Link Prediction in the Pinterest Network

Poorna Kumar
poornak@stanford.edu

Amelia Lemionet
lemionet@stanford.edu

Viswajith Venugopal
viswa@stanford.edu

Abstract

Link prediction is a classic problem in networks, of great practical relevance – it can suggest to users what items they should buy in an e-commerce site, who they should follow in a social network, etc. In our reaction paper, we summarize and critique important papers dealing with this task, including [6], which talks about how to use network structures to compute link prediction *scores*, [2], which formulates link prediction as a supervised learning problem and [3], which proposes a novel Supervised Random Walk method to leverage the power of both node/edge attributes and network structure. We critique our readings, and brainstorm future research directions.

Our proposed project is to perform link prediction for the Pinterest network, which is a tripartite network of users, pins and boards. Specifically, we would like to recommend boards for users to follow, and pins to be added to boards. We plan to use methods from the papers we surveyed as a baseline, and propose enhancements that use the unique structure of the Pinterest network, and the available metadata, in order to do link prediction better. We discuss the problem, dataset, the algorithms we plan to use, and the evaluation metrics in our project proposal.

1 Reaction paper: The Link Prediction Problem

Link prediction is the task of predicting the existence of unobserved links or future links between pairs of nodes in a network, based on the existing connections in the network and the attributes of nodes and edges (when available). In this paper, we mostly focus on social networks. Social networks are usually sparse: most nodes in the network are connected to a small fraction of all the other nodes in the network. This presents a challenge, because a 'good' (albeit useless) prediction could even be to predict no new links at all. Another challenge is that it is often unclear how much of a person's interactions with other entities can be modeled as a function of network structure, and how much of it is because of the entity's and/or person's own intrinsic attributes. At the same time, link prediction in social networks has great utility: to recommend pins or boards to a user on Pinterest, to suggest friends on Facebook, or even to detect potential links between people in crime or terror networks.

In this reaction paper, we examine several papers dealing with link prediction, from surveys to very recent research, to understand the problem and the methods used to tackle it. We have divided our papers into three broad categories based on their main content; for each category, we present a brief summary, and then a critique of the relevant papers. After this, we examine some promising directions for further research, and brainstorm on interesting extensions we can pursue.

1.1 Link Prediction based on Similarity Scores

Given a network, $G(V, E)$, we seek to accurately predict edges that are currently not present in the network. Let x and y denote two nodes in G that are currently not connected, then we can assign a similarity measurement $score(x, y)$ to the pair $\langle x, y \rangle$ [2] [7]. We can thus produce a ranked list in decreasing order where the pairs with higher scores are more likely to be connected than those with lower scores. The difference in prediction methods based on similarity measurements resides in the way the score is computed.

At a high level, there are three groups of score measuring methods, and different methods work well for different problems. (i) Local similarity indices, where $score(x, y)$ depends only on information coming from x and y. (ii) Global similarity indices, which take into account all the paths in the network. Since they consider more information, they are naturally more accurate but they also tend to be very slow and thus not very well suited for large networks. (iii) Quasi-local indices, which are in between, and trade-off accuracy for computational complexity.

Liben-Nowell et al. [6] provides an overview of similarity based methods and focuses primarily on predicting edges that will show up in the future given the present state of a social network. The networks studied are collaboration networks between authors. The data was partitioned into training and test sets, and most of the evaluations were made by comparing the proposed predictors to a random predictor.

The authors present several methods (both local and global) to measure similarity scores between edges. The highest scores would then translate into more likely future collaborations (edges). Some of the methods presented significantly outperform the random predictor, which means there is useful information in the network – however, there was no one method which was uniformly the best across datasets.

Finally, they make the interesting finding that it is easier to predict edges when the network is very diverse. For example, when considering scientists from many different disciplines, it is easier to group them by discipline and thus predict better than random, whereas when all collaborators are from the same field, 'there seems to be a strong random component to new collaborations'.

Critique: This paper constitutes a very good reference in terms of theoretical foundations. It provides a general overview of similarity methods, and for this reason is a very good starting point to understand link prediction. However, it is quite narrow in the range of datasets and methods considered. It focuses only on academic collaboration networks, and does not talk about how the methods generalize to networks in different domains, or talk about whether there are differences when dealing with networks where we are trying to infer unobserved links as opposed to predicting future links. The methods discussed also do not incorporate additional information such as node or edge attributes in the prediction model.

1.2 Link Prediction as a Supervised Learning Problem

A significant body of work [2] [4] deals with the formulation of link prediction as a supervised machine learning problem. Formally, when snapshots of a network at different points in time are available, this is done by considering the network at times t and t', G_t and $G_{t'}$, where $t' > t$, and generating the training set as follows: we choose pairs of nodes (x, y) which are not connected by an edge in G_t as examples; the label is positive if the nodes are connected in $G_{t'}$, and negative if they are not. For each example, we can compute features based on x, y and G_t. These can be based on topological attributes, distance metrics or neighborhood-based metrics computed from the network, as well as node attributes from x and y.

Al Hasan et al. [2] uses this formulation to perform link prediction on academic co-authorship networks. Specifically, given a dataset of research articles published over a few years, they partition the articles based on publication year, into two sub-ranges: the first partition forms the training years, and the second forms the test years. They construct the author-author collaboration graph (adding an edge between two authors if they have written a paper together), first from just the training years, and then from all the years – these form G_t and $G_{t'}$ respectively, and they generate training examples using the procedure described above. They use a combination of several features – some are obtained from the network structure (number of papers, number of collaborators, number of secondary neighbors[1], shortest distance, clustering index), while others are domain-specific (number of matching keywords in their papers, classification codes of their papers). They then use several machine learning classification models, like decision trees [9], SVMs [10] and Naive Bayes [5], and discuss the performances of the different models.

Benchettara et al. [4] extends this work by proposing specific additional features to use in bipartite networks (an example is a user-item purchase network in an e-commerce website, where a user and an item are connected if the user purchased the item), or in unimodal projections of bipartite networks (where we construct a network out of one of the sets in the bipartite network, connecting nodes if they have at least m common neighbors – in our example, this could be a network of users, with edges linking users who have bought at least m items in common). The paper argues that there is extra information we can exploit in this case – concretely, if we are trying to predict the existence of a link between a user u and and an item a, for every 'direct' topological feature (like degree or clustering coefficient) that we use to derive a score between u and y, we also add an 'indirect' feature, where we compute the same feature between u, and other users that purchased a (and aggregate them by taking the mean, min or max, as appropriate), and also between a and the other items that u purchased. They show that, in various domains, augmenting the training set with these indirect features improves performance in the link prediction task.

Critique: Since the proposed framework transforms link prediction problems into supervised learning problems, it allows us to leverage advances in machine learning to do better link prediction. Further, when evaluating their features, the papers seemed to often use the same models with default hyperparameters on WEKA – it is possible that some of their feature sets were more suited to different models/hyperparameters. Also, this work does not naturally take into account the natural network structure of the data – instead, handpicked network statistics have to be extracted from the data for every node pair, and making the best choice of features is a difficult problem.

[1] secondary neighbors are neighbors of neighbors

1.3 Supervised Random Walks

In 2011, **Backstrom et al.** [3] proposed a new link prediction algorithm that would rely on the intrinsic network structure, as well as edge and node attributes, to predict new links in a social network. Their proposed solution was to use these attributes to guide a supervised random walk on the graph. Particularly, they formulated a supervised learning task to learn a function to assign strength to edges in a graph such that a random walker is more likely to visit nodes to which new links would be created in the future, and developed an algorithm to learn the edge strength estimation function.

Critique: This method acts as a bridge between two classes of methods: i) unsupervised methods that use purely the network structure of the graph to predict links (such as PageRank, which also employs random walks), and ii) supervised classification methods that construct features gleaned from edge, node and network attributes, and then predict whether a given node will form an edge to another node or not. By learning the edge strength function in a supervised manner, this approach neatly sidesteps the problem of having to pre-define edge strengths (a problem which is commonly seen in unsupervised random walks). At the same time, it allows us to use the intrinsic connected structure of the network, instead of having to select features based on network structure (as we do when we use binary classifiers to predict links). Thus, it appears to combine the best of both worlds.

However, this method is quite compute-intensive: the function that is optimized is generally non-convex. We must therefore learn the edge strength functions many times, each from a different starting point. It is not clear how many runs are required before we can settle on the best value of the weights. Additionally, it does not fully solve our feature selection problem: we still have to decide how to represent edge strength as a function of the edge and node features (up to the unknown weight parameters).

1.4 Brainstorming: Promising Future Research Directions

Based on the papers we read, the following are some possible directions for future research that we believe are promising:

- **Extensions for k-partite graphs:** In Section 1.2, we discussed how [4] improved over [2] by proposing specific enhancements for bipartite graph. We believe there is scope to extend this work for the general k-partite case. Concretely, when trying to predict whether an edge will form between two nodes x and y, for each feature we compute, [4] adds an 'indirect' feature, which is the same feature computed on the projected unimodal graphs derived from the bipartite graph. We believe this could extend in interesting ways to general k-partite graphs. We discuss one possible extension in our proposal.
- **Incorporating natural language information:** When these methods leverage natural language data, the features they construct are usually restricted to keyword matching. However, there are more sophisticated ways to account for semantic similarity between natural language features, such as word2vec [8], that could be incorporated to make these algorithms perform better.
- **Missing data:** Social networks are often extremely large, and there is a high chance of encountering missing node/edge data. For example, many users on social media websites do not disclose all their demographic information. While this is a common problem, it is not currently clear how to work with missing data in a social network, and it seems like there is scope for future work in this direction.

2 Project proposal

The project we propose is to perform link prediction on the Pinterest network, based on current edges and node attributes. In the following sections, we introduce the dataset and the problem, and then talk about the methods we propose to apply, and how we will evaluate our results.

2.1 The dataset

We will analyze the Pinterest dataset provided by Prof. Leskovec, which consists of food boards, the pins in them, and the users who created them. More concretely, the data consists of the following four tables:

- **Users**: Consists of a list of 18,578,352 user ids that will later be associated to pins and boards.
- **Boards**: For the 1,246,675 boards presented, contains information on board id, board name, id of the user who created the board, and time when it was created.
- **Pins**: A description of 273,643,843 pins, including: pin id, time when it was created, and id of the board to which it belongs.

- **Follows**: A list of 48,036,898 connections between boards and users. The data contains board ids, user ids and the time at which the user started following the board.

The user-to-board graph is a bipartite graph, and the board-to-pin graph is another bipartite graph. By stacking these two graphs, we get a tripartite graph, visualized in Figure 1.

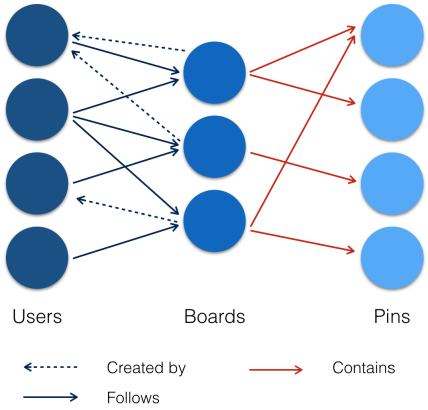

Figure 1: Illustration of Pinterest tripartite network

2.2 The problem

The tripartite structure of the network makes it very interesting to analyze – although there are no direct edges between users and pins, there is a clear relationship between them since a user creates (or follows) a board that contains pins. In this sense, whether we are thinking about analyzing user-boards edges or board-pins edges, we have more information that what is contained in either of the bipartite graphs alone.

Concretely, we will focus our work on predicting 'follow' links between users and boards and 'contains' links between boards and pins. Both these problems have obvious utility: it would be useful to help users discover boards that are interesting to them, and also to help users 'complete' their board by finding relevant pins for them to put on the board. Since the dataset consists of temporal data, we will separate our edges into two sets. One set will have edges created up to a certain time, and the other will have edges created after that time. We will then attempt to predict future links based on past information.

2.3 Methodology

Based on the readings presented in our reaction paper, we will try both the supervised learning model and random walk model. Let $G(V, E)$ be our entire network with node set V and edge set E, spanning from t_{start} to t_{end}. Let t_0 be an arbitrary time in between t_{start} to t_{end}. Our training graph consists of nodes and edges created before t_0, denoted by $V_{(0,t_0)}$ and $E_{(0,t_0)}$. Our test graph consists of the training nodes and edges, plus those edges that were created between nodes in the training set from time t_0 to t_{end}. Let us denote the edges in our test graph as $E_{(0,t_{\text{end}})}$. Then, the training graph is $G(V_{(0,t_0)}, E_{(0,t_0)})$ and the test graph is $G(V_{(0,t_0)}, E_{(0,t_{\text{end}})})$.

2.3.1 Supervised learning

Following the approach described in Section 1.2, we will tackle link prediction as a supervised learning problem; we will generate the labels just like previous work did, and we will use the following features:

- **Topological features**, like the Adar-Adamic distance [1], the distance between the nodes and other network features. Following the lead of [4], we will also augment these features by exploiting the k-partite nature of the graph. For example, when we are trying to decide whether to recommend a board b for a user u to follow, let's say that one of the features we use is the distance between b and u, $d(b, u)$. In this case, we augment the features with the following additional features. Let G^m_{BU} be the unimodal graph of boards, where two boards are linked if they have at least m users in common; we also have G^n_{BP}, where two boards are linked if they have at least n pins in common. Similarly, we have G^q_{UB} and G^p_{UP}, which are unimodal graphs for users. Then, we also add the following values as features (here, following the notation from [4], ψ is an aggregation operator – either max, min

or sum): $\psi_{x \in G^m_{BU}} d(b, x)$, $\psi_{x \in G^n_{BP}} d(b, x)$, $\psi_{x \in G^q_{UB}} d(u, x)$ and $\psi_{x \in G^p_{UP}} d(u, x)$. [4] showed that this approach significantly boosted performance for bipartite networks, and therefore we believe that this natural extension will work well for our k-partite network.

- **Metadata-based features:** although the users are anonymized, we have some metadata in the form of board names and descriptions. We propose to incorporate this into our model. There are a few ways to do this: we can cluster boards into sub-categories based on their names and descriptions, and incorporate the clustering information into our supervised learning; we can also incorporate the text directly, possibly by using vector representations for words. [8]

2.3.2 Supervised Random Walk

We also wish to explore the use of the methods described in Section 1.3. To predict links, we could use the following edge attributes:

- The age of the board, the average age of the pins on the board
- The board name and/or description: We can convert this text field to a feature either by vectorizing it via word2vec, or by using some other feature that can indicate similarity between names.
- The age of the edge itself.

2.4 Model assessment

As with all link prediction problems, our network is extremely sparse. Therefore, simply measuring the accuracy of our predictors does not make sense. Taking cues from the literature, we will use the following two measurements to assess our link predictors:

- We can use the AUC measure (for Area Under the ROC Curve [7]). This can be interpreted as the probability that a randomly chosen future edge in the test graph (which we want to predict) is given a higher score than a randomly chosen non-existent link in the test graph. Thus, we pick a missing link and a non-existent link and compare their scores repeatedly. Let n be the number of comparisons we make, n' the number of times the future edge has a higher score, and n'' the number of times they have the same score, then we can compute the AUC value as:

$$AUC = \frac{n' + 0.5n''}{n} \tag{1}$$

- We will also look at our predictions, and compute precision, recall and the F-1 score for the positive examples (that is, for the edges that do exist). This makes sense – because the graph is extremely sparse, finding positive examples is our challenge, and that is what we will evaluate our model on.

2.5 Potential Challenges

Some of the challenges we foresee when working on this problem are:

- The dataset is huge, so some of the metrics we wish to compute can become computationally expensive. It might be necessary to limit our analysis to a subset of the data
- When including text features (the board name and board description) in our model, we will have to find a good way to differentiate sentences that are very similar but not identical. In particular, since all the boards in our dataset are related to food, most of the board names are very close in meaning but have small nuances that might be helpful for our prediction.

References

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

[2] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning.

[3] L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 635–644. ACM, 2011.

[4] N. Benchettara, R. Kanawati, and C. Rouveirol. Supervised machine learning applied to link prediction in bipartite social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*, pages 326–330. IEEE, 2010.

[5] D. D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *European conference on machine learning*, pages 4–15. Springer, 1998.

[6] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

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

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[9] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. 1990.

[10] J. A. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.