Structural Link Analysis from User Profiles and Friends Networks: A Feature Construction Approach

William H. Hsu Joseph Lancaster Martin S.R. Paradesi Tim Weninger
Department of Computing and Information Sciences, Kansas State University
234 Nichols Hall
Manhattan, KS 66506-2302
+1 785 5326350

{bhsu | joseph | pmsr | weninger}@ksu.edu

Abstract

We consider the problems of predicting, classifying, and annotating friends relations in friends networks, based upon network structure and user profile data. First, we document a data model for the blog service *LiveJournal*, and define a set of machine learning problems such as predicting existing links and estimating inter-pair distance. Next, we explain how the problem of classifying a user pair in a social network, as directly connected or not, poses the problem of selecting and constructing relevant features. We document feature analyzers for attributes that depend only on graph attributes, those that depend on individual user demographics and set-valued attributes (e.g., interests, communities, and educational institutions), and those that depend on candidate user pairs. We then extend our data model using whole-network attributes and report machine learning experiments on learning the concept of a connected pair of friends from *LiveJournal* data. Finally, we develop a theory of dependent types for deriving causal explanations and discuss how this can be used to scale statistical relational learning up to our full corpus, a recent crawl of over a million records from *LiveJournal*.

General Terms

Algorithms, Experimentation

Keywords

data mining, link analysis, machine learning, social network analysis, user profiling.

1. Introduction

Analysis of friends networks provides a basis for understanding the web of influence [Ko01] in social media. In particular, the problems of determining the existence of links and of classifying and annotating known links are first steps toward identifying potential relationships. This inferred information can in turn be used to introduce new potential friends to one another, make basic recommendations such as community recruits or moderator candidates, or identify whole cliques and communities.

In this paper, we consider the problem of discovering links in an incomplete graph. We present an approach to link prediction that is based on graph feature analysis and intrinsic attributes of entities (users and communities). We report some promising preliminary results on radius-limited neighborhoods of the blogging service *LiveJournal* and discuss the results of exploratory experiments that point toward a need to differentiate the types of features in a friends network, namely:

1. those that depend on the demographics of the entire network
2. those that are computable for each user or each pair of user
3. those that depend on the existence of a reported, inferred, or suspected link

We derive some such features and discuss the costs of computing, selecting, and recombining them. Of particular interest in the domain of commercial weblogs and social media are demographic features relevant to collaborative recommendation of goods and formation of branding communities. The structural dependence and context-specific dependence of features determines what new features are feasible to construct, both in terms of statistical sufficiency and computational complexity.

In conclusion, we examine some new features that were derived by hand, discuss the algorithms used to compute them, and relate these specific algorithms to a broader class of relational database queries that form the basis of a more powerful feature construction system.

2. Background

2.1 Friends Networks from User Profiles

Social network services such as *MySpace* and *Facebook* allow users to list interests and link to friends, sometimes annotating these links by designating trust levels or qualitative ratings for selected friends. Some such services, such as Google's *Orkut*, are community-centric; others, such as the video blogging service *YouTube* and the photo service *Flickr*, emphasize social media; while some, such as Six Apart's *LiveJournal* and *Vox*, are organized around text-and-image weblogs. *LiveJournal* and its derivative services, such as *GreatestJournal*, *DeadJournal*, and *JournalFen*, are based on the same open-source server code. At the time of this writing, there are over 11.7 million *LiveJournal* accounts, 1.8 million of them active.

The friends network of *LiveJournal*, our topic of study, has two varieties of accounts: users and communities (we omit RSS feeds). One advantageous property of its data model, stemming from a common schema for the two account types (which could originally be converted from user to community), is that it provides a simple, flexible representation for entities and relations.

Start	End	Link Denotes
User	User	Trust or friendship
User	Community	Readership or subscribership
Community	User	Membership, posting access, maintainer
Community	Community	Obsolete

Table 1. *Types of links in the blog service LiveJournal.*

Table 1 shows the types of links in *LiveJournal* and their constituent attributes. Friendship is an asymmetric relation between two accounts, each represented by a vertex in a directed graph. The type of the start and endpoint defines the relationship set attributes of the link. For example, a user u who adds another user v to his or her **friends list** can specify the membership in any of up to 30 groups. These serve the dual purpose of blog aggregation (posts from each group's members are filtered into its aggregator page, which u can read or make public) and groups-based security (each group denotes a read/comment access control list). Access control lists for communities are associated with memberships (community-to-user links), while content is controlled by posters or subscribers. A user can "**watch**" a community in order to add all accessible posts to a main aggregator page or to custom groups. The set of accessible posts consists of either public posts only, or public and restricted (members-only) posts. The access control list is defined by the membership relation and individual posters' selections (whether to allow comments and whether to display them by default from no readers, all readers, non-anonymous readers, or community members). Acquisition of privileges is a community property, of which only membership may be acquired solely by user action ("**joining**" a community), if the moderator has specified open membership.

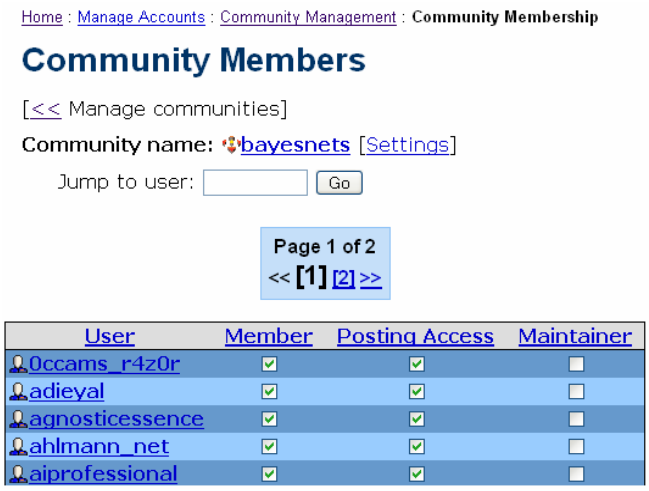

Figure 1. *LiveJournal access control list maintenance (community moderator interface).*

Thus, a reciprocal link between a user and a community means that the user both subscribes to the community and is an approved member. Links from user u to v are listed in the "Friends" list of u and in an optionally displayed "Friends Of" list of v. This list can be partitioned into reciprocal and non-reciprocal sublists for a user u:

Mutual Friends: $\{ v \mid (v, u) \in E \wedge (u, v) \in E \}$

Also Friend Of: $\{ v \mid (v, u) \in E \wedge (u, v) \notin E \}$

The community analogue of the "Friends Of" list is the "Watched By" (subscriber) list, whose members have the community name listed in the "Friends: Communities" sections of their individual user profile pages. The community analogue of the "Friends" list is the "Members" list.

The friends network for *LiveJournal* consists of a very large central connected component and many small islands, most of which are singleton users. There are a few source vertices, corresponding to accounts that link to others but have no reciprocated friendships; these are usually RSS or blog aggregator accounts owned by individuals. Additionally, there are sink vertices corresponding to accounts watched by others, but which have named no friends. Some of these are channels for announcement or dissemination of creative work.

2.2 Link Identification

In previous work [HKP+06], we introduced a link prediction problem for *LiveJournal*: given a graph in which the existence of a candidate link is hidden (elided if it exists), classify it as present or absent given all other attributes of the graph and of the endpoints. Our initial approach to link identification consisted of dividing friends network features into graph features and interest-based features.

Graph features could be computed simply by scanning the graph, in the case of pair-distance metrics, performing all-pairs shortest path (APSP) search:

1. Indegree of *u*: popularity of the user
2. Indegree of *v*: popularity of the candidate
3. Outdegree of *u*: number of other friends besides the candidate; saturation of friends list
4. Outdegree of *v*: number of existing friends of the candidate besides the user; correlates loosely with likelihood of a reciprocal link
5. Number of mutual friends w such that $u \rightarrow w \wedge w \rightarrow v$
6. "Forward deleted distance": minimum alternative distance from *u* to *v* in the graph without the edge (u, v)
7. Backward distance from *v* to *u* in the graph

These were supplemented by interest-based features:

8. Number of mutual interests between *u* and *v*
9. Number of interests listed by *u*
10. Number of interests listed by *v*
11. Ratio of the number of mutual interests to the number listed by *u*
12. Ratio of the number of mutual interests to the number listed by *v*

2.3 Efficient feature analysis

The degree attributes can be enumerated in time linear in the number of users, as can the mutual friends count for each pair of users.

Forward deleted distance measures the distance from u to v by alternate routes, after the edge (u, v) is elided. The prediction task is thus to reconstruct the incomplete graph resulting from this erasure, to determine whether a particular link (u, v) existed. Forward deleted distance can be precomputed exhaustively for the entire graph in $\Theta(|E| (|V| + |E|)) = \Theta(|E|^2)$ time by erasing each edge in E and re-running a breadth-first search from the start vertex. If a candidate edge is not stored in the resulting cache, its deleted distance is that found by BFS on the original graph, in $\Theta(|V| + |E|)$ time. In a graph (V, E), backward distance requires $\Theta(|V| + |E|)$ using BFS for a particular candidate edge. Since the expected size of the edge set is $E[|E|] = k|V|$, about $k = 20$ on average across *LiveJournal*, the bottleneck computation is that of forward deleted distance: $\Theta(|E|2) = \Theta(k^2|V|^2)$, or $\Theta(|V|^2)$ with a large constant.

Using a straightforward string pair enumeration and comparison algorithm, the mutual interest counts are stored in matrix of |V|2 elements, each requiring constant time to check (given a maximum of 150 interests).previous work [HKP+06], we introduced a link prediction problem for LiveJournal: given a graph in which the existence of a candidate link is hidden (elided if it exists), classify it as present or absent given all other attributes of the graph and of the endpoints. Our initial approach to link identification consisted of dividing friends network features into graph features and interest-based features.

2.4 Methodologies for link mining

Getoor and Diehl [GD05] recently surveyed techniques for link mining, focusing on statistical relational learning approaches and emphasizing graphical models representations of link structure. Ketkar *et al.* [KHC05] compare data mining techniques over graph-based representations of links to first-order and relational representations and learning techniques that are based upon inductive logic programming (ILP).

Sarkar and Moore [SM05] extend the analysis of social networks into the temporal dimension by modeling change in link structure across discrete time steps, using latent space models and multidimensional scaling. One of the challenges in collecting time series data from *LiveJournal* is the slow rate of data acquisition, just as spatial annotation data (such as that found in LJ maps and the "plot your friends on a map meme) is relatively incomplete.

2.5 Other applications using graph mining

Popescul and Ungar [PU03] learn a kind of entity-relational model from data in order to predict links. Hill [Hi03] and Bhattacharya and Getoor [BG04] similarly use statistical relational learning from data in order to resolve identity uncertainty, particularly coreferences and other redundancies (also called deduplication). Resig *et al.* [RDHT04] use a large (200000-user) crawl of *LiveJournal* to annotate a social network of instant messaging users, and explore the approach of predicting online times as a function of friends graph degree.

There have been numerous recent applications of social network mining based on the text and headers of e-mail. One notable research project by McCallum *et al.* [MCW05] uses the Enron e-mail corpus and infers roles and topic categories based on link analysis A primary goal of this work is to extend the graph mining approach beyond link **prediction** and **recommendation** towards link **explanation** and **annotation**.

It may be much more useful to explain why a group of friends in a blog service created accounts *en masse* or added one another as friends than to recommend relationship sets that are already extant or structured according to a preexistent social group. For example, high school classmates often create accounts and encourage their peers to join the same service. In a few cases, this is encouraged or facilitated by a teacher, for a class project. Solving the problem of link prediction is not particularly useful in this case, because the user decisions have already been made or strongly constrained; however, it may be very useful to link other classmates **not** working on the same project to the same relationship set (perhaps they were encouraged to join the blog service by students who continued to use it after the class project).

Large groups such as web comic subscriberships, community co-members, etc. are also somewhat identifiable, and relating members of a blog service to one another through relationship sets is a typical entity-relational data modeling operation that can be made more robust and efficient through graph feature extraction.

3. Experiment Design

3.1 *LJCrawler v2*

To acquire the graph structure and attributes describe in the previous section, we developed an HTTP-based spider called *LJCrawler* to harvest user information from *LiveJournal* A multithreaded version of this program, which retrieves BML data published by Denga (the owners of *LiveJournal*), collects an average of up to 15 records per second, traversing the social network depth-first and archiving the results in a master index file. Because *LiveJournal*'s functionality for looking up users by user number is only available to administrators, we decided to compile a list of seeds for a disjoint-set representation of the disconnected social network. For purposes of this experiment, however, starting from just one seed (the first author's *LiveJournal* ID) and restricting the crawl to one connected component was sufficient.

Using *LJCrawler*, we compiled an adjacency list and the following ground features for each user:

- Account type (user, community)
- Interest list
- School list
- Communities watched list
- Community membership list
- Friends of list
- Friends list

3.2 Feature Analyzers

We define a single example to be a candidate edge (u, v) in the underlying directed graph of the social network, along with a set of descriptive features calculated from the annotated graph recorded by *LJCrawler*:

Other features: Additional planned features for continuing experiments include dates (update frequencies when taken differentially), user options such as maximum friends count, and content descriptors of *LiveJournal* entries and comments (average post length, word frequency, etc.).

3.3 Graph Search Algorithms for Computing Features

Computing the minimum forward and backward distances can be done more efficiently by using breadth-first search. Currently, a Java implementation of this algorithm requires under one minute on a 2GHz AMD Opteron system to process a 2000-node graph. However, enumerating all possible candidate pairs within a neighborhood of 2 nodes (1.6 million pairs for 4000 nodes) requires several hours on the same system.

We note that the amortized cost of running BFS to precompute all-pairs shortest paths (APSP) with the actual edge deleted (which is necessary to avoid knowing the prediction target in link predicton) is $\Theta(|E|\,(|V| + |E|))$. This is prohibitively large even for our "mid-sized" subgraphs of 10-50K nodes; when $|V|$ is about 11 million, $|E|$ is a little over 200 million, enumerating APSP is completely infeasible. However, we do not typically consider all of E, so the bottleneck is typically the first step plus a constant number of calls to BFS, requiring running time in $\Theta(k\,(|V| + |E|))$.

3.4 Generating Candidates

We considered several alternative ways to generate candidate edges (u, v):

The first technique is likely to be unscalable, as the number of candidates is $|V|^2$. The second requires having a representatively large sample of the full *LiveJournal* social network, in order to fit the distribution parameters accurately. The third was the most straightforward to implement. Two calls to the all pairs shortest path algorithm provided cost matrix, and one pass at each radius up to a maximum of 10 yielded the data shown in Table 2. To simplify the initial experiments, we defined the classification problem to be classification of $d(u, v)$ as 1 or 2.

This task is actually useful for social network recommender systems because discrimination of a direct friend from a "friend of a friend" (FOAF) is functionally similar to recommending FOAFs to link to directly. There are more detailed classification targets, such as placement, promotion, and demotion of linked friends within strata of trust (setting, increasing, and decreasing the security level), but choosing a user's friends to begin with is the more fundamental decision.

Table 2 and Table 3 report the distribution of inter-vertex distances in the friends network for two subnetworks induced by limiting the maximum number of nodes.

Distance d	Frequency (= d)	Cumulative (≤ d)
1	**6204**	**6204**
2	**107307**	**113511**
3	69896	183407
4	59926	243333
5	3400	246733
6	255	246988
7	16	247004
8	1	247005
9	0	0
10	0	0
∞	9731	256735

Table 2. *Number of candidate edges for the 1000-node LiveJournal graph.*

Distance d	Frequency (= d)	Cumulative (≤ d)
1	**19410**	**19410**
2	**370568**	**389978**
3	403075	793053
4	520373	1313426
5	123747	1437173
6	18453	1455626
7	2657	1458283
8	339	1458622
9	29	1458651
10	0	1458651
∞	174534	1633185

Table 3. *Number of candidate edges for the 4000-node LiveJournal graph.*

4. Results

4.1 Preliminary experiment: 941-node version

In a preliminary experiment, we constructed a 941-node subgraph, defining the concept *IsFriendOf* and trained three types of inducers with:

1. all attributes
2. all graph attributes excluding the forward and backward distances
3. the backward distances alone
4. the backward and forward distances alone
5. interest-related attributes alone.

Table 4 and Table 5 show the results for three inducers: the *J48* decision tree inducer, Holte's *1R* inducer (a single-rule classifier based on a single attribute) [Ho93], and the Logistic regression inducer. All accuracy measures were collected over 10-fold cross-validated runs. The J48 output wth all features achieves a significant boost over the next highest (distance only).

Inducer	All	NoDist	BkDist	Dist	Interest
J48	*98.2*	94.8	95.8	97.6	88.5
OneR	95.8	92.0	95.8	95.8	88.5
Logistic	91.6	90.9	88.3	88.9	88.4

Table 4. *Percent accuracy for predicting all classes using the 941-node graph.*

Inducer	All	NoDist	BkDist	Dist	Interest
J48	*89.5*	65.7	67.7	83.0	5.4
OneR	67.7	41.1	67.7	67.7	4.5
Logistic	38.3	33.3	0	4.5	4.5

Table 5. *Precision (true positives to all positives) using the 941-node graph.*

4.2 Experiments on restricted graphs

We developed an application, **`ljclipper`**, to restrict the overall friends graph to that induced by a subset of nodes of fixed number, found using breadth-first search starting from a given seed. Using a 4000-node subgraph summarized in Table 3, we generated 1633185 candidate edges. Note that all forward distances are greater than 1: when u and v are actually connected, we erase (u, v). In preliminary experiments, we then computed the length of the shortest alternative path. This is, however, a less scalable approach, because the asymptotic running time is dominated by the superlinear time required to compute

The complete listing of all twelve features is given in Section 2.2. The numerical types of all of the network features – both the ones describing the graph and those measuring and interests and ratios – makes data set amenable to logistic regression.

Inducer	Accuracy	Precision	Recall
J48	99.9	97.5	96.1
OneR	99.6	91.7	91.8

Table 6. *Percent accuracy, precision and recall using a 1000-node graph (10-fold CV).*

Inducer	Accuracy	Precision	Recall
J48	99.8	95.8	92.0
OneR	99.7	91.1	89.9

Table 7. *Percent accuracy, precision and recall using a 2000-node graph (10-fold CV).*

Inducer	Accuracy	Precision	Recall
J48	99.8	94.5	88.3
OneR	99.7	88.2	84.3

Table 8. *Percent accuracy, precision and recall using a 4000-node graph (10-fold CV).*

Table 6 through Table 8 show the accuracy, precision, and recall for the 1000, 2000, and 4000-node friends graphs. Trends of higher precision than recall, and diminishing precision and recall as the network grows larger, are observed. These trends are sustained for subsamples of size 10000 and size 100000, though precision and recall also diminish slightly with sampling.

4.3 Data acquisition and larger experiments

The crawler has been improved with several service-specific optimizations for fetching user info pages. Presently these do not use *LiveJournal*'s BML feed of user data, which is incomplete for our purposes (that is, not all ground attributes in our initial relations are provided). At press time, this crawler processes about 20000 user records per hour and thus would require over a week to crawl *LiveJournal.*

The current bottleneck is the $\Theta(|V|\ (|V| + |E|))$ step described in Section 3.3. This is the dominant term, because the constant k denoting the number of candidate edges is usually much smaller than n, e.g., 100-1000, so that $\Theta(k\ (|V| + |E|))$ is not only in $\Theta\ (|V| + |E|)$, but actually just a few hundred times the cost of a single BFS.

4.4 Interpretation

Using mutual interests alone, even with normalization based on the number of interests in u and v, results in very poor prediction accuracy using all inducers with which we experimented. Intermediate results are achieved using mutual friends count and degree (NoDist: 65.7% on predicting edges) and using forward deleted distance and backward distance (Dist: 67.7%). Using all 12 computed graph and annotation features resulted in the highest precision (All: 89.5%) and accuracy (All: 98.2%).

We note that *LiveJournal* once used a variant of normalized mutual interests to produce a list of potential friends, arranged in decreasing order of match quality. Although this was not the same type of recommender system as *LJMiner* supports, it shows that the state of the art user matching systems have a lot of room for improvement. The results indicate that features produced by *LJMiner*, used with a good inducer, can generate collaborative and structural recommendations.

5. Continuing Work

Scaling up: Our current research focuses on scaling up to tens of thousands and eventually millions of users. Crawling over 11-12 million records is at least technically feasible, but scaling up the graph analyzers is a challenge that may best be met with heuristic search.

Learning relational models: A promising area of research is the recovery of relational graphical models, including class-level (membership and reference slot) uncertainty. [GFKT02] *LJMiner* has yielded a ready source of semistructured data for both structure learning and distribution learning. Another potentially useful approach is to organize users and communities into clusters using this relational model. We have developed schemas for blog posts (entries, threads, comments) and for users and dynamic groups of users. This is related to previous preliminary work on relational data mining for personalization of web portals, especially computational grid portals. [HBJ03]. Much of the relational metadata in the bioinformatics domain comes from description languages for workflows and workflow components [Hs04]. The next step in our experimental plan is to use schemas such as our detailed ones for blog sevice users and bioinformatics information and computational grid users [Hs05] to learn a richer predictive model. Finally, modeling relational data as it persists or changes across time is an important challenge.

Acknowledgements

We thank Todd Easton and Kirsten Hildrum for helpful discussions concerning algorithms and the *LiveJournal* data model. We also thank Andrew King and Tejaswi Pydimarri for contributions to the original *LJMiner* system and Vikas Bahirwani for contributions to the second version.

References

[BG04] I. Bhattacharya & L. Getoor. Deduplication and group detection using links. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) Workshop on Link Analysis and Group Detection (LinkKDD2004)*, Seattle, WA, USA, August 22-25, 2004.

[CLRS02] T. H. Cormen, C. E. Leiserson, R. L. Rivest, & C. Stein. *Introduction to Algorithms, Second Edition*. Cambridge, MA: MIT Press, 2002.

[GD05] L. Getoor & C. P. Diehl. Link mining: a survey. *SIGKDD Explorations, Special Issue on Link Mining*, **7**(2):3-12.

[GFKT02] L. Getoor, N. Friedman, D. Koller, & B. Taskar. Learning Probabilistic Models of Link Structure. *Journal of Machine Learning Research*, 2002.

[HBJ03] W. H. Hsu, P. Boddhireddy, & R. Joehanes. Using probabilistic relational models for collaborative filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) Workshop on Statistical Learning of Relational Models (SRL)*, Acapulco, MEXICO, August, 2003.

[Hi03] S. Hill. Social network relational vectors for anonymous identity matching In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) Workshop on Statistical Learning of Relational Models (SRL)*, Acapulco, MEXICO, August, 2003.

[Ho93] R. C. Holte. Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. *Machine Learning*, 11(1):63-90.

[Hs04] W. H. Hsu. Relational graphical models of computational workflows for data mining. In *Proceedings of the International Conference on Semantics of a Networked World: Semantics for Grid Databases (ICSNW-2004)*, p. 309-310, Paris, FRANCE, June, 2004.

[Hs05] W. H. Hsu. Relational graphical models for collaborative filtering and recommendation of computational workflow components. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) Workshop on Multi-Agent Information Retrieval and Recommender Systems*, Edinburgh, UK, July 31, 2005.

[HKP+06] W. H. Hsu, A. King, M. S. R. Paradesi, T. Pydimarri, & T. Weninger. Collaborative and Structural Recommendation of Friends using Weblog-based Social Network Analysis. In *Proceedings of the 2006 AAAI Spring Symposium on Computatational Approaches to Analyzing Weblogs (CAAW 2006)*.

[KHC05] N. S. Ketkar, L. B. Holder, & D. J. Cook. Comparison of graph-based and logic-based multi-relational data mining. *SIGKDD Explorations, Special Issue on Link Mining*, **7**(2):64-71.

[Ko01] D. Koller. Representation, Reasoning and Learning. *IJCAI Computers and Thought Award Lecture,* 2001.

[MCW05] A. McCallum, A. Corrada-Emmanuel, & X. Wang. Topic and role discovery in social networks. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Edinburgh, UK, August, 2005.

[MH04] M. Mukherjee & L. B. Holder. Graph-based data mining on social networks. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) Workshop on Link Analysis and Group Detection (LinkKDD2004)*, Seattle, WA, USA, August 22-25, 2004.

[PU03] A. Popescul & L. H. Ungar. Statistical relational learning for link prediction. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) Workshop on Statistical Learning of Relational Models (SRL)*, Acapulco, MEXICO, August, 2003.

[RDHT04] J. Resig, S. Dawara, C. M. Homan, & A. Teredesai. Extracting social networks from instant messaging populations. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) Workshop on Link Analysis and Group Detection (LinkKDD2004)*, Seattle, WA, USA, August 22-25, 2004.

[SM05] P. Sarkar & A. Moore. Dynamic social network analysis using latent space models. *SIGKDD Explorations, Special Issue on Link Mining*, **7**(2):31-40.