Tess Wolossow

Ria Agarwal

CSCI-4974

Prof. George M Slota

Nearby Neighbors Link Prediction Algorithm

Introduction

In graph mining, one fundamental task is link prediction, which aims to predict the likelihood of a connection (or link) between specific nodes in a given network. One conventional approach, the common neighbor method, considers the immediate shared neighbors of nodes. However, to enhance the predictive accuracy algorithm, we investigate whether expanding the scope beyond direct neighbors can provide better predictions. This project addresses the question: Can we refine the common neighbor link prediction algorithm to account for not just direct neighbors but neighbors of neighbors (and neighbors of neighbors of neighbors, etc.) and use this as a better link predictor? Our approach will explore connections up to k=2 hops away from each node. This will enable us to capture more nuanced relationships in the network, potentially offering more accurate predictions of future edges.

Graph mining revolves around extracting information and patterns from graph networks. Link prediction serves as a model for understanding network evolution and structure. By advancing link prediction methods to encompass higher-order relationships, we not only push the boundaries of graph mining, but also allow for deeper analysis of the growth of any social network, recommender systems, etc.

The ability to predict links beyond immediate neighbors has broad implications. In social networks, predicting future friendships or collaborations can aid in community detection and targeted marketing. In recommender systems, understanding more complex associations between items or users can increase the accuracy of recommendations. Furthermore, in biological networks, predicting interactions between proteins or genes based on extended connectivity can contribute to drug discovery and understanding disease mechanisms. Thus, refining the common neighbors link prediction algorithm to consider higher-order relationships could have impact across multiple domains.

In this paper, we will delve into the methodology of refining the common neighbor algorithm, evaluate its effectiveness, and discuss the results of our experiments.

Background

For the proposed nearby neighbor link prediction algorithm to make sense, it's important to first understand basic link prediction methods and the significance of higher-order connectivity in networks.

The common neighbor algorithm is a simple but effective method for link prediction. Given two nodes *u* and *v* in a graph *G*, the algorithm predicts a link between them based on the number of common neighbors they share. The more common neighbors the two nodes have, the higher the likelihood of a future connection between them. The common neighbors algorithm can be expressed as $C = |N(u) \cap N(v)|$.

The Jaccard algorithm is a similar link prediction metric that compares the intersection of the neighbor sets of *u* and *v* to the union of their neighbor sets. The Jaccard coefficient can be expressed as $J = |N(u) \cap N(v)| \, / \, |N(u) \cup N(v)|$.

In network analysis, higher-order connectivity refers to exploring links or connections beyond direct neighbors. This includes neighbors of neighbors (2-hop connectivity), neighbors of neighbors of neighbors (3-hop connectivity), and so on.

Methodology

In this study, we implemented and evaluated link prediction algorithms using Python and the NetworkX library. The methodology involved the following steps: data retrieval, graph creation, algorithm implementation, and experimental evaluation.

The datasets we used were gathered from two sources. The collaboration network of authors was derived from a GML file (cond-mat.gml), representing collaborations among authors in the field of condensed matter physics retrieved from the arXiv repository. In this graph, each vertex represents an author in the archive, and each edge represents a collaboration between two authors. While the data itself is formatted as a weighted graph, where each edge weight corresponds to both the number of papers collaborated on and the number of co-authors involved, we treated it as a simple unweighted network. This dataset was read in using the NetworkX read_gml() method.

The second dataset was sourced from IMDb, which provides the data for personal or non-commercial use. The data files were downloaded from a publicly available website called https://developer.imdb.com/non-commercial-datasets/. The data given associates movie titles with their directors and writers. For each movie, we extracted these relationships to create two graphs, one of movie directors and one of movie writers. For both graphs, each vertex corresponds to a director or writer, and each edge represents a collaboration between the two nodes. As with the authors network, both of the IMDb networks were represented as unweighted graphs. We focused on a subset of movies, such as horror films, identified by their unique identifiers (IDs) stored in horrortitles.txt. We also created a file (using shell scripts to cut the relevant fields and choose only the movies/shows that had a comma separated list of more than one writer or director) for various other genres with similar format to horrortitles.txt (i.e.

biographytitles.txt, scifititles.txt and westerntitles.txt). The number of vertices ranged from 24,000 to 64,000 (depending upon the genre) and edges ranged from 86,000 to 432,000.

In our study we implemented four link prediction methods. The random links algorithm assigns random predictions (0 or 1) to potential edges, representing a baseline prediction approach. In this case, an assignment of 1 means an edge is predicted to exist, while 0 means it is not predicted. The common neighbors algorithm calculates the number of common neighbors between pairs of nodes within the graph data. The nearby neighbor algorithm extends the concept of common neighbors to include neighbors of neighbors. It explores indirect relationships up to a depth of k=2 from each node. The nearby neighbors algorithm can be expressed as $N = |N(u) \cap N(v)| + 0.25 * |N(N(u)) \cap N(N(v))|$

There were several steps involved in evaluating each of the algorithms. First we generated a subset (G1) of the main graph (G) containing the first 25% of the edges sorted in temporal order. This subset served as the basis for evaluating link prediction algorithms. Then the compute_link_prediction function was applied to calculate link prediction values each of the different algorithms within the constructed graph subset (G1). The predicted edges generated by each algorithm were then compared against the actual edges in the original graph (G). Precision metrics were calculated by determining the proportion of correctly predicted edges (true positives) out of a specified testing size (a third of the total degree size of G), representing a subset of edges for evaluation.

Results and Discussion

As seen in the table below, a random link prediction was always the worst out of all the methods by a factor of about 50. The best predictor for all 3 datasets was the Jaccard method, on average about twice as accurate as the common neighbors method. Our newly invented method, nearby neighbors, did about 25% worse than the common neighbors method.

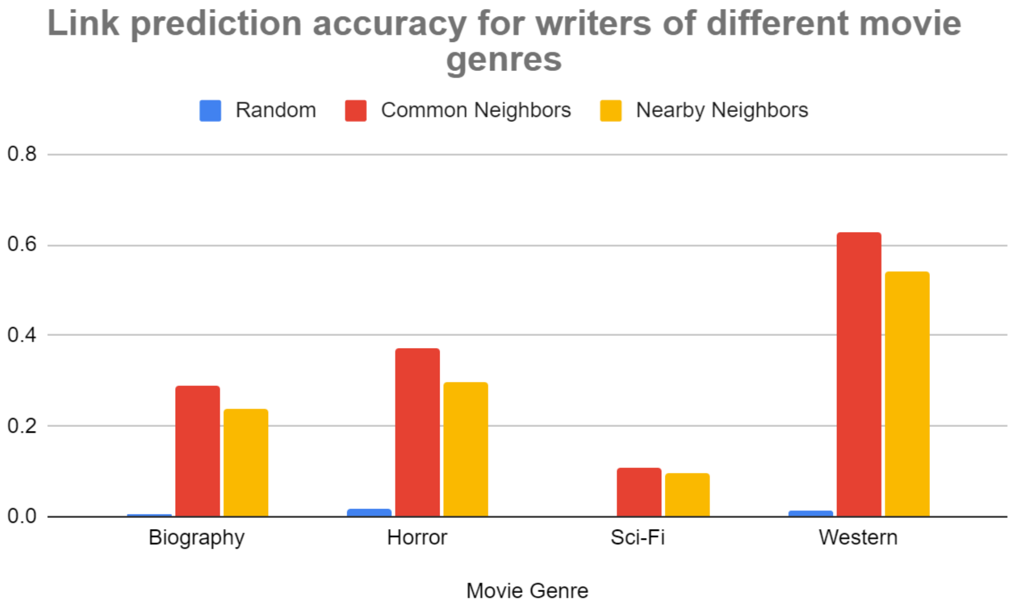
One interesting result was the difference in link prediction for the different datasets. Screenwriters were easier to predict than Movie/TV Directors and these were both more predictable than the future collaboration of authors of the condensed matter physics papers. It is possible the community of film writers and directors are more collaborative, since they tend to live nearby or attend more award shows/conferences and network better.

Another interesting result is that the nearby neighbors algorithm was more effective as the size of the network increased. The film writers network was the largest and also had the most accurate prediction, something that turned out to be true for all four link prediction methods tested. It might be interesting to look into whether the nearby neighbors method's effectiveness would continue to increase with larger datasets, and especially whether it would eventually be equally or more accurate compared to the common neighbors method.

We also compared to the difference in link (future collaboration) predictability based upon the genre of a movie/TV show. In the chart below you can see that screenwriters for Western movies were much more predictable than, for example, screenwriters of Sci-Fi movies.

Link Prediction Results for cond-mat and imdb datasets

	Random	Common Neighbors	Jaccard	Nearby Neighbors
Authors	0.0005	0.0871	0.2018	0.0528
Directors	0.0022	0.0958	0.1804	0.079
Writers	0.0155	0.3715	0.4518	0.3104



Conclusions

The Jaccard method was consistently the most accurate predictor across all datasets, showcasing its effectiveness in capturing network similarities. Common neighbors performed reasonably well but was surpassed by the Jaccard method, indicating the importance of considering shared neighbors in link prediction. Our nearby neighbors approach demonstrated slightly lower accuracy, but might be improved by further optimization or parameter tuning (i.e. setting the weighting factor of each nearby neighbor to less than 0.25 improved the predictability but was still not as good as just using common neighbors).

Collaboration predictability varied across different datasets. Screenwriters exhibited higher predictability even compared to movie directors, which could reflect potential differences in collaboration patterns within creative industries. Authors in the condensed matter archive had the lowest predictability.

The correlation between dataset size and nearby neighbors' effectiveness suggests scalability potential for this algorithm with larger networks. Future studies could explore its performance in handling increasingly complex datasets.

One future avenue for research would be to investigate the ideal weighting factor for incorporating nearby neighbors into link prediction models. In this study we used a factor of 0.25 for the 2-hop neighborhood, but further testing might reveal a more accurate factor. Another possible area of research would be to extend the analysis by incorporating additional datasets, such as movie actors or other industry professionals, to determine whether the patterns observed in this study would hold true.

Perhaps the most pressing area for future study would be to find strategies to optimize the computation of neighbors of neighbors in order to avoid the $O(N^2)$ complexity of our current algorithm and increase its flexibility and scalability, especially for use in testing with larger datasets. We could consider other data representations, such as adjacency matrices, to optimize storage and hopefully decrease computational overhead. Alternatively, we could try to implement parallel programming techniques (e.g., using mpi4py or Google Codelabs) to parallelize computation tasks, once again to decrease the total computation time needed.

In conclusion, our study underscores the importance of algorithm refinement, dataset diversity, and computational scalability in link prediction research. With these recommendations, future investigations can possibly help develope more accurate, scalable, and flexible link prediction methods applicable across industries, from academic collaborations to creative industries.

Citations

Liben-Nowell, D., & Kleinberg, J. (2003). The link prediction problem for Social Networks. Proceedings of the Twelfth International Conference on Information and Knowledge Management. https://doi.org/10.1145/956863.956972

M. E. J. Newman, The structure of scientific collaboration networks, Proc. Natl. Acad. Sci. USA 98, 404-409 (2001).