Temporally Evolving Models for Dynamic Networks

1 Introduction

The research of complex networks and large graphs generated a wide variety of stochastic graph models that try to capture the properties of these complex systems [2, 4, 5, 8, 7]. Most of the well-known models can describe a static graph extracted from a real-world dataset. They are capable of generating an ensemble of graphs, in which all graph instances are similar in terms of specific statistics to the original one. For example, models that capture the power-law degree distribution of real-world networks such as the Albert-Barabási one are dynamic but do not attempt to model the actual temporal evolution of large graphs. Our goal is to give temporal stochastic graph model for the temporal dynamics of these complex systems.

Our models address the link prediction problem introduced by Liben-Nowell and Kleinberg, in a *temporal* setting. More specifically, we try to predict accurately each new link in the graph at the time when it is created in the network. This experimental setting is similar to our method introduced for recommender systems [9]. In Section 3 we explain this setup in case of dynamic graphs. For baseline algorithm, we apply online matrix factorization [6, 10, 11] on temporal network data (see Section 4).

Various node centrality measures capture the "importance" of a node by using the structural properties of the graph [3]. While these metrics are widely investigated, few is known about the evolution of graph centrality in temporal graphs. In our work, we investigate the applicability of node centrality metrics in temporal graphs by examining their temporal behavior and computational complexity. We also use these metrics as side features in our matrix factorization models.

2 Datasets

We perform our experiments on a variety of Twitter, Last.FM and the Koblenz Collection data sets. Twitter is a highly temporal social system with dynamically evolving communities, a mass community with an ever changing graph structure. Our goal is to investigate the dynamics of this community that can be best described as an evolving complex network. The first issue on Twitter is to define a graph that well describes the system. One can define several networks in Twitter. We focus on follower networks, retweet cascades, root retweet networks, @mention graphs and @reply graphs. Our Twitter datasets contain tweets around events of global relevance, including Euromaidan, Maidan, Olympic, Occupy, Yosoy, 15o, 20n, MH17 and Ayotzinapa. Last.fm is an online service in music based social networking. It collects "scrobbles"—a word by Last.fm meaning that when you listen to a song, the name of the song is added to your music profile. We selected a representative, well-connected, yet anonymous random sample of users. These users had their location in UK with age between 14 and 50, inclusive. Only public scrobbles with a daily average activity between 5 and 500 and at least 10 friends that meet the first four conditions. Other temporal network datasets are available in the Koblenz Network Collection for Twitter mentions, arXiv author network, Digg, Flickr and YouTube graphs.

3 Experimental setting and evaluation metrics

In the dynamic link prediction task, we have to rank the best K links for the given node at the given time instance. Our dataset contains records $< u, v, t >$ of links between users u and v that appear at time t. Our goal is to recommend new links for user u at time t with the constraint that there is only a single link that appears at the given time t. This means that we have to maximize the rank of the given link in the actual

predicted list of links. A time sensitive or online link prediction system should retrain its model after each and every training record $< u, v, t >$. We have to generate new top-K recommendation list for *every* single record. The online top-K task is hence different from the standard recommender evaluation settings, since there is always a single neighbor only in the ground truth and the goal is to aggregate the rank of these single neighbors over the entire testing period. For our task, we need carefully selected quality metrics that we describe next. We use our full dataset both for training and testing. We iterate on the records one by one in temporal order. For a given record $< u, v, t >$, we allow the recommender algorithm to use full of the data *before* t in question for training and require a ranked top list of possible neighbors as output. We evaluate the given single actual neighbor v in question against the recommended top list of length K.

For measuring the accuracy of predicting a new link, we face the difficulty that only a single correct answer exists at the given time and the next edge arrives to be tested against an updated model. We propose DCG [9], a modified version of NDCG, the prefered model for batch top-K recommendation [1]. DCG is a slowly decreasing function of the rank and hence measures how close the actual new link appears in the top list.

4 Dynamic adjacency matrix factorization

Batch modeling algorithms may iterate several times over the graph until convergence. In our temporal setting, the model needs to be retrained after each new event and hence reiterations over the earlier parts of the data is ruled out.

In this section, we give an online factorization method for the graph adjacency matrix. Matrix factorization yields a low-rank approximation of the adjacency matrix with entries for non-edges filled with values that we consider an indication for the edge to appear. Links for a given node are predicted by taking the largest values in the corresponding row or column. In our algorithm, we allow a single iteration over the training data only, and this single iteration processes the events in the order of time. We use each record in the dataset as a positive training instance and generate negative training instances by selecting random items for each positive record. We use the regularized matrix factorization method of [12], and use the k-factor model for prediction.

Temporal modeling methods seem more restricted than those that may iterate over the data set several times and one would expect inferior quality by the online methods. Online methods however have the advantage of giving much more emphasis on recent events that we empirically verify in our research.

4.1 Centrality measures as side information

Matrix factorization algorithm may use so-called side information associated with the rows and columns of the matrix. In our experiment we use centrality measures as side information associated with the nodes. We may use directed centrality with different values for rows and columns of the same node. We compare the following metrics in the temporal setting of dynamic networks based on [3]: Harmonic Centrality, PageRank, HITS and SALSA.

5 Conclusion and Further Work

In this paper, we analyze the dynamic network data as a stream of nodes and edges. To predict link formation, the regularized matrix factorization model is proposed. Different centrality measures are used with online computation over the graph stream to identify the evolution of centrality. As the main lesson learned, we show how recent results in recommender systems can be deployed for the analysis of complex networks.

References

[1] A. Al-Maskari, M. Sanderson, and P. Clough. The relationship between ir effectiveness measures and user satisfaction. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 773–774. ACM, 2007.

[2] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[3] P. Boldi and S. Vigna. Axioms for centrality. *arXiv preprint arXiv:1308.2140*, 2013.

[4] B. Bollobás, O. Riordan, J. Spencer, G. Tusnády, et al. The degree sequence of a scale-free random graph process. *Random Structures & Algorithms*, 18(3):279–290, 2001.

[5] V. Csiszár, P. Hussami, J. Komlós, T. F. Móri, L. Rejtő, and G. Tusnády. When the degree sequence is a sufficient statistic. *Acta Mathematica Hungarica*, 134(1-2):45–53, 2012.

[6] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[7] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 57–65. IEEE, 2000.

[8] M. Kurucz and A. Benczúr. Geographically Organized Small Communities and the Hardness of Clustering Social Networks. *Annals of Information Systems*, pages 177–199, 2010.

[9] R. Pálovics, A. A. Benczúr, L. Kocsis, T. Kiss, and E. Frigó. Exploiting temporal influence in online recommendation. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 273–280. ACM, 2014.

[10] S. Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.

[11] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[12] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Investigation of various matrix factorization methods for large recommender systems. In *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, pages 1–8. ACM, 2008.