6-Link prediction

Complex Network Analysis Course
Reza Rezazadegan
Shiraz University, Spring 2025
https://dreamintelligent.com/complex-network-analysis-2025/

What is link prediction

Two ways of thinking about link prediction:

- **Finding the hidden or missing links (edges) in a network.**

Infer missing links from an observed network. The prediction of missing links is mostly used to identify lost or hidden links, such as inferring unobserved protein-protein interactions.

- **Predicting which links will be added to the network in future. (Network Evolution)**

In this scenario, the link prediction task is applicable to predicting future friendship or future collaboration, for instance, and it is also informative for exploring mechanisms underlying network evolution.

Link prediction methods are often based on a measure of similarity between the nodes. Thus, we studied *node importance* in the last section and we study *node similarity* in this chapter.

Applications of link prediction

Social Networks

Predicting friendships, professional connections, and interactions in social media platforms. Algorithms analyze common friends, shared interests, and interaction frequency to suggest new connections.
Example:

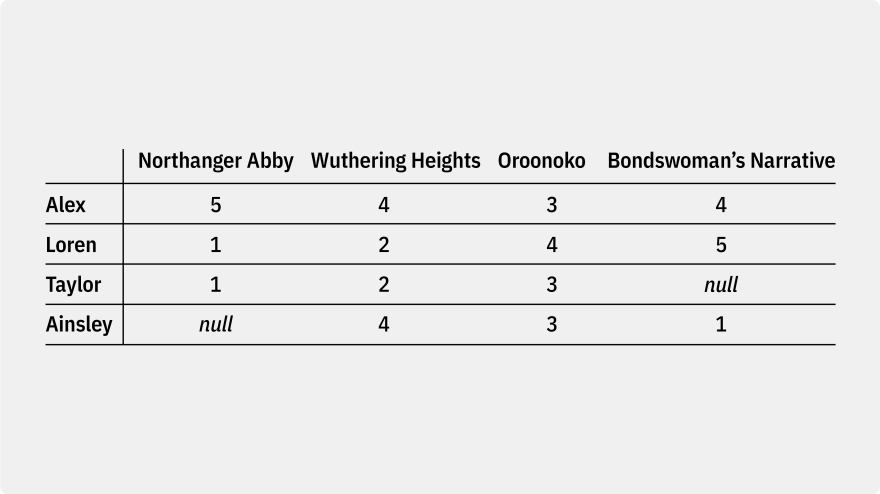
- Facebook and LinkedIn use link prediction to suggest new friends or professional connections.
- X and Instagram recommend accounts to follow based on mutual followers and engagement patterns.

Recommender systems

Note: this is one of the student presentation topics.

Suggesting products or services to customers.
Networks of users, products, and purchase behaviors are analyzed to predict which products a user is likely to buy.

Collaborative Filtering is a common method used in recommender systems, based on the principle that *similar users share similar interests*. Such methods recommend items to users based on how other users with similar preferences and behavior have interacted with that item. A matrix of user behaviors (such as ratings, purchases, etc.) towards items is used to measure user similarity.



	Northanger Abby	Wuthering Heights	Oroonoko	Bondswoman's Narrative
Alex	5	4	3	4
Loren	1	2	4	5
Taylor	1	2	3	*null*
Ainsley	*null*	4	3	1

Image source: IBM Think

Example:

- Amazon and Netflix use collaborative filtering, a form of link prediction, to recommend products and movies based on previous user interactions.
  - Larry Hardesty, The history of Amazon's recommendation algorithm
  - Instagram's algorithm in 2025 is leveling up with Predictive Artificial Intelligence
- Spotify predicts which songs a user might like based on shared listening behavior with similar users.

    - The Inner Workings of Spotify's AI-Powered Music Recommendations: How Spotify Shapes Your Playlist
- Ying et al. used graph neural networks to improve Pinterest recommendations.
    - Ying et al. Graph Convolutional Neural Networks for Web-Scale Recommender Systems, 2018
    - Wang, et al., Neural Graph Collaborative Filtering, 2020

Drug discovery and predicting drug side effects

Identifying potential

- drug-target interactions (DTIs),
- protein-protein interactions (PPIs),
- gene-disease associations.

How it works: Biological entities are modeled as networks where nodes represent proteins, genes, or compounds, and edges represent known interactions. Link prediction finds missing links to suggest potential interactions.

Example:

- Machine learning models predict new drug interactions, expediting drug repurposing. *HetioNet*, a biomedical knowledge graph, has been used to predict new disease-drug associations.
    - Bang, et al., Biomedical knowledge graph learning for drug repurposing by extending guilt-by-association to multiple layers, Nature, 2023
- Zitnik, et al., Modeling polypharmacy side effects with graph convolutional networks, Bioinformatics, 2018

Completing knowledge graphs

Filling missing facts in knowledge graphs like Google Knowledge Graph and Wikipedia.

How it works: Link prediction identifies new relationships between entities in knowledge bases.
Example:

- Google's search engine uses link prediction to complete missing entity relationships.
- How Google Knowledge Graph Works

Academic Research and Collaboration Networks

Application: Recommending collaborations between researchers based on past co-authorship networks.
How it works: Researchers are nodes, and co-authored papers form edges. Algorithms predict new collaborations by analyzing shared citations, affiliations, or research topics.
Example:

- Google Scholar and ResearchGate suggest potential co-authors based on common publications.
- Pavlov, Ichise, Finding Experts by Link Prediction in Co-authorship Networks, 2007

Fraud Detection and Anomaly Detection

Detecting fraudulent financial transactions, cyber threats, and illicit activities.
How it works: Fraudulent accounts often have different linking patterns than legitimate users. Link prediction detects unusual or missing links that indicate suspicious behavior.

Example:

- Venkatesh Ramanathan, Application of Graph Convolution Algorithms at PayPal, 2020

Community Detection and Network Growth Modeling

Application: Identifying groups or clusters within networks and predicting how networks evolve.
How it works: Link prediction helps discover hidden relationships, leading to better understanding of communities.
Example:

- In marketing, businesses identify customer segments and target them effectively.
- Mohan et al. (2020) proposed a link prediction-based approach for scalable community detection.

Link prediction in the Patent Citation Network

Methods for link prediction

Link prediction methods are often based on a similarity measure between nodes, meaning that similar nodes are predicted to have a hidden link.

Node similarity measures can be divided into: Local neighborhood-based, global neighborhood-based and random walk-based. There are also similarity measures that are based on a network embedding (representation).

Local neighborhood overlap measures

These measures are a function of the number of common neighbors of the two nodes. So, if u, v are two nodes we consider $N(u) \cap N(v)$. There are a number of measures based on this quantity:

Jaccard index

$$S_J(u,v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}$$

Exercise: Consider the 15th Florentine family network. Compute the Jaccard index for all node pairs by hand. Then for different values of the similarity threshold, say, 0.3, 0.5, 0.8 compute:

- Which new links are added to the graph.
- Which ones of the original links of the graph are "predicted".
- Draw the resulting graph by hand.

Adamic-Adar index

This index gives more weight to

$$S_{AA}(u,v) = \sum_{w \in N(u) \cap N(v)} \frac{1}{\log k_w}$$

Link prediction using local overlap means that we expect the nodes which have common neighbors to be connected to each other as well.

One limitation of these measures is that they consider only local neighborhoods and e.g. if the two nodes have no common neighbors, their similarity becomes zero.

Global Neighborhood overlap: Katz index

The difference between the local and global similarity measures is similar to the difference between node degree and node centrality.

These global neighborhood methods often have much better performance than local neighborhood ones, however they are computationally more difficult to compute.

Katz Similarity

The Katz measure we defined in the last chapter, can be used a s similarity measure as well:

$$S_{Katz}(u,v) = K(u,v) = \sum_{\gamma} \alpha^{l(\gamma)}$$

Here the sum is over the paths γ between u, v and $l()$ is path length. $0 < \alpha \leq 1$ is a fixed number.

Normalized Katz similarity

One problem with Katz measure is that it is higher for nodes of higher degree. For this reason we normalize Katz measure.

Normalization is a method which is often used in network theory, in which a quantity is compared to the expected value of the same quantity among all the networks which have the same degree distribution as the original network.

More precisely, imagine we have a network G and $K(u,v)$ is the Katz index of two nodes in G. We consider *randomizations* of G in which the links are rewired in such a way that degrees of the nodes re preserved. Then we can measure the expected value $E[K(u,v)]$ among all these randomizations. We can then either take the z-score of $K(u,v)$ or simply normalize:

$$\frac{K(u,v)}{E[K(u,v)]}$$

This normalized measure tells us how the actual quantity compares to its expected value. If it is greater than one, it means that the Katz similarity of u, v is higher than what one expects in a random network.

The expected value can be obtained using one of the following methods:

- Constructing a number (say, a thousand) randomizations of the network and computing the expected value (and standard deviation) of the quantity.
- Computing or approximating the expected value.

Exercise: as an example of the second approach, show that if A is the adjacency matrix of the network then

$$E[A(u,v)] = \frac{k_u k_v}{2L}.$$

Leicht, Holme, and Newman used the fact that the growth of the number of paths can be approximated by the largest eigenvalue λ_1 of A. This gives:

$$E[A^k(u,v)] \simeq \frac{d_u k_v \lambda_1^{k-1}}{2L}$$

$$S_{LHN} = \delta_{u,v} + \frac{2L}{d(u)d(v)} \sum_{k=0}^{\infty} \alpha^k \lambda_1^{1-k} A^k(u,v)$$

Random walk methods: Personalized page Rank

Simply put, we can use the probability of reaching u by a random walk starting at v as a similarity measure between u, v. It can be described in terms of Page Rank as follows. Remember the stochastic matrix M from the last chapter.

$$Q_u = \beta M Q_u + (1-\beta) e_u$$

$e_u = (0, \ldots, 1, \ldots, 0)$ with 1 at the position of the node u.
$Q_u[v]$ the stationary probability of reaching v by a random walk starting at u.

$$S_{RW}(u,v) = Q_u(v) + Q_v(u)$$

Supervised link prediction methods

Although working well in practice, the above methods have strong assumptions on when links may exist. For example, the common neighbor methods assume that two nodes are more likely to connect if they have many common neighbors. This assumption may be correct in social networks, but is shown to fail in protein-protein interaction (PPI) networks, i.e. two proteins sharing many common neighbors are actually less likely to interact.

Another drawback: they do not make use of (explicit) node features (say, age, gender, etc.).

Methods based on network embedding

Imagine we have a graph $G = (V, E)$ and an embedding or (*representation*) $\phi : V \to \mathbb{R}^n$.
Note that we have the shortest path distance on V and the Euclidean distance on $\mathbb{R}^m$. We are interested in embeddings that send nearby nodes in V to nearby vectors in the Euclidean space. Such embeddings can be obtained using random walk methods (such as Node2Vec or DeepWalk, which among your presentation topics) or **Graph Neural Networks** (discussed in later chapters).

Roughly speaking, these methods *learn* a rule for predicting links, instead of assuming one from the beginning.

Note: the coordinates of $\phi(u)$ are called the **latent features** of v, while the attributes of the node (say, age, gender, income, etc.) are called **explicit features**.

For now imagine we have such an embedding ϕ. If there is an edge between two nodes u, v, it maps them to nearby vectors. Thus, it is reasonable to expect that if ϕ maps two nodes a, b to nearby vectors then there may be a hidden link between them!

$$s_\phi(u, v) = \frac{\langle \phi(u), \phi(v) \rangle}{||\phi(u)||||\phi(v)||}$$

Classifier-based methods

In these methods, the similarity measures mentioned above are used as features to train a classifier such as Support Vector Machine, KNN, Decision Tree, etc.

Other methods

Generative models

Unlike traditional discriminative approaches, generative models aim to reconstruct the entire graph, capturing both observed and potential links.

- **GraphLP:** This model utilizes the feature learning capability of deep-learning architectures to automatically extract structural patterns for link prediction, assuming that real-world graphs are not locally isolated. It explores high-order connectivity patterns to leverage hierarchical structures within graphs.
  - Xian, et al., Generative Graph Neural Networks for Link Prediction, 2022

Ensemble models

- **Stacked Models:** By systematically evaluating numerous link prediction algorithms and combining them into a single model, stacked approaches achieve nearly optimal accuracy across diverse real-world networks. This method leverages the strengths of individual predictors to enhance overall performance.
  - Ghasemian et al., Stacking models for nearly optimal link prediction in complex networks, PNAS, 2020. https://github.com/Aghasemian/OptimalLinkPrediction

Non-GNN

- **Gelato:** This topology-centric framework applies a topological heuristic (i.e. Autocovariance which is based on random walks) to a graph enhanced by attribute information via graph learning. Trained with an N-pair loss on an unbiased dataset, Gelato addresses class

imbalance and achieves superior accuracy and efficiency compared to state-of-the-art GNNs.

- Huang et al., Link Prediction without Graph Neural Networks, 2023

Evaluating link prediction

- **First problem:** how can we evaluate a link prediction model?
  Note that a lot of link prediction methods are based on a node similarity measure $s(u, v)$ and to decide whether there is a link between u, v we must first choose a threshold and then declare that nodes whose similarity is greater than ϵ are connected by a link. In the classifier-based methods, the classifier produces a probability and we need a threshold again.
  Note that:
- **Second problem:** We need a method for choosing the value of ϵ.
- Not all the "real" links in the network are "predicted" to exist in this method. (They might be regarded as "suspicious links".)

Evaluating link prediction

Mote that link prediction can be thought of as a *binary classification problem*, i.e. for each pair of nodes u, v in the graph, we need to decide whether there should be a link between them (positive case) or there should not (negative case).

For more on classification in Machine Learning you can look at the slides of my Machine Learning course.

A random sample of the links (say 90%) is used for training the model. (Although some models such as similarity-based ones do not need training.)
The rest of the links are used for testing.

We need negative sampling as well! This means choosing a random sample of the negatives class (non-existing links).

We choose a sample of existing links (positives) and non-existing links (negatives) for training the model and then test the model on the remaining set.
This way we get four cases: true/false positives and true/false negatives.

One issue: class imbalance, i.e. there are much more non-existing links in a network compared to the existing ones.

- Wu et al., Link Prediction on Complex Networks: An Experimental Survey, 2022
- Yang et al., Evaluating Link Prediction Methods, 2015

Choosing the similarity threshold

We can find a value for the threshold that gives the best trade-off between true positives and true negatives.

More

Predicting the *whole* of the links in a graph.

Predicting the links for a *new* node: related to the above problem.

These two problems are related to network evolution and network generation discussed in future chapters.

Link prediction workshop

In this workshop we pick a link prediction algorithm such as Jaccard index and, using the Networkx library, apply it to a network such as the network of characters of Les Miserables (which is available in Networkx).
Increasing the threshold from 0 to 1, we evaluate the algorithm and also qualitatively analyze whether the predicted links make sense and whether the old links in the network are predicted or not.

Time permitting we will do the same with a more advanced link prediction algorithm.

The Jupyter notebook for this workshop is available on course Github.