PREDICTING LINKS AND LINK CHANGE IN FRIENDS NETWORKS: SUPERVISED TIME SERIES LEARNING WITH IMBALANCED DATA

WILLIAM H. HSU
234 Nichols Hall
Kansas State University
Manhattan, KS 66506

TIM WENINGER
234 Nichols Hall
Kansas State University
Manhattan, KS 66506

MARTIN S.R. PARADESI
234 Nichols Hall
Kansas State University
Manhattan, KS, 66506

Abstract

We address the problem of predicting links and link change in friends networks and introduce a new supervised learning method for both types of prediction. This extends previous based on directed graph features such as the indegree of candidate friends and pair dependent relational features such as common interests. In this new work, we consider how differential user data, such as that produced using regular crawls from a social network site, can be used to produce a time series with which we can identify prediction problems over both links and link change. A key issue we address is the rarity of change between two successive versions of a social network, resulting in severe imbalance between positive and negative examples of change. We compare existing approaches towards coping with this problem, present positive results on new crawls of *LiveJournal*, and consider how temporal data can enhance the relational link mining process.

INTRODUCTION

The problem of predicting links between entities such as users and communities in a friends network can be treated as one of supervised inductive learning for classification. In previous work (Hsu et al. 2007), we introduced a system for classifying pairs of users who were known to lie within a radius of 2 of one another as *friends* or *friends of friends*. This classification task was defined on a data sets consisting of 1000 and 4000 users from the blog service *LiveJournal.* Analysis of the graph structure and pair-dependent sets (e.g., mutual friends and common interests) produced a set of 12 features for each candidate pair. From this set of features, an effective predictor for link existence could be learned. However, there were two key limitations to this approach. First, the features made available to machine learning algorithms included certain information that is not always available for prediction tasks. For example, in many social networks such as *Facebook* and *LinkedIn*, a user who has been added to the friend set of another is prompted for whether to issue a reciprocal link, whereas realistic prediction may require link existence to be identified before such information is known. Thus, some latency is inherent in realistic prediction task specifications. Second, a key unaddressed problem in this and other related work is that treating link existence as a function of a single snapshot of the friends network fails to take into account the full history of the graph. We show in this paper that data about changes to the link structure over time can provide an effective set of indicators for future change. Therefore, we formulate the problem of link existence prediction in terms of predicting change, given the recent history of: graph topology, user features, and features of candidate friends. This approach provides the features required for machine learning algorithms to be able to learn predictive cues about imminent change in link existence – i.e., addition and deletion of friendships. Our experiments demonstrate that predictors learned from atemporal data tend to grossly oversimplify the conditions of link change, resulting in poor generalization quality. By using temporal data, we capture both graph-based and user-based indicators of imminent link change, resulting in much higher prediction accuracy, precision, and recall.

The novel contribution of this time series formulation is that it makes available the full information contained in regularly collected snapshots of a social network site. This information, in turn, supports more robust learning to predict link change than for a single crawl.

BACKGROUND

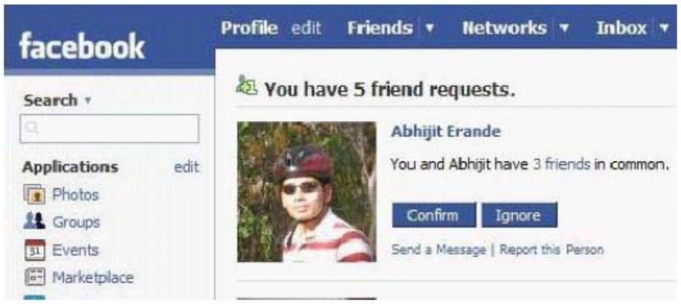
Friends Networks from User Profiles

In our original study of link prediction by classification (Hsu et al. 2006), we defined a simple data model for a feature set (i.e., an attribute vector or tuple). Each tuple corresponds to a candidate pair. We first enumerated tuples of seven graph features such as *indegree of candidate friend* from *LiveJournal*'s friends network, and augmented this schema with five relational attributes such as *number of mutual interests*. This system precomputes data for supervised inductive learning whose objective is to classify candidate friendships based on observable features of the source entity u and a target entity v. Some features, such as degree, were dependent on only u or only v; others, such as *backward distance*, were measured between v and u. The most basic prediction task is to take as input an unlabeled instance (tuple of all 12 node-dependent and pair-dependent attributes) and label (u, v) as existing or not. Ground truth, i.e., whether $(u, v) \in E$, is known. We refer the interested reader to (Hsu et al. 2007) for full documentation of these attributes and the inductive learning experiments conducted.

Link prediction versus change prediction

The link prediction problem has been studied as a classification problem (Popescul and Ungar 2003), with applications of relational link extraction by clustering becoming prevalent in recent research (Getoor and Diehl 2005). Our technical objective in this new work was to extend the problem definition, from classification on static instances to classification of instances in a sequence as being examples of *change* or *no change* in link structure. We henceforth refer to this problem as *link change prediction*.

The key contribution of our original work (Hsu et al. 2006) was that it identified a set of graph features that could be used to learn to classify two users *known* to lie within a radius of 2 of each other as being distance 1 apart (friends) or distance 2 apart (friends of friends). The test set accuracy of decision surfaces learned using this data was in excess of 97% on 1000-node data sets, while the precision and recall were in excess of 80% each. Limitations of this work were that it provided only a method for predicting whether two users were friends given that their *entire* set of mutual friends, the overall degree of both the initiating friend and receiving friend, and the reciprocity of the friendship. This does not provide as fully realistic a training scenario as a real incremental crawl of a social networking site would, because for a typical u and v where the existence of (u, v) in the edge set E is being predicted, a) the candidates are not necessarily known to lie less than or equal to two nodes apart and b) it is not *necessarily* known whether (v, u) is already in E. $(v, u) \in E$ is a triggering event for adding one friendship (or breaking others), and is employed in social networks that send notifications and prompt the user to choose whether to reciprocate, as is the case with *Facebook* (shown in Fig. 1), *LinkedIn*, and *Flickr.*



You have 5 friend requests.

Abhijit Erande

You and Abhijit have 3 friends in common.

Figure 1. Facebook reciprocation dialogue

The key novel advance of the second work (Hsu et al. 2007) was that it derived algorithms for computing graph features efficiently, in time quadratic in the number of nodes, i.e., O(kn) = O(k |V|), where k is the square of the average degree of a node in the friends network. We bounded this number empirically to be between 20 and 30 on average for *LiveJournal*, although there are some "islands", i.e., users with indegree and outdegree 0. Such users' profiles cannot be crawled due to their being in separate strongly connected components and due to *LiveJournal's* lack of random access to users' records by number. The record for indegree was over 5000, held by **`iharthdarth`**, the author of a popular web comic.

In the second paper, we still used the "distance 1 vs. 2" problem as a benchmark, but recognized the need for better tests. In exploratory experiments at this time, we found that *ab initio* classification of user pairs by expected distance yielded poor results for the exact case, and mediocre results for learning an upper bound.

Methodologies for link mining

In addition to the classification approach of Popescul and Ungar (2003), we studied the SUBDUE system of Mukherjee and Holder (2004), which uses graph algorithms to find frequently occurring subgraphs as a preprocessing step for supervised inductive learning. Bhattacharya and Getoor (2004) similarly use statistical relational learning from data to address the deduplication problem.

Our approach to link mining is based on classical inducers for supervised learning, such as J48, Logistic Regression, and OneR (Holte 1993). However, the learning framework introduced in (Hsu, et al. 2006), and extended in (Hsu et al. 2007) and this work, is extensible to other approaches, such as the inductive logic programming (ILP) system compared against SUBDUE by Ketkar, Holder, and Cook (2005).

EXPERIMENT DESIGN

Data Acquisition: LJCrawler v3

LJCrawler v3 is a multi-threaded, parallel HTTP crawler that effectively gathered user information in a breadth-first manner beginning with *LiveJournal* user **`darthvader`**, which was randomly selected. Specifically, the crawler queried user data including, but not limited to, user age, interests, friends, schools, communities, etc. Running on a single computer, *LJCrawlerv3* is bounded by only by network capacity, and was shown to crawl at most 200 users per second. Keeping within the *LiveJournal* terms of service, we limited the crawling capacity to generally five users per second.

The crawler was run for a total of three hours every six hours for seven days from 00:00 CST on September 27, 2007, to 18:00 CST on October 03, 2007, for a total of 28 crawls.

Prediction tasks

We identified two prediction tasks – one based on a single feature set from a single run of *LJCrawlerv3* and another based on all 28 crawls. Both tasks shared the target concept of a *newly-added friendship*. That is, a candidate pair (u, v) was labeled as positive if and only if (u, v) $\notin E_{t-1}$ and (u, v) $\in E_t$. Specifically, the first task (atemporal) was based on the friendships in the initial crawl from 00:00 CST on September 27, 2007, as compared to the friendships that existed in the final crawl at 18:00 CST on October 03, 2007.

Alternatively, the second task (temporal) was based on learning from the incremental differences between each successive pair out of the 28 crawls. In order to learn from successive pairs, we structured the training features in the following manner:

$$f_{t-4}, f_{t-3}, f_{t-2}$$
$$f_{t-3}, f_{t-2}, f_{t-1}$$
$$f_{t-2}, f_{t-1}, f_t$$

where f_{t-k} is the feature tuple from the crawl at time $t – k$ (k crawls ago).

Handling imbalanced data

The problem of imbalance between the frequency of positive and negative examples is endemic to change prediction in many domains. For example, Kubat, Holte, and Matwin observed that certain anomaly detection problems such as detecting oil spills in satellite images involved a vastly greater number of negative examples than positive (1998). Similarly, we encounter many more cases where a user neither adds nor deletes friends during a short period of a few days or weeks. Anecdotally, users often delete friends as a regular mass action (such as a "friends cut") or add entire social groups or cliques upon their own arrival at a blog or social network service, or when a friend joins who is known offline to the user.

The approach commonly taken to cope with imbalance is to *downsample* negative examples or *upsample* positive examples. (Kubat and Matwin 1997). Our approach effectively upsamples positive examples by taking all available instances of change: known additions of friends-links for users who were not friends previously, or retractions of links for users who were. We then generate random cases (predominantly negative for "change") either according to a:

1. **Fixed Ratio (FR)**, where the number of positive and negative cases is deliberately equalized, i.e., kept at a 1-to-1 ratio
2. **Fixed Count (FC)**, where the number of positive and negative cases are sampled randomly from among the population.

For example, given a hypothetical population size of 1000 with 900 negative examples and 100 positive examples, if we wished to find a sample of size 100 the FR-sample would contain exactly 50 negative and 50 positive examples, whereas the FC-sample would contain approximately 90 negative examples and only 10 positive examples. Note, the FR-sample does not maintain the original distribution of the population in order to provide more positive training examples, whereas the FC-sample is a simple random sampling of the population. Therefore, we propose that an effective and prudent evaluation of our prediction problem is to use the FR-sample as training data and the FC-sample as test data.

Atemporal data set

We adapt a feature set first derived in (Hsu et al. 2006):

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. "Forward deleted distance": minimum alternative distance from *u* to *v* in the graph without the edge (*u*, *v*)
6. Backward distance from *v* to *u* in the graph
7. Number of mutual interests between *u* and *v*
8. Number of mutual friends w such that $u \rightarrow w \wedge w \rightarrow v$
9. Number of schools that *u* and *v* list in common

The last feature is a new one adopted by *LiveJournal* in recent years.

We refer the reader to (Hsu et al. 2007) for details of efficient feature analysis. The new implementation of bidirectional breadth-first search produces a speedup of several times.

We concentrated in this work on predicting additions only rather than additions and deletions. The target feature is the Boolean variable *BecameFriends*.

Temporal data set

The novel contribution of this research, specifically, is the use of graph and link features, and their changes through time. In order to describe the temporality of the graph data the feature set of each pair is shown in a three time-tick window starting with T_0, T_1, T_2, and the proceeding feature set row is shifted to exclude the first time tick and include the next. This is done until the final time tick is reached, giving 25 feature tuples for each pair *u*, *v*.

Enumerating the differences in 28 individual time ticks was shown to be computationally expensive, effectively magnifying the complexity of the temporal feature extraction by the number of time ticks.

Temporal data was thus gathered for 1000, 2000 and 4000 *u, v* pairs. And four training sets were created: FR and FC with and without link features for each size in the same manner as described in the above section on the atemporal data set.

EXPERIMENTAL RESULTS

Experiment Design

Data was generated for 1000, 2000 and 4000 *u, v* pairs. Furthermore, we modified the feature extraction algorithm to generate *four* total sets of training data for each size:

1. Fixed Ratio (FR) with Graph Features
2. FR without Graph Features
3. Fixed Count (FC) with Graph Features
4. FC without Graph Features

The nominal attribute **becameFriends** is denoted as *yes* if pair *u*, *v* was not friends before T_0 and was friends after T_{28}, and *no* otherwise. As proposed in the previous section, our atemporal experiment was evaluated in two ways: (1) trained with FR-sample data and tested with FC-sample data (FR/FC), (2) 10-folds cross validation on FC-sampled data (FC/FC).

With this atemporal data the WEKA implementation of the J48 algorithm was run. The results are shown in Table 1.

Table 1. J48 results, atemporal data set

	FR/FC						FC/FC					
	Graph Features (%)			All Features (%)			Graph Features (%)			All Features (%)		
m	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.
1000	83.0	2.0	100.0	83.0	2.0	100.0	99.5	0.0	0.0	99.3	0.0	0.0
2000	90.7	4.6	90.0	92.1	5.0	90.0	99.5	0.0	0.0	99.4	33.3	2.0
4000	95.1	2.0	66.7	92.6	2.0	100.0	99.9	0.0	0.0	99.7	11.1	16.7

FC/FC results were expectedly poor because in many cases the training sample had only 4 or 5 positive examples for training. In fact, FC/FC on graph features recorded 0 positive examples correctly through all sample sizes. However, when FR-sample data is used for training (FR/FC) the results are noticeably higher. Furthermore, the differences in results of experiments with graph features only and experiments with all features are minimal; this observation is consistent with earlier work.

New results: time series task

Data was again generated for 1000, 2000 and 4000 u, v pairs similar to the atemporal results except that each feature tuple was replicated as f_{t-k_1}, f_{t-k_2}, etc. as described in the earlier sections. This realignment allows the inducer to learn temporal actions that may lead to changes in the friendship status of a pair. This hypothesis is empirically shown to be true in Tables 2-4.

Table 2. Results for all inducers, FR/FR, time series

		J48			Logistic			OneR		
	m	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.
Graph Feature	1000	99.0	100.0	87.5	100.0	100.0	100.0	100.0	100.0	100.0
	2000	98.9	100.0	90.1	93.2	71.7	60.6	99.4	100.0	94.4
	4000	99.4	99.8	97.8	77.9	58.4	27.3	95.1	88.9	90.9
All Features	1000	99.0	100.0	87.5	100.0	100.0	100.0	100.0	100.0	100.0
	2000	98.9	100.0	90.1	95.2	75.0	84.5	99.4	100.0	94.4
	4000	99.3	99.5	97.5	83.1	75.1	43.7	95.1	88.7	90.9

Specifically, Table 2 shows that when FR-samples are cross validated (FR/FR) very high scores are the result. However, by training and testing on population-inconsistent data inherent in the FR-sample these results are likely skewed by overfitting and the resulting prediction rules may not be suitable for real-world data. Table 3 shows more sound results because the inducer is tested with FC-sample data.

Table 3. Results for all inducers, FR/FC, time series

			J48			Logistic			OneR	
	m	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.
Graph Feature	1000	59.9	1.5	75.0	67.3	1.8	74.0	50.7	1.2	75.0
	2000	85.5	2.9	49.3	80.1	0.6	14.1	80.2	1.7	37.4
	4000	84.3	2.3	41.0	80.6	0.4	8.7	77.5	2.1	54.3
All Features	1000	59.9	1.5	75.0	59.9	1.5	75.0	50.7	1.2	75.0
	2000	84.9	2.0	90.1	89.1	0.9	84.5	79.6	1.0	94.4
	4000	84.0	1.0	20.0	79.3	0.5	13.9	76.9	1.5	46.7

Table 3 shows promising results, in that, even though very few positive examples were used in validation because the FC-sample data contains only ≈5% positive examples (which correspond to the population's original distribution).

In our final experiment, we train and cross validate inducers with FC-sample data. The results are shown in Table 4.

Table 4. Results for all inducers, FC/FC, time series

			J48			Logistic			OneR	
	m	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.	Acc.	Prec.	Rec.
Gra. Feat.	1000	100.0	100.0	100.0	99.2	100.0	3.5	99.4	83.7	36.0
	2000	99.9	100.0	99.1	99.1	65.5	4.3	99.3	93.6	26.2
All Feat.	1000	100.0	100.0	100.0	99.3	69.9	29.0	99.4	67.9	45.5
	2000	100.0	100.0	100.0	99.4	66.2	35.5	99.5	78.9	57.9

Interpretation

As table 4 shows, J48 is able to learn a predictor for the time series problem documented in earlier sections that achieves 100% accuracy, precision, and recall on (10-fold) cross validation data. This performance is far superior to that of logistic regression and OneR, as well as compared to that of the atemporal predictor whose results are shown in earlier in this section.

CONTINUING WORK

Integrating relational and temporal features

Relational attributes include pair-dependent attributes such as mutual interests, friends, schools, etc. and link-dependent attributes such as causal explanations for a friendship (how two people met and what their relationship is) and reported outcomes of the friendship (what they self-identify as having done together).

There are two ways in which our time series and relational data models can be combined to produce an enriched overall data model for link change prediction:

1. Incorporate previously derived relational attributes into the time series. This can be done using classical time series analysis methods such as:
   a. windowing: expanding the definition of a tuple or "feature vector" (attribute vector) to include values of a feature at different time lags
   b. smoothing: finding an exponential trace or moving average of an attribute
2. Incorporate summative attributes of the time series into the relational model. This can be done using time series filters and aggregation methods such as:
   a. moment analysis: finding the mean, variance, skewness, and kurtosis of specific quantitative variables over time and using these moment values as attributes
   b. queuing theoretic parameter estimation: using statistical inference with a process model (e.g., Poisson or Weibull analysis) to find waiting times for events such as adding or deleting friends (or groups of friends)

We are continuing to add relational features cf. [HKPW07] that are part of the overall data model.

ACKNOWLEDGEMENTS

This research was supported in part by an NSF ITR and NSF FIBR grant. We thank Vikas Bahirwani for contributions to the third implementation of the feature construction system, Jan Crow for helpful discussions on statistical analyses, Todd Easton for suggestions on algorithms for graph feature analysis, and Daniel Andresen for recommendations on data acquisition and change prediction.

REFERENCES

Bhattacharya I., Getoor L., 2004, “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* (LinkKDD'04), Seattle, WA, USA.

Getoor, L. Diehl C.P., 2005, “Link mining: a survey,” *SIGKDD Explorations, Special Issue on Link Mining*, vol. 7, no. 2, pp. 3-12.

Holte R.C., 1993, “Very Simple Classification Rules Perform Well on Most Commonly Used Datasets,” *Machine Learning*, vol. 11, no. 1, pp. 63-90.

Hsu W.H., King A., Paradesi M. S. R., Pydimarri T., Weninger T., 2006, “Collaborative and Structural Recommendation of Friends using Weblog-based Social Network Analysis,” In *Proceedings of the 2006 AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs* (CAAW'06).

Hsu W.H., Lancaster J., Paradesi M. S. R., Weninger T., 2007, “Structural Link Analysis from User Profiles and Friends Networks: A Feature Construction Approach,” In *Proceedings of the 1st International Conference on Weblogs and Social Media* (ICWSM'07), pp. 75-80.

Ketkar N. S., Holder L. B., Cook D. J., 2005, “Comparison of graph-based and logic-based multi-relational data mining,” *SIGKDD Explorations, Special Issue on Link Mining*, vol. 7 no. 2, pp. 64-71.

Kubat M., Holte R., Matwin S., 1998, “Machine Learning for the Detection of Oil Spills in Satellite Radar Images,” *Machine Learning*, vol. 30, no. 2/3, pp. 195 - 215.

Kubat M., Matwin S., 1997, “Addressing the Curse of Imbalanced Training Sets: One-Sided Selection,” In *Proceedings of the 14th International Conference on Machine Learning* (ICML'97), pp. 179-186.

Mukherjee M., Holder L. B., 2004, “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* (LinkKDD'04), Seattle, WA, USA.

Popescul A., Ungar L. H., 2003, “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.