Classifier	node2vec		DynGEM		dynae		dynbae	
		AUC	AP	AUC	AP	AUC	AP	AUC	AP
Facebook forum	Logistic Reg.	**0.851**	0.865	0.754	0.728	0.585	0.580	0.672	0.672
	Random Forest	**0.883**	0.891	0.746	0.761	0.604	0.582	0.713	0.667
	Gradient Boosting	**0.832**	0.844	0.724	0.749	0.591	0.612	0.714	0.701
Enron employees	Logistic Reg.	**0.880**	0.890	0.870	0.850	0.615	0.599	0.653	0.662
	Random Forest	**0.866**	0.841	0.823	0.767	0.649	0.602	0.620	0.582
	Gradient Boosting	**0.867**	0.850	0.818	0.770	0.655	0.607	0.641	0.630

Table 3. The AUC score & AP in extrapolation setting

Dataset	Classifier	node2vec		DynGEM		dynae		dynbae	
		AUC	AP	AUC	AP	AUC	AP	AUC	AP
Facebook forum	Logistic Reg.	**0.750**	0.804	0.701	0.725	0.618	0.585	0.564	0.545
	Random Forest	**0.748**	0.800	0.500	0.502	0.555	0.540	0.585	0.563
	Gradient Boosting	**0.746**	0.792	0.519	0.534	0.530	0.512	0.566	0.560
Enron employees	Logistic Reg.	**0.846**	0.860	0.810	0.820	0.582	0.599	0.601	0.593
	Random Forest	**0.864**	0.872	0.657	0.668	0.531	0.543	0.585	0.605
	Gradient Boosting	**0.856**	0.862	0.630	0.621	0.537	0.548	0.585	0.595

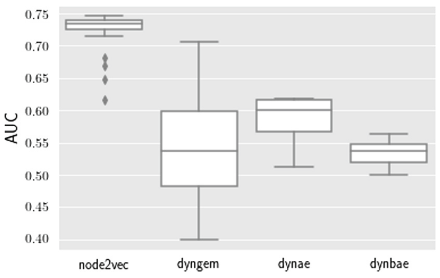

Fig. 1. Extrapolation using Logistic Regression

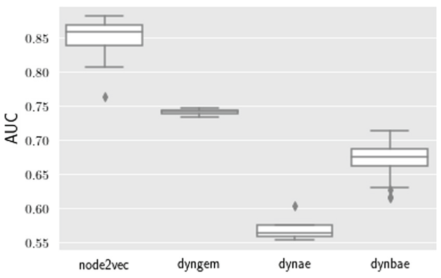

Fig. 2. Interpolation using Random Forest

5 Conclusions

This work compares orthogonally different techniques for RL in dynamic graphs regarding link prediction. The comparison consists in two settings that are not often presented together in the literature. We find that the random-walk techniques, particularly the standard node2vec, outperform the graph-autoencoder techniques. Furthermore, the Bayesian adaptation of node2vec performs poorly even though it uses the same similarity measure.

References

1. Bamler, R., Mandt, S.: Dynamic word embeddings. In: Proceedings of the 34th International Conference on Machine Learning, ICML 2017, pp. 380–389. PMLR (2017)
2. Barkan, O.: Bayesian neural word embedding. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI 2017, pp. 3135–3143. AAAI Press (2017)
3. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013)
4. Bishop, C.M.: Pattern Recognition and Machine Learning. Information Science and Statistics. Springer, New York (2006)
5. Bražinskas, A., Havrylov, S., Titov, I.: Embedding words as distributions with a Bayesian skip-gram model. In: Proceedings of the 27th International Conference on Computational Linguistics. Association for Computational Linguistics (2018)
6. Cai, H., Zheng, V.W., Chang, K.C.C.: A comprehensive survey of graph embedding: problems, techniques, and applications. IEEE Trans. Knowl. Data Eng. **30**(9), 1616–1637 (2018)
7. Dasgupta, K., Singh, R., Viswanathan, B., Chakraborty, D., Mukherjea, S., Nanavati, A.A., Joshi, A.: Social ties and their relevance to churn in mobile telecom networks. In: Proceedings of the 11th International Conference on Extending Database Technology: Advances in Database Technology, EDBT 2008, pp. 668–677. ACM, New York (2008)

8. De Winter, S., Decuypere, T., Mitrović, S., Baesens, B., De Weerdt, J.: Combining temporal aspects of dynamic networks with Node2Vec for a more efficient dynamic link prediction. In: 2018 IEEE/ACM International Conference on Advances in Social Analysis and Mining (ASONAM), pp. 1234–1241. IEEE (2018)
9. Goel, R., Jain, K., Kobyzev, I., Sethi, A., Forsyth, P., Poupart, P.: Relational representation learning for dynamic (knowledge) graphs: a survey. arXiv.org (2019). http://search.proquest.com/docview/2231646581/
10. Goyal, P., Chhetri, S.R., Canedo, A.: dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. Knowl.-Based Syst. **187**, 104816 (2020)
11. Goyal, P., Kamra, N., He, X., Liu, Y.: DynGEM: deep embedding method for dynamic graphs. arXiv preprint arXiv:1805.11273 (2018)
12. Grover, A., Leskovec, J.: Node2Vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM (2016)
13. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, pp. 1024–1034 (2017)
14. Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: methods and applications. arXiv preprint arXiv:1709.05584 (2017)
15. Kalman, R.E.: A new approach to linear filtering and prediction problems. J. Basic Eng. **82**(1), 35–45 (1960)
16. Kipf, T., Welling, M.: Variational graph auto-encoders (2016). arXiv.org
17. Liben-Nowell, D., Kleinberg, J.: The link-prediction problem for social networks. J. Am. Soc. Inform. Sci. Technol. **58**(7), 1019–1031 (2007)
18. Ma, X., Sun, P., Wang, Y.: Graph regularized nonnegative matrix factorization for temporal link prediction in dynamic networks. Phys. A **496**, 121–136 (2018)
19. Mahdavi, S., Khoshraftar, S., An, A.: dynnode2vec: scalable dynamic network embedding. In: 2018 IEEE International Conference on Big Data (Big Data), pp. 3762–3765. IEEE (2018)
20. Mitrović, S., Baesens, B., Lemahieu, W., Weerdt, J.D.: tcc2vec: RFM-informed representation learning on call graphs for churn prediction. Inf. Sci. (2019)
21. Nguyen, G.H., Lee, J.B., Rossi, R.A., Ahmed, N.K., Koh, E., Kim, S.: Continuous-time dynamic network embeddings. In: Companion Proceedings of the The Web Conference 2018, WWW 2018, pp. 969–976. International World Wide Web Conferences Steering Committee (2018)
22. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710. ACM (2014)
23. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015). URL http://networkrepository.com
24. Singer, U., Guy, I., Radinsky, K.: Node embedding over temporal graphs. arXiv preprint arXiv:1903.08889 (2019)
25. Trivedi, R., Farajtabar, M., Biswal, P., Zha, H.: Representation learning over dynamic graphs. arXiv preprint arXiv:1803.04051 (2018)
26. Troncoso, F., Weber, R.: A novel approach to detect associations in criminal networks. Decis. Support Syst. **128**, 113–159 (2019)
27. Van Belle, R., Mitrović, S., De Weerdt, J.: Representation learning in graphs for credit card fraud detection. In: ECML PKDD 2019 Workshops. Springer (2019)

28. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016, 13–17 August 2016, pp. 1225–1234. ACM (2016)
29. Wang, Y., Yao, Y.: A brief review of network embedding. Big Data Min. Anal. **2**(1), 35–47 (2019)
30. Yang, Y., Ren, X., Wu, F., Zhuang, Y.: Dynamic network embedding by modeling triadic closure process. In: Thirty-Second AAAI Conference On Artificial Intelligence, pp. 571–578. AAAI (2018)