Link Prediction : A Survey

Introduction:

As part of the research on large and complex networks, a certain amount of attention was given to social networks-which are structures whose nodes represent people or entities of a network and the relationship between them represents collaboration or interaction or influence between the entities. Social networks are dynamic in nature; they grow and change quickly over time by the addition of new interactions between the entities of the network. Understanding how they evolve is the main goal here. Link prediction problem studies and defines a computational problem underlying the social network evolution.

Concept:

Link prediction is given a snapshot of a network, can we infer any new interactions among its members which are likely to occur in the near future [1]. There are many factors that can be considered while predicting links between entities of a network that do not exist currently. These factors may be external or internal to the network. For example, two authors in a collaboration network who do not know each other and do not have any short chain of acquaintances may start collaborating in the near future, if one of the authors move to a different institution geographically located where the other author works. In this case, the chances of them collaborating increases. These factors are external to the network. Predicting links using external factors such as these is a difficult task. However, there are factors that are internal to the network using which predicting links become much easier. These methods use network topology to predict links.
Link prediction has applications in various fields like in large organizations where predicting promising links between its employees helps in the development of the organization. In monitoring terrorist networks where links between individuals can be predicted even though no interactions are observed between them. It also finds applications in monitoring and controlling computer viruses that use email as a vector. It can be used to provide recommender systems, to predict unobserved links between protein-protein interaction networks in biological system.

Problem Description:

Consider a graph $G\ (V, E)$ where V is the set of nodes and E is the set of links in the graph G. Multiple links and self connections are not allowed in G. Let U represent the universal set consisting of all possible links. Then, the set of non-existent links is U-E. The task of link prediction is to predict the missing links or links that may occur in future in the set U-E.

Metrics:

In order to check the accuracy of links predicted, the observed links E is divided two parts: The training set, E_T and the probe set, E_P. The training set is used as known information while the probe set is used for testing. No information of the probe set is used for prediction purposes. It is now very clear that,

$$\text{i) } E_T \cup E_P = U$$
$$\text{ii) } E_T \cap E_P = \Phi$$

There are two standard metrics used to test the accuracy of the prediction algorithms: Area Under the receiver operating characteristic Curve (AUC) and Precision. Every prediction algorithm assigns score to links in $U\text{-}E_T$ depicting the likelihood of its existence. These scores are arranged in decreasing order such that given a particular link; its occurrence is more likely than the link below it in the ordered list. The AUC evaluates the algorithms performance based on the overall list while Precision focuses on only L links with top scores.

i) AUC:

The AUC value gives the probability that a random chosen missing link (a link in E_p) is given higher score than a randomly chosen non-existent link (a link in $U\text{-}E$). During algorithmic implementation, we randomly select a non existing link from $U\text{-}E$ and a missing link from E_P to compare their scores. If among n independent comparisons, there are n' times missing link having higher score and n" times they have same score, then the AUC value is,

$$\text{AUC} = \frac{n' + 0.5n''}{n}$$

If all scores are generated from identical distribution then the value of AUC will be 0.5. Thus, the degree to which the value exceeds 0.5 indicates how better the algorithm performs compared to pure chance.

ii) Precision:

The Precision value is defined as the ratio of relevant items selected to the number of items selected. That is, if we take top L links as the links predicted, among which L_r links are right (L_r links are present in E_P) then the value of precision is,

$$\text{Precision} = \frac{Lr}{L}$$

Clearly, higher the precision value higher the prediction accuracy.

Classification of Prediction Algorithms:

Based on the methods employed for prediction of links, they are classified into three categories,

i) Similarity Based Algorithms
ii) Maximum Likelihood Methods
iii) Probabilistic Models

Similarity Based Algorithms:

The simplest of all prediction algorithms is the similarity based algorithms where every pair of nodes x and y, are assigned a score S_{xy} which directly defines their similarity. All non-observed links are ordered according to their similarity score and links connecting similar nodes are expected to have higher existence likelihoods.

Similarity between the nodes can be defined using the attributes of the nodes i.e., two nodes are more similar if they have many common features. However attributes of nodes are generally hidden and thus to define the similarity of nodes we concentrate on the structural similarity between the nodes which depends on the network structure.

Similarity based methods can be classified in a number of ways be it, local vs global, parameter-free vs parameter-dependent or node based or path based. They can also be classified as structural equivalence where the assumption is that the link indicates similarity between its end points, and regular equivalence where the assumption is that two nodes are similar if their neighbours are similar.

Here, we classify 20 similarity based indices into three categories: the first 10 being Local indices, the next 7 being Global indices and the last 3 being Quasi-Local indices.

a) Local similarity indices:

These indices consider only information related to the immediate neighbourhood of the nodes. The different local indices are

i) Common Neighbours (CN):

It is of the notion that two nodes are likely to have a link if they have more number of common neighbours. Given nodes x and y, let $\Gamma(x)$ and $\Gamma(y)$ represent neighbours of

x and y respectively. Then,

$$s_{xy}^{CN} = |\Gamma(x) \cap \Gamma(y)|.$$

Where | Q | is the cardinality of the set Q.

ii) Salton Index:

Salton Index [6] is defines as,

$$s_{xy}^{Salton} = \frac{|\Gamma(x) \cap \Gamma(y)|}{\sqrt{k_x \times k_y}},$$

Where k_x is the degree of x. It is also referred to as cosine similarity index.

iii) Jaccard's Index:

Jaccard's index [37] was proposed by Jaccard and is defined as,

$$s_{xy}^{Jaccard} = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}.$$

It is similar to definition of probability.

iv) Sorensen Index:

The Sorensen index [38] is mainly used in ecological community data. It gives more importance to common things between the nodes. It is defined as,

$$s_{xy}^{Sørensen} = \frac{2|\Gamma(x) \cap \Gamma(y)|}{k_x + k_y}.$$

v) Hub Promoted Index (HPI):

The Hub Promoted Index [39] finds great application in metabolic networks. This index is of the notion that links that are adjacent to hubs have higher chances of getting linked to it. It is defines as,

$$s_{xy}^{HPI} = \frac{|\Gamma(x) \cap \Gamma(y)|}{\min\{k_x, k_y\}}.$$

vi) Hub Depressed Index (HDI):

The Hub Depressed Index has an opposite effect on hubs and is defined as,

$$s_{xy}^{HDI} = \frac{|\Gamma(x) \cap \Gamma(y)|}{\max\{k_x, k_y\}}.$$

vii) Leicht-Holme-Newman Index (LHN1):

The Leicht-Home-Newman Index [32] compares the number of common neighbours between two nodes with expected number of common neighbours unlike Jaccard's Index which compares with the maximum possible common neighbours.

$$s_{xy}^{LHN1} = \frac{|\Gamma(x) \cap \Gamma(y)|}{k_x \times k_y},$$

Here, the denominator is proportional to the expected number of common neighbours.

viii) Preferential Attachment Index (PA) :

The Preferential Attachment method is used to generate evolving scale free networks [41] where probability that a new link is connected to a node x is proportional to its degree k_x. Extending this concept we can find the probability that a new link connects two nodes x and y. Thus, it is defined as,

$$s_{xy}^{PA} = k_x \times k_y,$$

The Preferential Attachment method has the least computational complexity compared to other indices as it does not require much information on the neighbourhood.

ix) Adamic Adar (AA) :

The Adamic Adar Index [46] was proposed by L.A. Adamic and E.Adar. It assigns more weight to less connected members. It is defined as,

$$s_{xy}^{AA} = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log k_z}.$$

x) Resource Allocation Index (RA) :

The Resource Allocation Index [47] was primarily developed for resource allocation dynamics in complex networks. Let x and y be any two nodes and assuming there is no link between them, x can send resource to y through their common neighbours. If we assume that the resource is divided equally among all neighbours, then the amount of resource received by y from x is,

$$s_{xy}^{RA} = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{k_z}.$$

Comparison of metrics between local indices:

Indices	PPI	NS	Grid	PB	INT	USAir
CN	0.889	**0.933**	**0.590**	0.925	**0.559**	0.937
Salton	0.869	0.911	0.585	0.874	0.552	0.898
Jaccard	0.888	**0.933**	**0.590**	0.882	**0.559**	0.901
Sørensen	0.888	**0.933**	**0.590**	0.881	**0.559**	0.902
HPI	0.868	0.911	0.585	0.852	0.552	0.857
HDI	0.888	**0.933**	**0.590**	0.877	**0.559**	0.895
LHN1	0.866	0.911	0.585	0.772	0.552	0.758
PA	0.828	0.623	0.446	0.907	0.464	0.886
AA	0.888	0.932	**0.590**	0.922	**0.559**	0.925
RA	**0.890**	**0.933**	**0.590**	**0.931**	**0.559**	**0.955**

Liben-Nowell and Zhou systematically compared all the local indices on different complex networks [47] including protein-protein interaction network (PPI), electronic grid (GRID), Internet (INT), US airport network (USAir), co authorship network on network science (NS) and US political blogs (PB). According to the experimental results shown in the table below RA performs best while CN and AA have second best overall performance among all local indices.

The PA method has the worst performance out of all indices. However we are still interested in PA because it requires the least information and has least computational complexity. The LHN1 index has the second worst performance. However compared to other indices it is good at uncovering missing links connecting small degree nodes.

b) Global Similarity Indices:

These indices take into account the entire network unlike local indices which take into account only the immediate neighbourhood.

i) Katz Index:

The Katz Index [63] is based on ensemble of all paths, which directly sums up all paths and is exponentially damped by length to give more weight to shorter paths. The following mathematical expression defines Katz Index,

$$s_{xy}^{Katz} = \sum_{l=1}^{\infty} \beta^l \cdot |paths_{xy}^{<l>}| = \beta A_{xy} + \beta^2 (A^2)_{xy} + \beta^3 (A^3)_{xy} + \cdots,$$

where β is a free parameter used to control path weight. If the value of β is very small then Katz index is similar to CN as long paths are not given much importance.

ii) Leicht-Holme-Newman Index (LHN2):

The Leicht-Holme-Newman Index [36] is a variant of Katz index. It is of the notion that if the neighbours are similar then the nodes are similar.

$$S = \phi AS + \psi I = \psi (I - \phi A)^{-1} = \psi (I + \phi A + \phi^2 A^2 + \cdots),$$

Where Ø and Ψ are parameters used to control similarity. If Ψ=1, then LHN2 is similar to Katz index.

iii) Average Commute Time (ACT):

Let x and y be two nodes and m(x, y) denote the average number of steps required to move from node x to node y. Then, the Average Commute Time is given by,

$$n(x, y) = m(x, y) + m(y, x),$$

Using the Laplacian matrix, it can be derived as,

$$n(x, y) = M(l_{xx}^{+} + l_{yy}^{+} - 2l_{xy}^{+}),$$

Now, assuming nodes are similar if they have small average commute time we get,

$$s_{xy}^{ACT} = \frac{1}{l_{xx}^{+} + l_{yy}^{+} - 2l_{xy}^{+}}.$$

iv) Cosine based on L+:

The cosine similarity is defined as cosine of the node vectors, namely [65]

$$s_{xy}^{cos^+} = cos(x,y)^{+} = \frac{v_x^T v_y}{|v_x| \cdot |v_y|} = \frac{l_{xy}^{+}}{\sqrt{l_{xx}^{+} \cdot l_{yy}^{+}}}.$$

v) Random Walk with Restart (RWR): ***

The Random Walk with Restart Index is a direct application of PageRank algorithm [66]. Consider a random walker strictly at x, who moves iteratively to neighbour with probability c and returns with probability (1-c). Then,

$$\vec{q_x} = cP^T\vec{q_x} + (1-c)\vec{e_x},$$

where q_{xy} is the probability that the walker locates at node y, P is the transition matrix. Thus RWR index is defines as,

$$s_{xy}^{RWR} = q_{xy} + q_{yx}.$$

vi) SimRank:

The SimRank index [69] is similar to LHN2. It is also of the notion that two nodes are similar if they are connected to similar nodes. It is defined as,

$$s_{xy}^{SimRank} = C \cdot \frac{\sum_{z \in \Gamma(x)} \sum_{z' \in \Gamma(y)} s_{zz'}^{SimRank}}{k_x \cdot k_y}$$

It can also be interpreted as a random walk process where in it measures how soon two random walkers starting at x and y respectively meet at a certain node.

vii) Matrix Forest Index:

The Matrix Forest Index defines the ratio of number of spanning rooted forests such that x and y belong to the same tree rooted at x to all spanning rooted forests of the network. It is defined as,

$$S = (I + L)^{-1},$$

Where L is the Laplacian matrix.

Comparison between local and global indices:

Local indices take into account only the immediate neighbourhood while global indices ask for whole topological information and provide much accurate predictions. However, global indices have their own share of drawbacks. It takes more time for computation purpose. Sometimes, global topological information may not be available. It is not suitable if the information is decentralised.

c) Quasi-Local Indices:

The Quasi-Local Indices take into account information that is in between that of Local indices and global indices.

i) Local Path Index (LPI):

The Local Path Index [47, 72] provides a better trade-off of accuracy and computational complexity by taking into account a wider horizon than CN. It is defined as,

$$S^{LP} = A^2 + \epsilon A^3,$$

Where ε is a parameter. This is equal to CN with ε = 0. This can be extended to include higher order paths, as

$$S^{LP(n)} = A^2 + \epsilon A^3 + \epsilon^2 A^4 + \cdots + \epsilon^{n-2} A^n,$$

Where n > 2. As the value of n increases, more information is required and the computational complexity also increases. When $n \to \infty$, it becomes similar to Katz index which takes into account all the paths. However, the computational complexity involved is much greater than Katz index.

It is seen that LPI performs better than some of the local indices like CN, RA, AA as the information they use are less distinguishable and the chances of multiple pair of nodes getting same score is more. When compared with global indices like Katz, LHN2, it is seen that Katz is best for AUC value while LPI is best for precision value. However, for a network with small average shortest distances LPI provides better predictions for both AUC and Precision.

ii) Local Random Walk (LRW): ***

To measure the similarity between two nodes, say x and y LWR [73] places a random walker initially at node x and the density vector $\pi_x(0) = e_x$ and evolves as $\pi_x(t+1) = P^T \pi_x(t)$ for $t \geq 0$. LRW is defined as,

$$s_{xy}^{LRW}(t) = q_x \pi_{xy}(t) + q_y \pi_{yx}(t).$$

Where q is the initial configuration function and can be determined as $q_x = k_x / M$.

iii) Superposed Random Walk (SRW):

Superposed Random Walk [73] is similar to LRW where the random walker is repeatedly released at the starting point, resulting in a stronger similarity between the target nodes and the nodes nearby. It is defined as,

$$s_{xy}^{SRW}(t) = \sum_{\tau=1}^{t} s_{xy}^{LRW}(\tau) = \sum_{\tau=1}^{t} [q_x \pi_{xy}(\tau) + q_y \pi_{yx}(\tau)],$$

Where t denotes time steps.

Maximum Likelihood Methods:

The maximum likelihood methods assume some organising principles of the network structure with detailed rules and parameter obtained by maximizing the likelihood of observed structure. Then, the likelihood of non-observed links is estimated using the rules and parameters.

Maximum likelihood methods from the point of view of practical applications are time consuming and a well designed algorithm can handle only a limited number of nodes in the network and fails for larger networks. In addition, they are not the most

accurate ones. However, it provides good insight into the network organisation.

i) Hierarchical Structure Model:

It is seen that many of the networks are hierarchically organised. These networks are divided into groups and are further subdivided into groups of groups and so on. Focusing on these groups can provide better ways of predicting missing links.

The steps involved in predicting links are

1) Sample a large number of dendrograms with probability proportional to their likelihood.

$$p_r^* = \frac{E_r}{L_r R_r}$$

$$\mathcal{L}(D, \{p_r\}) = \prod_r p_r^{E_r} (1 - p_r)^{L_r R_r - E_r}.$$

2) For each pair of nodes, find mean connecting probability (P_{ij}) using average of probability over all sampled dendrograms.

3) Sort the node pairs in decreasing order of their P_{ij}.

ii) Stochastic Block Model:

In Stochastic Block Models the nodes are divided into groups and the probability that two nodes are connected depends on the groups to which they belong. It captures community structure, role-to-role interactions and other factors where group membership plays an important role on how nodes interact with each other.

Given a partition μ of the network where each node belongs to one group and the connecting probability for two nodes in groups α and β is Qαβ then, the likelihood of observed structure is,

$$\mathcal{L}(A|\mathcal{M}) = \prod_{\alpha \leq \beta} Q_{\alpha\beta}^{l_{\alpha\beta}} (1 - Q_{\alpha\beta})^{r_{\alpha\beta} - l_{\alpha\beta}},$$

where lαβ is the number of edges between groups α and β and rαβ is the number of pairs of nodes such that one is in α and the other is in β. Qαβ can be computed as,

$$Q_{\alpha\beta}^* = \frac{l_{\alpha\beta}}{r_{\alpha\beta}}.$$

Once the likelihood has been computed, reliability is calculated for each link using Bayes theorem as,

$$R_{xy} = \mathcal{L}(A_{xy} = 1|A) = \frac{\int_\Omega \mathcal{L}(A_{xy} = 1|\mathcal{M})\mathcal{L}(A|\mathcal{M})p(\mathcal{M})\mathrm{d}\mathcal{M}}{\int_\Omega \mathcal{L}(A|\mathcal{M}')p(\mathcal{M}')\mathrm{d}\mathcal{M}'},$$

Where $p(\mu)$ is a constant. Reliability describes the likelihood of existence of a link using the observed structure which can be used to predict missing or future links.

Probabilistic Models:

Probabilistic models aim at extracting the underlying structure from the observed network and then predict links using the learned models. Given a network *G*, probabilistic models build a target function to establish a model with a set of parameters θ that best fits the network *G*. Then, the probability of occurrence of a non-existent link is calculated using conditional probability property.

i) Probabilistic Relational Model (PRM):

PRM's are usually applied over relational databases where properties of objects depend not only on other properties of the object but also on properties of other objects. PRM's use three graphs unlike traditional graphs which use only one and they are: the data graph G_d, the model graph Gm and the inference graph G_i.

The data graph $G_d\ (V_d, E_d)$ represents the input network, where nodes are objects and the edges represent the relationship between the objects. Each node $v_i \in V_d$ and edge $e_i \in E_d$ has a type t, i.e., $T(v_i) = tv_i$ and $T(e_i) = te_i$. Each item (node or edge) $t \in T$ has a set of attributes X_t associated with it. PRMs basically represent the joint probability distribution over the values of all attributes of the graph.

The model graph $G_m\ (V_m, E_m)$ represents dependencies among attributes at item level. The attributes of an item can depend on other attributes of the same item or attributes of different items. Now, each node in Vm has a set of attributes. The attributes with same type in Gd are tied together. Thus, Gd is now split into examples of same types. Gm has two parts: the dependent structure among all the type attributes and the conditional probability distribution (CPD) associated with nodes in Gd.

The inference graph $G_i\ (V_i, E_i)$ represents probabilistic dependencies among several

variables in a single test set. Every item-attribute pair from Gd gets a CPD from Gm. The relations present in Gd dictates the CPD assigned by Gm and in turn dictates the structure of G_i.

Depending on the type of model graph being used it can be divided into three categories.

a) Relational Bayesian Networks (RBN):

The model graph used by RBNs are directed acyclic graphs with a set of CPDs P. P contains the CPD of each variable given its parents. Joint probability distribution of a variable x is calculated as,

$$p(\mathrm{x}) = \prod_{t \in T} \prod_{X_i^t \in \mathrm{X}^t} \prod_{v:T(v)=t} p(x_{v_i}^t | pa_{x_{v_i}^t}) \prod_{e:T(e)=t} p(x_{e_i}^t | pa_{x_{e_i}^t}).$$

Where p_{ax} is denotes the parent of x.

b) Relational Markov Networks (RMN):

The model graphs used by RMNs are undirected graphs with a set of potential functions. If C is the set of cliques and every clique $c \in C$ is associated with a set of variables then joint probability distribution of a variable x is calculated as,

$$p(\mathrm{x}) = \frac{1}{Z} \prod_{c \in C} \Phi_c(x_c),$$

Where Z is a normalizing constant.

c) Relational Dependencies Network (RDN):

The model graphs used by RDNs are bi-directed graphs with a set of CPDs that can be used to represent cyclic dependencies.

ii) Probabilistic Entity Relationship Models (PERM):

PERMs are directed acyclic probabilistic entity relationship models (DAPER) which uses directed arcs to describe the relationship between the attributes of objects. The DAPER model uses six classes and they are,
a) Entity classes – specifies the objects of real world.
b) Relationship classes – specifies the relationship between entities.
c) Attribute classes – specifies properties of entities or relationships.
d) Arc classes – represents dependencies between corresponding attributes.

e) Local Distribution classes – local distributions for attributes to corresponding attribute classes.
f) Constraint classes – specifies how to derive inference graph from DAPER model.

DAPER models are used when the relational structures are uncertain.