Impact of Global Network on Localized Link Prediction

Alex Martinez

December 08, 2015

1 Introduction

The problem of link prediction and its variants have driven significant research efforts over the past decade. Real world applications, ranging from social networks (Facebook friend recommendation) to national security (detecting links in terrorist networks), have motivated interest in this problem. Link prediction can be reframed into other varieties, such as missing link prediction or link recommendation. We will explore the latter framing.

After considerable literature review, I found a lack of research into the impact of network structure when link prediction is constrained to a subset of the total network. One possible context of this situation is using performing link recommendation for a group of individuals at a networking event. Newly edges are formed exclusively between attendees, so what is the impact of utilizing external network information including nodes currently not present at the event. Does this information improve accuracy or does this external network play only a small factor in a local instance of link prediction?

With this motivation, we set out to modify existing link prediction algorithms for this special case and apply them to an example network. We utilized unsupervised and supervised random walk, as well as some similarity metrics and linear classifiers in our aim of measuring this impact.

2 Literature Review

Again, link prediction is a thoroughly explored research topic still experiencing advances. Most relevant to our work is Leskovic and Backstom's [2] algorithm for combining network structure and node and edge features. Briefly, this algorithm was built in response to the two primary approaches of link prediction at the time and their shortcomings. One approach views link prediction as a classification task, framing existing edges as positive training examples. Issues with this are class imbalance and feature extraction, primarily w.r.t. condensing non-linear network relationships into features. The second ranks nodes through network exploration (unsupervised random walk), but this fails to utilize node/edge attributes. Supervised random walk (SRW) builds on these approaches by combining network structure AND rich node/edge attributes. This is done by taking the unsupervised random walk model and utilizing a function f to map node/edge attributes to edge transmission probabilities. Given its reported accuracy improvement, we set out to implement this algorithm. Its implementation constituted the vast majority of this student project.

Other relevant work include supervised and unsupervised approaches to improve the performance of the mapping function f, as researched by Cukierski et al [1] and Feng et al [4]. I had planned on experimenting with deep learning in an attempt to account for hidden network attritubes, but that's now left for a future work.

With regards to other algorithms, Leskovec, countless other papers, and even this course cited the Liben-Nowell and Kleinberg experiment and subsequent advocation for the Adamic/Adar score as a high performance similarity metric. [10,12]

3 Methods

3.1 Data

Acquiring network evolution data in which future links are temporarily restricted to a subset of the graph proved impossible difficult to find. Additionally, since a major point of this experiment would be to capture the non-linear factors present when link prediction is constrained and how the global network relates to that, we thought at first that it would be inappropriate to take a time lapse data and manually remove all future nodes outside of an arbitrary node subset of interest.

However, collecting this anonymized social network snapshots from all involved participants of a networking event of any scale proved untenable given the constraints of this class project.

We then compromised by artificially removing a number of edges from the graph, and using link prediction algorithm to accurately predict those deleted edges (formulated this way, the problem is more missing link detection than future link prediction).

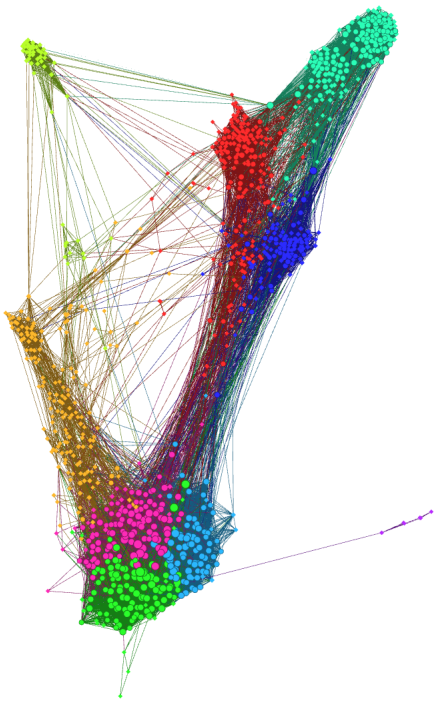

Figure 1: Facebook network visualization. Community partitions determined by modularity score

Because we could get the extremely desirable $\approx 174,000$ node Iceland Facebook graph used by Leskovic in his SRW paper, and because it was difficult to find a relevant dataset even without a time lapse, we ultimately settled for a much smaller Facebook ego network dataset [8].

3.2 Data Analysis

As node features are defined with respect to a particular node's ego network, we choose the largest ego network, which involves 1045 nodes and 53498 edges. Each node has associated with it 576 anonymized binary features (e.g. "education;concentration;id;anonymized feature 215"). For the first graph, we subsets of 400 nodes on which to restrict link prediction. Of those 400 nodes, there are 3308 edges. While we did experiment with different subset sizes, this size proved adequate in balancing divvying up internal and external network structure (A larger dataset would have allowed for a larger ratio of subset/external nodes).

Analyzing the global graph structure yielded some insight. Both the degree distribution (figure 1) and network visualization (figure 2) reveal a modest amount of preferential attachment, despite

the sample small size allowing for much noise.

3.3 Algorithms

3.3.1 Linear Classifiers: Random Forest Baseline

As baselines, we experimented with scikit-learn supervised classifiers Random Forest and Gradient Boosted Trees. Starting with a poor performance from RF, I was unable to tune GBT parameters for performance beyond that of RF. Surprisingly, I received better performance from linear regression which I anticipated being relatively worse at handling binary feature vectors.

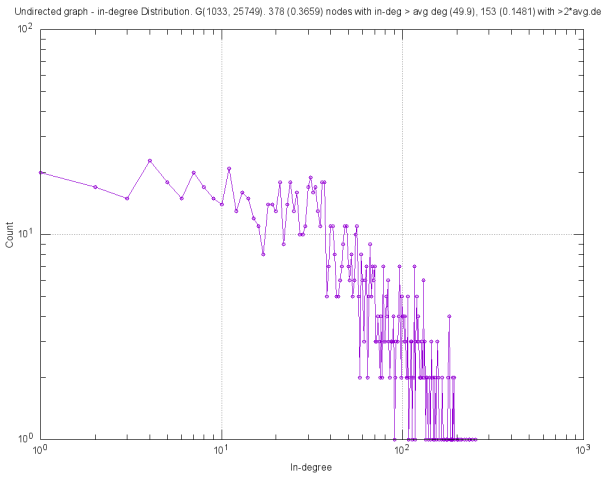

Figure 2: Degree distribution weakly follows a power law

3.3.2 Adamic/Adar and Jaccard Similarity Metrics

Similarity metrics typically comparing the neighbors of the two nodes that make up an edge. In the case of Adamic/Adar, we compute the score as follows:

$$\sum_{n \in CN(x,y)} \frac{1}{\log\left(k_n\right)}$$

Where $CN\left(x,y\right)$ is the set of all neighbors common to both x and y, and k_n is the degree of node n.

Although the literature review strongly suggests the Jaccard coefficient to be inferior, we'll include it as well:

$$\left| \frac{CN\left(x,y\right)}{AN\left(x,y\right)} \right|$$

Where $AN\left(x,y\right)$ is the union set of x and y's neighbors

3.3.3 Random Walk w/ Restarts (RWR)

RWR corresponds strongly with personalized pageRank. Without considering node/edge attributes and with undirected edges, pagerank would be statistically very similar to the degree distribution of the graph, as reflected by

$$D = \frac{1}{2|E|} \begin{bmatrix} p_1.degree \\ \vdots \\ p_N.degree \end{bmatrix}.$$

However, as SRW builds on RWR, we will briefly elaborate:

We are given a graph G, restart probability α, and a transition matrix S, in which $S_{u,v}$ indicates the probability of a jump from node u to v. Then a random walk with restarts is run from source node s. The probability of RWR remaining on each node at a step t is given by:

$$p^{t+1} = \left(1-\alpha\right) Sp^t + \alpha q_s$$

where q_s is a zero vector save for one 1 at the sth index. After enough iterations, $p$converges such that $p_{final} = p_{final}S$

Regarding link prediction, RWR is used to rank nodes by p_{final}. Top ranked nodes are then predicted as destinations of future links of s.

3.4 Algorithm: Supervised Random Walk (SRW)

RWR differs from SRW in the derivation of S. The former uses either a uniform distribution or some heuristic the skew the distribution rooted in graph properties such as degree, Adamic/Adar score, edge/node creation time, etc. The latter however, trains a function $f_w(\psi_{uv})$ that maps edge/node features, as represented by ψ_{uv}, to a transition probabilities s_{uv}.

The relaxed form of the optimization problem is as follows:

$$\min_w \qquad f(w) = \|w\|^2 + \lambda \sum_{d \in D, l \in L} h(p_l - p_d) \tag{1}$$

Where D contains positive training examples, L contains negative training examples, p_u is the RWR score of node u, λ is a regularization parameter, and $h(\cdot)$ is a loss function.

We will use a fairly complicated gradient-based optimization procedure to compute f_w. I'll include an algorithm description here for the sake of completeness, though I recommend looking at the work of Leskovic and Backstrom for more details [6].

3.4.1 SRW Gradient-Based Optimization.

For each source node s, we first build a random walk stochastic transition probability matrix Q' from our $f_w(\psi_{uv})$ function's current approximation of edge strengths.

$$Q'_{uv} = \begin{cases} \frac{f_w(\psi_{uv})}{\sum_t f_w(\psi_{ut})} & \text{if (u,v)} \in E \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

However, we also need to account for the probability α that a "restart" occurs, so we update the random walk transition probability matrix to

$$Q_{uv} = (1-\alpha) Q'_{uv} + \alpha q_s \mathbf{1}(v == s) \tag{3}$$

Where $q_s = \mathbf{1}(v == s)$

This is the Q we will use to update stationary distribution at time t p^t until we have a reasonable convergence of the following expression:

$$p^T = p^T Q \tag{4}$$

With these relationships established, let's return to the optimization problem. To minimize $F(w)$, we compute the gradients $\frac{\partial F(w)}{\partial w}$:

$$\begin{aligned} \frac{\partial F(w)}{\partial w} &= 2w + \lambda \sum_{l,d} \frac{\partial h(p_l - p_d)}{\partial w} \\ &= 2w + \lambda \sum_{l,d} \frac{\partial h(\delta_{ld})}{\partial \delta_{ld}} \left(\frac{\partial p_l}{\partial w} - \frac{\partial p_d}{\partial w} \right) \end{aligned} \tag{5}$$

Where $\delta_{ld} = p_l - p_d$.
...and $\frac{\partial p_u}{\partial w}$:

$$\frac{\partial p_u}{\partial w} = \frac{\partial\left(\sum_j p_j Q_{ju}\right)}{\partial w} = \sum_j Q_{ju}\frac{\partial p_j}{\partial w} + p_j\frac{\partial Q_{ju}}{\partial w} \tag{6}$$

To solve, we need expressions for $\frac{\partial Q_{ju}}{\partial w}$,$p$ and $\frac{\partial p}{\partial w}$.
The first expression analytically, the expression being,

$$\frac{\partial Q_{ju}}{\partial w} = \begin{cases} (1-\alpha)\dfrac{\frac{\partial f_w(\psi_{ju})}{\partial w}\left(\sum_k f_w(\psi_{jk})\right) - f_w(\psi_{ju})\left(\sum_k \frac{\partial f_w(\psi_{jk})}{\partial w}\right)}{\left(\sum_k f_w(\psi_{jk})\right)^2} & (j,u) \in E \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The remaining terms p_u and $\frac{\partial p_u}{\partial w}$ are recursively entangled. However, we can still compute the gradient iteratively. Here's one algorithm as descirbed in [6]:

3.4.2 Other Considerations

For the purposes of evaluation, we start off with the standards set by Leskovic and Backstrom as well, and experiment with other functions and parameters in later trials.

- Differentiable Loss function: Wilcoxon-Mann-Whitney (better performance than squared and huber loss)

$$h(x) = \frac{1}{1+\exp(-x/b)}$$

- Non-negative, differentiable edge strength function: Exponential edge strength (using sigmoid has little effect on performance)

$$a_{uv} = \exp(\psi_{uv} \cdot w)$$

- $\alpha, b = 0.3, 1e-4$
- $\lambda = 1$
- Constant bias feature

Initialize PageRank scores p and partial derivatives $\frac{\partial p_u}{\partial w_k}$:
  foreach $u \in V$ **do** $p_u^{(0)} = \frac{1}{|V|}$
  foreach $u \in V, k = 1, \ldots, |w|$ **do** $\frac{\partial p_u}{\partial w_k}^{(0)} = 0$
$t = 1$
while *not converged* **do**
  foreach $u \in V$ **do**
    $p_u^{(t)} = \sum_j p_j^{(t-1)} Q_{ju}$
  $t = t + 1$
$t = 1$
foreach $k = 1, \ldots, |w|$ **do**
  while *not converged* **do**
    foreach $u \in V$ **do**
      $\frac{\partial p_u}{\partial w_k}^{(t)} = \sum_j Q_{ju} \frac{\partial p_j}{\partial w_k}^{(t-1)} + p_j^{(t-1)} \frac{\partial Q_{ju}}{\partial w_k}$
    $t = t + 1$
return $\frac{\partial p_u}{\partial w}^{(t-1)}$

Figure 3: Iterator that computes p and $\frac{\partial p}{\partial w}$

4 Results and Findings

4.1 Evaluation Methodology

We half the 400 node subset into a training and testing set. The test set is evaluated on two standard performance metrics: the Area under the ROC curve (AUC) and the Precision at Top 20 (Prec@20).

Learning Method	AUC	Prec@20
Linear Regression: Node features	0.6684	2.45
Linear Regression: Node + Network features	0.6873	2.56
Random Forest	0.6798	2.31
GBT	0.609	2.18
Adamic/Adar	0.5290	1.66
Jaccard	0.5133	1.58
Random Walks||Degree	0.6243	2.34
Supervised Random Walk	0.6122	1.85

(a) Training on only the cropped node subset

Learning Method	AUC	Prec@20
Linear Regression: Node features	0.6342	2.37
Linear Regression: Node + Network features	0.6552	2.52
Random Forest	0.6023	2.13
GBT	0.5978	2.01
Adamic/Adar	0.5384	1.78
Jaccard	0.5190	1.60
Random Walks||Degree	0.6021	2.30
Supervised Random Walk	0.5878	1.86

(b) Training on the full network

Now to artificially generate two timestamps, my first instinct was to remove nodes randomly. However, graph analysis revealed a model of preferential attachment. Due to this, removing nodes randomly would take far more edges from high degree nodes than from low degree nodes. Link removal must reflect the link creation process. Fitting a power law to the distribution yields $x_{min} = 13$ and $\alpha = 1.748897$, which will be used as a heuristic to skew the random deletion of node subset edges.

4.2 Results

Note that all nodes came with a we had a 576 element binary feature vector. For edge same a_{uv}, linear regression trained on 4×576 dimension vector, which consisted of the concatenation of $features_u$, $features_v$, $xnor\,(features_u, features_v)$, and $xor\,(features_u, features_v)$. This slightly improved performance, relative to using 2×576 feature vector concatenation of only $features_u$ and $features_v$. In addition to these features, we included a bias term, adamic/adar score, $AN\,(x, y)$, $CN\,(x, y)$, and Jaccard Score to also boost performance.

However, SRW, didn't have the luxury of expanded training features, given how long it took to run the algorithm. Due to memory issues with just the 400 node set, I was forced to reduce the dimensions to just $xnor\,(features_u, features_v)$ features and the additional features. Unfortunately, training on the full network still proved difficult, so I batched the 576 $xnor\,(features_u, features_v)$ scores into the 26 feature types captured by the binary feature vector. To remain consist with the larger network, I reduced the feature dimensionality for the smaller network as well for a final feature count of 31.

5 Discussion/Conclusion

Relative to Prec@20 and AUC scores found in the literature review, my models appear to be underperforming across the board.

It appears that linear regression with the expanded feature set performs best. Regarding the random walk models, the unsupervised model appears to have much greater success. I could understand that RWR's lack of node/edge attributes can compromise the performance gain it receives from better account for network structure. As far as why SRW fail, I could only assume is was due to the reduce feature dimensionality imposed on me by computational constraints. The link prediction accuracies are so bad, in fact, that it's hard to determine the impact external network structure has on prediction, as I originally set out to understand. that more models perform worse on the full network model than the constrained model would suggest that the extra information actually impairs learning.

This conclusion should be taken with a grain of salt however, as I feel my experiment was compromised by so many factors that the results are near worthless. 1) I had to use synthetic data that utilized a simplistic heuristic to simulate the evolution of a graph between two timestamps. If there were any special relationships incoded in the node attributes, their anonymization of the binary feature vector name hindered my ability to intuit what that relationship could be. 2) Noise poses a serious threat to such a small dataset. 3) But even if I could have gotten a larger dataset, I'd had the same runtime and memory issues I had running SRW, issues that forced me to seriously compromise my node/edge feature space. With me having runtime issues with data of that size, I can't imagine how I would have process a dataset set large enough for noise to grow negligible, such as the $\approx 174,000$ node graph used in the original SRW paper. I think this is what accounts for the failure of SRW.

Looking at the literature review, it seems the relationship between other models reflect what was found empirically. Linear regression does seem to outperform decision trees, and RWR performs inbetween those two models. Additionally Adamic/Adar and Jaccard trail behind all of those, with the former holding a performance lead over the latter.

For future work, I suppose I can consult Professor Leskovic and discuss how to improve my models. Maybe even more importantly I can discuss code optimizations, so that I can not only run SRW with the full feature vector, but maybe even run it with the Iceland dataset, if he's somehow allowed to release it to me. Maybe after all that, I can finally resume work on determining the impact of external network data on a constrained link prediction problem. For now, I must deem this experiment as unsuccessful.

References

1. Cukierski, William, Benjamin Hamner, and Bo Yang. "Graph-based Features for Supervised Link Prediction." The 2011 International Joint Conference on Neural Networks (2011): n. pag. Web

2. Berend, Gabor, and Ervin Tasnadi. "Supervised Prediction of Social Network Links Using Implicit Sources of Information" (2015) n. pag. Web.

3. Youyou, Wu, Michal Kosinski, and David Stillwell. "Computer-based Personality Judgments Are More Accurate than Those Made by Humans." Proceedings of the National Academy of Sciences Proc Natl Acad Sci USA 112.4 (2015): 1036-040. Web.

4. Liu, Feng, Bingquan Liu, Chengjie Sun, Ming Liu, and Xiaolong Wang. "Deep Belief Network-Based Approaches for Link Prediction in Signed Social Networks." Entropy 17.4 (2015): 2140-169. Web.

5. Kim, Myunghwan, and Jure Leskovec. "The Network Completion Problem: Inferring Missing Nodes and Edges in Networks." Proceedings of the 2011 SIAM International Conference on Data Mining (2011): 47-58. Web.

6. Backstrom, Lars, and Jure Leskovec. "Supervised Random Walks." Proceedings of the Fourth ACM International Conference on Web Search and Data Mining - WSDM '11 (2011): n. pag. Web.

7. Katsimpras, Georgios, Dimitrios Vogiatzis, and Georgios Paliouras. "Determining Influential Users with Supervised Random Walks." WWW '15 Companion (2015): 787-92. Www2015. 24th International Conference on the World Wide Web. Web.

8. Jure Leskovec and Andrej Krevl, "SNAP Datasets: Stanford Large Network Dataset Collection" Jun 2014

9. Facebook Recruiting Competition. Kaggle. https://www.kaggle.com/c/FacebookRecruiting

10. D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In 12th CIKM, pages 556-559, 2003.

11. Wang, Hongzhi. "Entity Resolution on Multiple Relations." Innovative Techniques and Applications of Entity Resolution. N.p.: n.p., n.d. N. pag. Print.

12. Lada A. Adamic and Eytan Adar. Friends and neighbors on the web. Social Networks, 25(3):211–230, July 2003.