Games of Friends: a game-theoretical approach for link prediction in online social networks

Giovanni Zappella
Dipartimento di Matematica "F.Enriques"
Università degli studi di Milano
Milan, Italy
giovanni.zappella@unimi.it

Alexandros Karatzoglou
Telefonica Research
Barcelona, Spain
alexk@tid.es

Linas Baltrunas
Telefonica Research
Barcelona, Spain
linas@tid.es

Abstract

Online Social Networks (OSN) have enriched the social lives of millions of users. Discovering new friends in the social network is valuable both for the user and for the health of OSN since users with more friends engage longer and more often with the site. The simplest way to formalize friendship recommendation is to cast the problem as a link prediction problem in the social graph. In this work we introduce a game-theoretical approach based on the Graph Transduction Game. It scales with ease beyond 13 million of users and was tested on a real world data from *Tuenti* OSN. We utilize the social graph and several other graphs that naturally arise in *Tuenti* such as the wall-to-wall post graph. We compare our approach to standard local measures and demonstrate a significant performance benefit in terms of mean average precision and reciprocal rank.

Introduction

Online Social Networks (OSN) have become an indispensable part of our lives facilitating communication and connecting people on a massive scale. Popular social networks such as Twitter, LinkedIn and Tuenti have attracted millions of users while still growing in a staggering pace. In these networks friendships are often started on-line and only later-on "graduate" to the real world.

Fundamental to all on-line social networks is the goal to effectively model the friendship patterns between users. Using these models OSN's can facilitate link formation and friendships among users and thus increase the value of the site for it's members. Users will subsequently spend more time on the network and thus increase the site's traffic and the potential for monetization. Moreover more users will recognize the value of the network and will join the site. Typically friendship suggestion systems in OSN's are responsible for a large fraction of the created edges in the social graph.

The task of friendship suggestion is cast as a link prediction problem e.g. (Sarkar, Chakrabarti, and Jordan 2012), (Backstrom and Leskovec 2011). That is given a snapshot of a large social graph S and a user i find the users that i has the biggest probability of forming a connection with. This problem can also be seen as a ranking problem in that we are looking for an optimal ranking of the nodes (users) in the graph with respect to connection formation potential to user i.

Motivation While link formation is a well studied problem it is particularly hard as OSN tend to exhibit very heterogeneous behavior. The amount of friends among users can differ by orders of magnitude and social graphs tend to evolve very fast in fact the number of edges on the Tuenti social graph increases by 0.5% a day. Moreover link prediction in an OSN is particularly hard as even the most connected users only have edges to a tiny fraction of the network making the data extremely sparse.

Multigraph While the social graph created by the friendship connections of the users is central to all link prediction models, OSN's allow for the interaction of users in many diverse ways. Users post messages in their friends wall's or co-tag pictures with other users etc. These interactions form a new set of graphs that exhibit different properties to the social graph that is typically unweighted and of binary nature (i.e. a user is a friend or not). Moreover these activity graphs can provide information both about the strength of a friendship between two users and subsequently about the chance that they will be forming a friendship e.g. it is safe to assume that users that co-tag a picture or attend the same event have a significant probability of meeting outside the OSN and thus also forming a friendship in the OSN. The set of all these additional graphs form a *multigraph* over the users, i.e., the users are connected with more then one type of edges.

Target In summary, our aim is to model diverse user interaction data along with the social graph in order to create a link prediction model for the purpose of friendship recommendation. We ultimately aim to have this system serve recommendation on a real world OSN (*Tuenti.com*) and thus strive to satisfy constrains imposed on such a system by the requirements of an live on-line system under heavy use e.g. training the model several times a day, compact representation and fast prediction time in order to serve several requests per second.

Related Work

Link prediction The problem of link recommendation is a well studied problem, with several methods developed over the past years. For example, (Lü and Zhou 2011) provide an excellent summary of many link prediction techniques. One of the most well known techniques are the Random Walks with Restart (Tong, Faloutsos, and Pan 2006) where the probability to end up in a node after a random walk on the graph is used for prediction. The scalable version of this method requires a partitioning of the graph and this is a notoriously difficult problem particularly with industry scale social graphs. Moreover the methods deals with a single graph. A supervised version of the algorithm was recently introduced whereby features on the nodes and on the edges can be used to guide the random walk (Backstrom and Leskovec 2011). This work focuses on node features though it might be possible to potentially use it for features on edges. It does require a rather complex computation of the affinity matrix within the optimization procedure.

Latent Factor Models Factor models have been used before for link prediction, in (Miller, Griffiths, and Jordan 2010),(Zhu 2012) a probabilistic approaches are used to optimize over the user factors U. A similar method is introduced in (Menon and Elkan 2011) using a *AUC* loss function along with a gradient descent optimization method. Thought effective on a single graph these approaches do not seem to adapt well to the large scale data that we are dealing with (i.e. the Tuenti Multigraph 10^7 users, 10^9 edges). All of these methods focus on a single graph.

Local Methods Factor Models and Random Walks exploit the global structure of the graph to provide a similarity measure between nodes. Local methods such as Common Neighbors (Newman 2001) are based on computing local statistics and measures on the node of interest. To avoid biases introduced by the power law distribution of edges in social networks often local measures are discounted by a factor proportional to the in-degree of the shared nodes.The Adamic-Adar measure (Adamic and Adar 2003) discounts common connections by the log of their in-degree, while other similarity measures used in link prediction include the Salton measure (Salton and McGill 1986) and the the Jaccard coefficient. The method introduced in this paper can be seen as a local method and therefore enjoys benefits of these methods, such as scalability.

A Game Theoretic Approach

In this section we describe the game introduced in (Erdem and Pelillo 2011) for nodes classification on graphs. There two main ideas behind the casting of our link-prediction problem in this particular game:

- A social network is a group of (supposed rational) agents those take decision for themself, exactly like in every non-cooperative game
- Homophily is widespread in OSN from many points of view, so we assume that users of the social network want to behave exactly like their friends

Within this game we do not try to find the most influential users or to introduce other complex models, we just expect that users will be uniform to their friends. Roughly speaking, we expect they will try to became friends to the users that are friends of their friends (as in most of the heuristics for friends recommendation).
We have to keep in mind that, since this game is non-cooperative, each player maximizes its own payoff disregarding what it can do to maximize the sum of utilities of all the players (the so-called social welfare). We now present the problem in a more formal way.

In the Graph Transduction Game (later called GTG), the graph topology is known in advance and each node of the graph is a player of the game. Each player interacts only with its neighbors on the graph.

The game is defined as $\Gamma = (I, S, \pi)$, where $I = \{1, 2, ..., n\}$ is the set of players, $S = \times_{i \in I} S_i$ is the joint strategy space, and $\pi : S \to \mathbb{R}^n$ is the combined payoff function which assigns a real valued payoff $\pi_i(s) \in \mathbb{R}$ to each pure strategy profile $s \in S$ and player $i \in I$. In our particular case each player has only two strategies: "friend" or "non-friend" of a given user u.

A mixed strategy of player $i \in I$ is a probability distribution x over the set of the pure strategies of i. Each pure strategy k corresponds to a mixed strategy where all the strategies but the k-th one have probability equals to zero.

We define the utility function of the player i as

$$u_i(s) = \sum_{s \in S} x(s)\pi_i(s)$$

where $x(s)$ is the probability of s.

We assume the payoff associated to each player is additively separable (this will be clear in the following lines). This makes GTG a member of a subclass of the multi-player games called polymatrix games (Howson 1972).

For a pure strategy profile $s = (s_1, s_2, ...s_n) \in S$, the payoff function of every player $i \in I$ is:

$$\pi_i(s) = \sum_{j \sim i} w_{ij} \mathbb{I}_{\{s_i = s_j\}}$$

where $i \sim j$ means that i and j are neighbors, $\mathbb{I}$ is the indicator function and w_{ij} is the weight on the edge between i and j. Please note that we do not make any assumption on the weights.
This can be re-written in matrix form as

$$\pi_i(s) = \sum_{j \sim i} A_{ij}(s_i, s_j)$$

where $A_{ij} \in \mathbb{R}^{c \times c}$ is the partial payoff matrix between i and j, defined as $A_{ij} = I_c \times w_{ij}$ Here I_c is the identity matrix of size c and $A_{ij}(x, y)$ represent the element of A_{ij} at row x and column y. The utility function of each player $i \in I_U$ can be re-written as follows:

$$\begin{aligned} u_i(s) &= \textstyle\sum_{i \sim j} x_i^T A_{ij} x_j \\ &= \textstyle\sum_{i \sim j} w_{ij} x_i^T x_j \\ &= \textstyle\sum_{i \sim j} w_{ij} \sum_{k=1}^{c} x_{i_k} x_{j_k} \end{aligned}$$

where k is an action selected from the player's set. Since the utility function of each player is linear, it is easy to see that players can achieve their maximum payoff using pure strategies.

In a non-cooperative game, a vector of strategies S_{NE} is said to be a (pure strategies) Nash Equilibrium, if $\forall i \in I$, $\forall s'_i \in S_i : s'_i \neq s_i \in S_{NE}$, we have that

$$u_i(s_i, S^{-i}_{NE}) \geq u_i(s'_i, S^{-i}_{NE})$$

where $u_i(s_i, S^{-i})$ is the strategy configuration S except the i-th one, that is replaced by s_i. In practice, no player i will change its strategy s_i to an alternative strategy improving its payoff.

The problem of finding a Nash Equilibrium in the general case is a PPAD-Complete (Daskalakis, Goldberg, and Papadimitriou 2006). Clearly, we are not going to compute a solution for an NP-Hard problem on a large scale dataset; in order to rank the potential friends, we do not really need to compute it.
Every time we select a user u, we have a strategy profile given by past choices of the users in the social network (being friend or not friend of somebody-else). Usually, this strategy profile is not a Nash Equilibrium, so some users should be able to improve their payoff just changing strategy (i.e. becoming friend of another user). We use an algorithm called GTG-Rank that ranks the users to suggest to the user u according to how much a user $i \neq u$ can improve his payoff by changing its strategy.
In practice, every time we consider a user u, we have a profile of strategies (friend/non-friend) and we rank the user in a certain set B using the improvement between their current payoff and the payoff they will get changing their choice.
Since this is a local method, it is particularly efficient and can be easily parallelized. Predictions for each node are independent and only information about a small portion of the graph is needed. This is particularly suitable for computations on huge datasets those are often spread across multiple machines.

In the case where we have to rank a set $B \subseteq V$ of nodes, considered a fixed node i; the expected running time is $O(|B|d + |B| \log |B|)$, where d is the average degree. On web-scale networks d is often in the order of $\log |V|$ and $|B| \ll |V|$.

In the next section we will explain how to run GTG-Rank on data from Tuenti.com and all the details about the graph and the payoffs (those at the moment are completely generic).

Experiments

In this section we present a collection of experiments that evaluate the proposed algorithm. We first give a detailed description of the dataset and experimental setup that are used in the experiments. Then, we investigate the impact of different graphs on the results for the friends recommendation task. Finally, we evaluate the recommendation performance of $GTG - Rank$ compared with some well-known baselines. We want to emphasize that we focused out attention on extremely scalable algorithms, such algorithms should perform the prediction for all users in matter of hours on a 16 cores and 64GB RAM machine. We tested also different alternatives such Random Walk with Restarts (Tong, Faloutsos, and Pan 2006), but unfortunately we could not include the results as the Pegasus (Kang, Tsourakakis, and Faloutsos 2009) implementation took about 1h20m per user to compute the predictions on an Hadoop cluster with 12 nodes.

The experiments were designed to address the following research questions:

- Is additional information beyond social network on the users' behavior can be useful to predict friendships creation?
- What kind of information is more useful for the friendship recommendation task?
- Does the presented algorithm scales equally good and perform better than then well known scalable heuristics such e.g. the Adamic-Adar score?

Dataset

We ran our experiments on an on-line real-world commercial social network: Tuenti.com. Tuenti, often referred as the Spanish Facebook, is a Spanish social network with about 13M users. The platform includes all the common features of modern on-line social networks: status updates, wall posting, private messages, photo sharing, link sharing, events invitations, etc...

The dataset was created with an anonymized dump of database in July 2012, and includes four different graphs:

- **Social graph**: an unweighted undirected graph representing friendships among users. An edge of this graph is created every time a user "add a friend" on the social network. The total number of users in this graph (and in the network) is 12,996,961 and the number of edges is 818,234,588. This graph was contains all the edges created from the beginning of Tuenti.
- **Co-tagging graph**: an undirected weighted graph of users those has been tagged in the same photo. If one user is in this graph, it means that he has been tagged at least once in one photo, and the weights on the edges are the count of how many time two users have been tagged in the same photos (it is a sum over photos). Due to the huge amount of pictures uploaded each day on the website (it is in the millions range), the graphs has been generated using all the photo uploaded from the beginning of January 2012 to the end of June 2012. In order to reduce the noise we discarded all the photos with more than 100 people tagged in them. The number of nodes is 1,407,407 the number of edges is 124,615,559.
- **Wall-To-Wall graph**: an undirected weighted graph of user posting on other users' wall. Wall posting is a generic public communication from user to user, and it is also used to share photo, video or link with a particular friend. Users can post only on friends' wall, so this is a subset of the social graph. The number of users is 7,343,198 the number of edges is 31,501,921.

- **Invitations graph**: an undirected weighted graph of users those attend to the same events (the invitation to the event has been accepted). Also in this case, in order to reduce the noise, we discarded all the events with more than 100 attending users. The number of users is 1,679,156 the number of edges is 74,208,780.

The number of users in the social network is much higher than the number of users those are actively using advanced functionality. We also computed the co-occurrence of users in different graphs: 1,337,854 users are in both wall-to-wall graph and co-tag graph, 1,480,431 users are in both invitations graph and wall-to-wall, but only 435,412 users are in both invitations graph and co-tag graph. Since our algorithm works on a single weighted graph, we have to "merge" multiple graph in an undirected weighted one. What we did for this preliminary experiments was a simple sum over a regularized weights as follow [1]:

$$\bar{w}_{ij} = \sum_{G} \frac{w_{ij}^{G}}{\sum_{(k,z)\in E_G} w_{kz}^{G}}$$

where $\bar{w}_{ij}$ is the weight used in our resulting graph and w_{ij}^{G} is the weight of the edge (i, j) in the graph G.

We would like to exploit all this information about the "strength" of the connections in order to improve our predictions, so these weights are used in the GTG's payoff function defined above.

Evaluation Protocol

We temporally order the data and split it in train and test set. All the edges (friendships, wall posts, photo co-tag, etc.) created before April 2012 (included) have been used as training set, the friendships created in May and June 2012 have been used as test set.

As reported before, Tuenti has more than 13M but the total number of users in the dataset is 12,996,961 because some of them has been created after April 2012, and each graph in the dataset has a different number of users due to different activity of the users in the network. All the data reported in the previous descriptions are referred to the train set and are summarized in Table .

We made a comparison with two different test sets:

- **ALL USERS** is a set containing edges created by 4,448,744 users, those have added at least one friend between May 1st and June 30th. Basically, this set contains all the users we those consider "active" users. It is important since we want to improve the experience for all the users of our OSN.
- **250K ACTIVE** is a set containing all the edges created between May 1st and June 30th, by 250,000 randomly picked users those have connections in all the graphs (social, wall-to-wall, co-tag and invitations). This is a subset of ALL USERS, with rich extra information.

[1]We tested also more complicated methods, mostly involving linear regression, but the final results of these (slower) approaches are never better than those obtained in this way

Please note that now Tuenti's registration is open to all, but at the time this dataset has been created it was an invitation-only social network.

For the testing procedure we adopt a similar strategy to (Cremonesi, Koren, and Turrin 2010). As we do not have negative feedback for the implicit data we have to emulate it in order to produce a ranking. We first randomly select 100 Tuenti's users those are not friend of the current one. We predict the scores for the test users for user u_i then we form a ranked list by ordering all the items according to their predicted scores.

We use the Average Precision evaluation measure given by:

$$AP = \sum_{k}^{r} \frac{rel(\kappa) \times P@\kappa}{K} \tag{1}$$

$$rel(\kappa) = \begin{cases} 1 & \text{if item at position } \kappa \text{ is relevant} \\ 0 & \text{if item at position } \kappa \text{ is not relevant} \end{cases} \tag{2}$$

We compute the Mean Average Precision (MAP) by computing the average precision over each user and average by the number of users $MAP = \sum_{i}^{n} \frac{AP(u_i)}{n}$. MAP is a list-wise measure that emphasizes ranking relevant items higher. This is particularly relevant in recommendation where users tend to notice only the top 5-10 recommended items in a list. This is particularly suitable for real-world applications.

We also report MRR (Mean Reciprocal Rank) values, since most of the time the rank of the first relevant result is extremely important, given that users typically pay attention to the first 5-10 items in a recommendation list.

The notion of a relevant item clearly depends on the underlying data. In our application domain, we denote a user to be relevant for the current user user if they are connected in the social graph. Although this is a very simple approach and one could use communication counts, tags, etc.. in a more sophisticated manner. As such this is something we will return to in the future.

Algorithms

In our experiments we compare $GTG - Rank$ with two well-know algorithms for link prediction those can scale on large scale data.

- **Random**: it's a random predictor. This toy baseline is useful in order to understand the absolute quality of our predictions.
- **Common neighbors (CN)**: the similarity score assigned to each couple of users by this algorithm is the count of the common neighbors.

$$CNScore(i, j) = \sum_{k\in Users} \mathbb{I}\{k \sim j, k \sim i\}$$

This simply means that if we have a lot of friends in common, we are likely to be (or become) friends.

- **Adamic-Adar (AA)**: is a variant of the common neighbor count, where users are weighted using their degree.

	Social graph	Co-tag graph	Wall-To-Wall graph	Invitations graph
Users	12996961	1407407	7343198	1679156
Edges	818234588	124615559	31501921	74208780
Avg. Degree	62.96	88.54	18.76	44.19
Nodes in biggest CC	12909834	1398887	7028785	1656500
Time range start	10/9/2005	1/1/2012	3/17/2010	11/25/2007
Time range end	4/30/2012	4/30/2012	4/30/2012	4/30/2012
Number of days	2395	120	775	1618

Table 1: Graph's information. Some of there values were calculated using WebGraph (Boldi and Vigna 2004).

SOCIAL	RANDOM		GTG-RANK		AA		CN	
	MAP	MRR	MAP	MRR	MAP	MRR	MAP	MRR
ALL USERS	0.1172	0.1469	**0.8107**	**0.6895**	0.7875	0.6665	0.7863	0.6656
250k ACTIVE			0.5395	0.7644	**0.5412**	**0.7670**	0.5403	0.7663
SOCIAL+W2W	RANDOM		GTG-RANK		AA		CN	
	MAP	MRR	MAP	MRR	MAP	MRR	MAP	MRR
ALL USERS	0.1172	0.1469	**0.8123**	**0.6910**	0.7869	0.6665	0.7863	0.6657
250k ACTIVE			**0.5475**	**0.7745**	0.5414	0.7717	0.5402	0.7664
SOCIAL+CO-TAG	RANDOM		GTG-RANK		AA		CN	
	MAP	MRR	MAP	MRR	MAP	MRR	MAP	MRR
ALL USERS	0.1172	0.1469	**0.8193**	**0.6967**	0.7874	0.6665	0.7863	0.6656
250k ACTIVE			**0.572**	**0.7887**	0.5447	0.7706	0.5432	0.7692
SOCIAL+INV	RANDOM		GTG-RANK		AA		CN	
	MAP	MRR	MAP	MRR	MAP	MRR	MAP	MRR
ALL USERS	0.1172	0.1469	**0.8211**	**0.6977**	0.7875	0.6665	0.7863	0.6657
250k ACTIVE			**0.5590**	**0.7865**	0.5437	0.7703	0.5425	0.7693
ALL GRAPHS	RANDOM		GTG-RANK		AA		CN	
	MAP	MRR	MAP	MRR	MAP	MRR	MAP	MRR
ALL USERS	0.1172	0.1469	**0.8204**	**0.6976**	0.7846	0.6664	0.7835	0.6655
250k ACTIVE			**0.5835**	**0.7937**	0.5456	0.7717	0.5442	0.7705

Table 2: Results averaged on 10 runs, variance is not reported since it was insignificant in our experiments and did not influence our findings nor our conclusions. In the first column on the left are reported the names of the graphs used for the experiments (in bold) and the test sets used (not in bold). For each predictor we report MAP and MRR.

$$AAScore(i,j) = \sum_{\{k:k\sim j, k\sim i\}} \frac{1}{\log(deg(k))}$$

This means that connections to users with few friends are more important that connections to users with a huge amount of friends in the social network. This is particularly true in practice: let suppose that a user is friend of the famous Spanish chef Ferran Adrià. Since Adrià is a VIP, he probably has thousands of friends in the social network, but most of them do not are really in contact with him, and probably do not know each other. So, the friendship recommender should take into account this information.

These algorithms are not only intuitive, but they have also a strong theoretical justification (Sarkar, Chakrabarti, and Moore 2010).

Experimental results

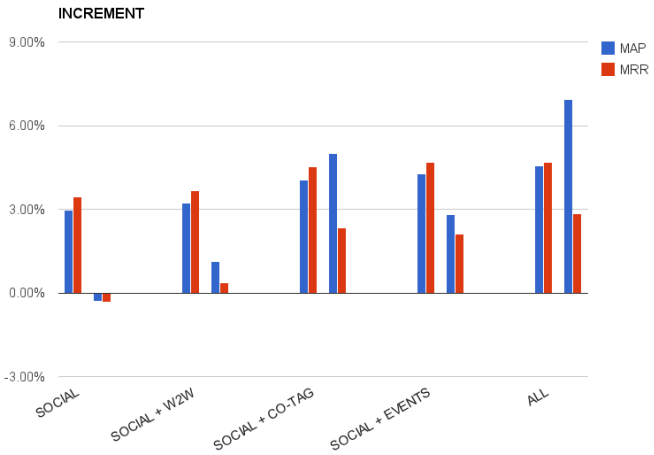

Figure 1: In this chart is reported the increment of performance of $GTG - Rank$ over its best competitor for each combination of graphs. $GTG - Rank$ performs almost always better that its competitor and it clearly exploits additional information much better that AA and CN.

Our experiments are not conclusive, but we can find some interesting results:

- $GTG-Rank$ almost always performs better than its competitors
- Using extra information is useful for predictions and $GTG - Rank$ can exploit it
- Invitations graph and Co-tag graph seem to be the most useful, this makes perfect sense since most of the people those are "close" in the real world usually are also friends in OSN
- We have to further investigate the relevance of the time-component, since a small but recent graph (such the Co-tag one) provides really useful information

Conclusions and ongoing work

In conclusion, we presented a new scalable method for friends recommendations that can keep into account information about the connections strength (weights on the edges). Our method works locally, so it needs information only about a small portion of the graph and it can be easily parallelized.
We compared the proposed algorithm with two well-known competitors on a large-scale commercial social network. The proposed algorithm performs significantly better than its competitors in most of the cases and, in particular, its competitors are not able to exploit information provided by modern social networks.

At the moment we are working in two different directions:

- create new methods to merge graphs in order to get better results
- extend methods created for signed graphs (i.e. (Cesa-Bianchi et al. 2012b; 2012a)) in order to predict new friendships

References

Adamic, L. A., and Adar, E. 2003. Friends and neighbors on the web. *Social Networks* 25(3):211 – 230.

Backstrom, L., and Leskovec, J. 2011. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '11, 635–644. New York, NY, USA: ACM.

Boldi, P., and Vigna, S. 2004. The webgraph framework i: Compression techniques. In *International World Wide Web Conference*, WWW '04.

Cesa-Bianchi, N.; Gentile, C.; Vitale, F.; and Zappella, G. 2012a. A correlation clustering approach to link classification in signed networks. In *Proceedings of Conference on Learning Theory*, COLT '12.

Cesa-Bianchi, N.; Gentile, C.; Vitale, F.; and Zappella, G. 2012b. A linear time active learning algorithm for link classification. In *Proceedings of Advances in Neural Information Processing Systems*, NIPS '12.

Cremonesi, P.; Koren, Y.; and Turrin, R. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proc. of RecSys '10*, 39–46.

Daskalakis, C.; Goldberg, P. W.; and Papadimitriou, C. H. 2006. The complexity of computing a nash equilibrium. In *Proceedings of the 38th annual ACM symposium on Theory of computing*, STOC '06.

Erdem, A., and Pelillo, M. 2011. Graph transduction as a non-cooperative game. In *Graph-base Representation in Pattern Recognition*. Lecture Notes in Computer Science.

Howson, J. T. 1972. Equilibria of polymatrix games. In *Management Science, 18*.

Kang, U.; Tsourakakis, C. E.; and Faloutsos, C. 2009. Pegasus: A peta-scale graph mining system - implementation and observations. In *IEEE International Conference On Data Mining*, ICDM '09.

Lü, L., and Zhou, T. 2011. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications* 390(6):1150 – 1170.

Menon, A. K., and Elkan, C. 2011. Link prediction via matrix factorization. In *Proceedings of the 2011 European conference on Machine learning and knowledge discovery in databases - Volume Part II*, ECML PKDD'11, 437–452. Berlin, Heidelberg: Springer-Verlag.

Miller, K.; Griffiths, T.; and Jordan, M. I. 2010. Nonparametric latent feature models for link prediction. In *Advances in Neural Information Processing Systems (NIPS) 22*.

Newman, M. 2001. Clustering and preferential attachment in growing networks. *Physical Review E* 64(2).

Salton, G., and McGill, M. J. 1986. *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, Inc.

Sarkar, P.; Chakrabarti, D.; and Jordan, M. 2012. Nonparametric link prediction in dynamic networks. In *Proc. of the International Confernece Machine Learning*.

Sarkar, P.; Chakrabarti, D.; and Moore, A. 2010. Theoretical justification of popular link prediction heuristics. In *Proceedings of the 23rd Annual Conference on Learning Theory*, COLT '10.

Tong, H.; Faloutsos, C.; and Pan, J.-Y. 2006. Fast random walk with restart and its applications. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06, 613–622. Washington, DC, USA: IEEE Computer Society.

Zhu, J. 2012. Max-margin nonparametric latent feature models for link prediction. In *Proc. of the International Confernece Machine Learning*.