A Survey of Link Prediction in Complex Networks

VÍCTOR MARTÍNEZ, FERNANDO BERZAL, and JUAN-CARLOS CUBERO,
University of Granada

Networks have become increasingly important to model complex systems composed of interacting elements. Network data mining has a large number of applications in many disciplines including protein-protein interaction networks, social networks, transportation networks, and telecommunication networks. Different empirical studies have shown that it is possible to predict new relationships between elements attending to the topology of the network and the properties of its elements. The problem of predicting new relationships in networks is called link prediction. Link prediction aims to infer the behavior of the network link formation process by predicting missed or future relationships based on currently observed connections. It has become an attractive area of study since it allows us to predict how networks will evolve. In this survey, we will review the general-purpose techniques at the heart of the link prediction problem, which can be complemented by domain-specific heuristic methods in practice.

CCS Concepts: • **General and reference → Surveys and overviews**; • **Mathematics of computing → Graph algorithms**; • **Information systems → Data mining**; • **Theory of computation → Graph algorithms analysis;**

Additional Key Words and Phrases: Undirected networks, topological properties, similarity-based techniques, probabilistic techniques

ACM Reference Format:
Víctor Martínez, Fernando Berzal, and Juan-Carlos Cubero. 2016. A survey of link prediction in complex networks. ACM Comput. Surv. 49, 4, Article 69 (December 2016), 33 pages.
DOI: http://dx.doi.org/10.1145/3012704

1. INTRODUCTION

A link is a connection between two nodes in a network. This simple concept can be used to represent extremely complex systems where a large number of elements interact among them. The proliferation of data that can be represented as networks has created new opportunities but also new challenges in the field of data mining. A large number of problems related to network mining are currently being studied, including community detection [Fortunato 2010], structural network analysis [Kossinets and Watts 2006], and network visualization [Tamassia 2013]. One of the most interesting network-related problems is link prediction, which consists of inferring the existence of new relationships or still unknown interactions between pairs of entities based on their properties and the currently observed links [Liben-Nowell and Kleinberg 2007]. The approaches and techniques designed to solve this problem enable the extraction of

This work is partially supported by the Spanish Ministry of Economy and the European Regional Development Fund (FEDER), under grant TIN2012-36951, and partially by the Ministry of Education of Spain under the program "Ayudas para contratos predoctorales para la formación de doctores 2013" (grant BES-2013-064699).
Authors' addresses: V. Martínez, F. Berzal, and J.-C. Cubero, Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain; emails: victormg@acm.org, berzal@acm.org, JC.Cubero@decsai.ugr.es.

DOI: http://dx.doi.org/10.1145/3012704

implicit information present in the network and the identification of spurious links, as well as modeling and evaluating network evolution mechanisms.

Link prediction methods have been successfully applied to biological networks in order to predict previously unknown interactions between proteins [Martínez et al. 2014], significantly reducing the costs of empirical approaches [Schwikowski et al. 2000]. They have also been used to model highly dynamic systems, such as email or telephone call networks [O'Madadhain et al. 2005]. Link prediction techniques are widely present in our daily lives, suggesting people we may know but we are not still connected to in our social networks [Liben-Nowell and Kleinberg 2007] or products we could be interested in in electronic commerce [Huang et al. 2005].

Networks have been extensively studied since the proposition of the first basic models to identify the laws that drive network formation and lead to their structural features [Newman 2003]. Some techniques that could be considered as link prediction methods were then proposed. However, it was not until specific link-prediction-oriented seminal works [Liben-Nowell 2005], which performed a comprehensive analysis of the problem, that this field came under the spotlight due to its applicability and usefulness in a great variety of contexts.

Link prediction is grounded in the empirical evidence that two entities are more likely to interact if they are similar. Similarity in networks must be understood as an abstract concept and could vary between networks. Understanding the domain that the network represents is a crucial step to define the similarity between two nodes. In most domains, it has been observed that nodes tend to form highly connected communities [Palla et al. 2005]. This has led to the common definition of similarity as the amount of relevant direct or indirect paths between nodes.

One of the main difficulties in link prediction is achieving a good balance between the amount of information considered to perform the prediction and the algorithm complexity of the techniques needed to collect that information. Since actual networks are usually formed by hundreds of thousands or even millions of nodes, the techniques used to perform link prediction must be highly efficient. However, considering only local information could lead to poor predictions, especially in very sparse networks.

Different reviews about this topic have been previously published and have influenced this work [Liben-Nowell and Kleinberg 2007; Al Hasan and Zaki 2011; Lü and Zhou 2011; Wang et al. 2015]. However, new approaches have been developed since their publication and a new review of the state of the art is desirable. A large number of domain-specific link prediction techniques have been proposed. However, these techniques are left out of this review since most of them are tuned variations of the basic methods that will be described later. In this work, we focus on link prediction techniques in undirected networks using derived topological features. These techniques are more versatile than attribute-based methods since topology-based techniques are not domain specific.

We make several contributions in this survey. First, we perform a detailed and thorough study of the state of the art of link prediction approaches and methods using a unified notation. This study includes the computational complexity analysis of the most important techniques, which is not always provided by their original authors. Second, we propose a taxonomy to classify link prediction techniques attending to the methodology that they employ and the amount of information they use. Finally, we perform an empirical study of the techniques by applying the most important methods to a set of networks with different properties and evaluating their results.

2. THE LINK PREDICTION PROBLEM

The link prediction problem in undirected networks is defined as follows. Given a snapshot of an undirected network in time t where each node represents an entity

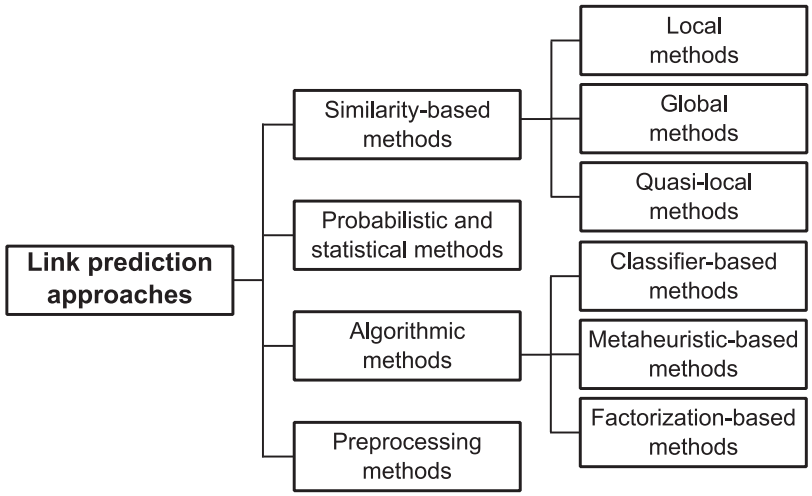

Fig. 1. Our proposed taxonomy for link prediction techniques.

or actor and each link represents an interaction or relationship between the pair of entities connected through the link, the link prediction problem can be formally defined as inferring the subset of missing links (existing but nonobserved links) in the current snapshot or that will be formed in time $t + \Delta$.

Most existing link prediction techniques consider it a ranking problem where pairs of unconnected nodes are given a score proportional to the likelihood of the existence of a link between them. A threshold is usually established: all pairs with a score above the threshold are considered as positive instances and all pairs below the threshold are viewed as negative instances. This threshold can be specified by the user, application dependent, or automatically determined. As far as we know, automatic threshold selection in link prediction remains an unexplored problem. The link prediction problem can be viewed as a binary classification problem for links in the network where two classes are considered: positive or existence of link and negative or absence of link.

A large number of link prediction techniques have been proposed in the specialized literature. These techniques differ in different aspects, including the evolution rules that they model, their computational complexity, or the type or amount of information they consider. We propose a custom extended version (see Figure 1) of the taxonomy presented by Lü and Zhou [2011]. These taxonomies classify the previously proposed methods attending to the approach they follow and the amount of information that they take into account. Each method is described in detail next.

2.1. Applications of Link Prediction

Link prediction techniques have found a large number of applications in very different fields. Any domain where entities interact in a structured way can potentially benefit from link prediction. Some interesting or widely used applications of link prediction are described later.

Link prediction techniques are used to improve similar users' selection in recommender systems that follow a collaborative approach, leading to better recommendation results [Schafer et al. 2007]. A similar application is related to social networks, which have become extremely popular in modern society. The users of these systems expect to have simple and effective mechanisms to find their acquaintances among the

massive amount of users registered. Most social networks are using link prediction techniques to automatically suggest acquaintances with a high degree of accuracy.

In the field of biology, link prediction techniques are being applied to find possible interactions between pairs of proteins in a protein-protein interaction network (PPI network) [Qi et al. 2006]. In vitro experiments to determine which proteins interact are expensive in money and time, so studied targets are carefully selected when there is prior evidence, which could be obtained computationally.

Another application is found in collaboration prediction in scientific coauthorship networks. Collaboration data is easily accessible, since some journal indexing sites make public their collections. Link prediction methods have become a tool to better understand how some research fields will evolve by predicting which authors or groups could potentially collaborate in the future [Pavlov and Ichise 2007].

Entity resolution, also known as record linkage or deduplication, consists of finding duplicated references or records in a dataset. Traditionally, entity resolution only relied on attribute similarity between entries. However, recently, some authors have shown that considering context information in network-structured domains using link prediction techniques to take into consideration the similarity between the instances can lead to improvements in entity resolution [Bhattacharya and Getoor 2007].

Social network analysis has been widely used to analyze the structure of criminal and terrorist networks in order to fight against organized crime [Krebs 2002]. For example, Xu and Chen [2008] have shown that the topology of some criminal networks does not change if a significant fraction of links are reinserted using link prediction techniques. These results suggest that link prediction can reveal actual links in criminal networks, allowing us to anticipate certain criminal actions.

Finally, networks can be used to analyze how tendencies spread across society. Network analysis can be used to improve marketing studies. Some authors have shown how link prediction can be used in viral marketing in order to achieve better marketing plans [Richardson and Domingos 2002].

2.2. Terminology and Notation

A graph or network G is an ordered pair $G = (V, E)$, where V is a set of optionally labeled vertices or nodes and E is a set of links between pairs of elements from the set V. A link between two nodes x and y is noted as $e_{x,y}$. The number of nodes in the network, also known as the size of the network, is denoted as $|V|$. The number of links is denoted as $|E|$. We can distinguish between directed links (noted as arcs), which connect a source node to a destination node, and undirected links (noted as edges), when there is no concept of source and destination. A directed graph is composed only of arcs. Similarly, an undirected graph is composed only of edges. Finally, a mixed graph can contain both types of links (arcs and edges).

The set of nodes connected through an edge to a node $x \in V$ is called the neighborhood of x and is denoted as Γ_x. In undirected graphs, the degree of a node x is defined as the number of edges connected to the node and will be denoted as $|\Gamma_x|$. In directed graphs, the degree of a node is the sum of the out-degree and the in-degree, which are the count of outgoing arcs and incoming arcs, respectively. The average degree of a network is denoted as $\langle\Gamma\rangle$ and is equal to the average degree of all its nodes.

Let a loop be an edge or arc connecting a node to itself. A simple graph is defined as a graph without loops and with no more than one edge or arc between each pair of vertices. The techniques reviewed in this survey assume there are no loops in the network.

A path is a sequence of links that connect a sequence of nodes in the graph. In directed networks, path steps are restricted to move from the source node to the destination node of the same arc. The path length is the number of links in the path. The shortest

path between two vertices is the path with the smaller length between those vertices. Multiple shortest paths for a pair of vertices can coexist. A graph is called connected if there is a path between each pair of nodes $x, y \in V$. If the graph is not connected, it is composed of components. A component is a connected subgraph. A connected graph has only one component. If one of the components has a significantly larger number of nodes compared to the other components, it is usually called the main or giant component.

3. SIMILARITY-BASED METHODS

Similarity-based methods assume that nodes tend to form links with other similar nodes. These methods stem from the hypothesis that two nodes are similar if they are connected to similar nodes or are near in the network according to a given distance function. These approaches define a function $s(x, y)$ that assigns a score known as similarity for every pair of nodes x and y. This measure is computed for each interesting pair of nodes, usually those with nonobserved links between them. Pairs of nodes are ranked in decreasing order based on their similarity scores, so links at the top of the ranking are supposed to be more likely to be present in the set of missing links.

The definition of similarity is not a trivial task, since it has a heuristic component. The similarity function can vary between networks even from the same domain. As a nonsurprising result, a large number of similarity-based methods with different definitions of similarity have been proposed. It has been empirically shown that the similarity between nodes can be defined in terms of network topological properties.

As an additional contribution of this survey, the algorithmic complexity is also computed for every similarity-based method using the big O notation. Three variables have been considered to measure problem size: v as the number of nodes, e as the number of edges, and k as the maximum degree of a node. Some simple optimizations will be taken into account in our algorithmic complexity analysis.

For example, the intersection between two sets of size n and m, respectively, can be computed with complexity $O(n + m)$ using hash tables. Insertion and lookup time complexities are $O(1)$ using hashing. A set is iterated to fill a hash table. The second set is iterated to check whether its members already are in the hash table. Collisions can be totally avoided since each node has a unique assigned number. The set union operation has the same time complexity.

Matrix operations are also considered. Matrix addition and subtraction are $O(n^2)$. However, in sparse networks, they can be computed with complexity $O(nt)$, where $t \ll n$. The inversion of a matrix of size $n \times n$ is of cubic time complexity $O(n^3)$ for the naive algorithm. Some algorithms have been proposed to improve this efficiency, but they are always worse than quadratic. The multiplication of two matrices of size $n \times m$ and $m \times p$, respectively, has a complexity $O(mnp)$ in dense matrices. However, in sparse matrices, this efficiency can be reduced to $O(mtp)$, where $t \ll n$. It should be noted that the adjacency matrices of real-world networks are typically sparse.

3.1. Local Approaches

Local similarity-based approaches use node neighborhood-related structural information to compute the similarity of each node with other nodes in the network. These approaches are faster than nonlocal techniques and highly parallelizable. In addition, they allow us to handle efficiently the link prediction problem in very dynamic and changing networks such as online social networks. Their main drawback is that using only local information restricts the set of nodes similarity can be computed for to distance-two nodes (neighbors of neighbors). This can be a big drawback as many links are formed at distances greater than two in many real-world networks, especially in non-small-world networks [Liben-Nowell and Kleinberg 2007]. However, these methods have shown a very competitive prediction accuracy against more complex techniques. It

should be noted that, since these methods are limited to two-hop nodes, their time complexity is $O(vk^2 f(k))$, where $f(k)$ is the complexity of computing the similarity between a pair of nodes and their spatial complexity is $O(vk^2)$.

3.1.1. Common Neighbors (CN). Common neighbors is the simplest local technique. The similarity between two nodes is defined as the number of shared neighbors between both nodes [Liben-Nowell and Kleinberg 2007]. It makes sense to assume that, if two individuals share many acquaintances, they are more likely to meet than two individuals without common contacts. Different studies have confirmed this hypothesis by observing a correlation between the number of shared neighbors between pairs of nodes and the probability of being linked [Newman 2001]. This method defines the similarity function as

$$s(x, y) = |\Gamma_x \cap \Gamma_y|.$$

Despite its simplicity, this measure performs surprisingly well on most real-world networks and beats very complex approaches. This method is the basis for other approaches presented later. Using this method to compute similarity for all possible pairs results in a local link prediction technique with $O(vk^2(k + k))) = O(vk^3)$ time complexity.

3.1.2. The Adamic-Adar Index (AA). This similarity measure, initially proposed by Lada Adamic and Eytan Adar, was intended to measure the similarity between two entities based on their shared features [Adamic and Adar 2003]. However, each feature weight is logarithmically penalized by its appearance frequency. If we take neighbors as features, it can be written as

$$s(x, y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{\log |\Gamma_z|}.$$

This equation is a variation of the common neighbors similarity function where each shared neighbor is penalized by its degree. This intuitively makes sense in a large number of real-world networks. For example, in social networks, the amount of resources or time that a node can spend on each of its neighbors decreases as its degree increases, also decreasing its influence on them. Its computational complexity is, again, $O(vk^3)$.

3.1.3. The Resource Allocation Index (RA). This index is motivated by the resource allocation process that takes place in complex networks [Zhou et al. 2009]. It models the transmission of units of resources between two unconnected nodes x and y through neighborhood nodes. Each neighborhood node gets a unit of resource from x and equally distributes it to its neighbors. The amount of resources obtained by node y can be considered as the similarity between both nodes. This similarity function is formulated as

$$s(x, y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{|\Gamma_z|}.$$

It should be noted that this measure is strongly related to the common neighbors and the Adamic-Adar index [Martínez et al. 2016]. The resource allocation index has been shown to be the local measure that achieves better results in a large number of networks. Some recent works have led to the conclusion that the performance of these metrics increases by making the penalization for high-degree nodes more pronounced in scale-free networks [Virinchi and Mitra 2013]. This method also has a $O(vk^3)$ time complexity.

3.1.4. Resource Allocation Based on Common Neighbor Interactions (RA-CNI). Resource allocation based on common neighbor interactions is motivated by the resource allocation

process where each node sends a unit of resource to its neighbors [Zhang et al. 2014]. However, this method also takes into consideration the return of resources in the opposite direction. It is defined as

$$s(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{|\Gamma_z|} + \sum_{e_{i,j} \in E, |\Gamma_i| < |\Gamma_j|, i \in \Gamma_x, j \in \Gamma_y} \left(\frac{1}{|\Gamma_i|} - \frac{1}{|\Gamma_j|} \right).$$

Experimental results obtained by its original authors show that this method achieves a better precision than the original local resource allocation score in many real-world networks. The computational complexity of this method is, however, $O(vk^4)$ in the worst case, worse than the aforementioned methods.

3.1.5. The Preferential Attachment Index (PA). This index is a direct result of the well-known Barabási-Albert complex network formation model [Barabási and Albert 1999; Mitzenmacher 2004]. Many real network node degrees follow a power law distribution, resulting in scale-free networks that could not be explained by previous network formation models. Albert-László Barabási and Réka Albert built a theoretical model based on the observation that the probability of link formation between two nodes increases as the degree of these nodes does. This formation model leads to the concept of "the rich get richer," which generates the power law degree distribution observed in scale-free networks. The similarity between two nodes, according to the Barabási-Albert model, can be estimated as

$$s(x,y) = |\Gamma_x||\Gamma_y|.$$

This measure can be also applied in nonlocal contexts, since it does not rely on shared neighbors. However, its prediction accuracy is usually poor when applied as a global measure. The computational complexity of this method is $O(vk^2)$, faster than the methods based on shared neighbors.

3.1.6. The Jaccard Index (JA). This widely used coefficient in information retrieval systems was proposed by Paul Jaccard (1868–1944) to compare the similarity and diversity of sample sets [Jaccard 1901]. It measures the ratio of shared neighbors in the complete set of neighbors for two nodes. This similarity function is defined as

$$s(x,y) = \frac{|\Gamma_x \cap \Gamma_y|}{|\Gamma_x \cup \Gamma_y|}.$$

It can be easily seen that this method is yet another variation of the common neighbors method where there is a penalization for each nonshared neighbor. The algorithmical time complexity of this method is $O(vk^2(2k + 2k)) = O(vk^3)$.

3.1.7. The Salton Index (SA). This index is also known as the cosine similarity [Salton and McGill 1983]. This measure is closely related to the Jaccard index, and some works have shown that, in most practical situations, the Salton index yields a value that is approximately twice the Jaccard index [Hamers et al. 1989]. This similarity function is defined as

$$s(x,y) = \frac{|\Gamma_x \cap \Gamma_y|}{\sqrt{|\Gamma_x||\Gamma_y|}}.$$

The computational time complexity of this method is, again, $O(vk^3)$.

3.1.8. The Sørensen Index (SO). This index was developed by the botanist Thorvald Sørensen in 1948 to compare the similarity between different ecological community data samples [Sørensen 1948]. Despite its similarity with the Jaccard index, it is less

sensitive to outliers [McCune et al. 2002]. Sørensen similarity is defined as

$$s(x, y) = \frac{2|\Gamma_x \cap \Gamma_y|}{|\Gamma_x| + |\Gamma_y|}.$$

The algorithmic time complexity of this method for all possible distance-two pairs is $O(vk^3)$.

3.1.9. The Hub Promoted Index (HPI). This index was proposed as a result of studying modularity in metabolic networks [Ravasz et al. 2002]. These networks show a hierarchical structure with small highly internally connected modules that are also highly isolated from each other. The main goal of this similarity measure is to avoid link formation between hub nodes and promote link formation between low-degree nodes and hubs. This index defines similarity as

$$s(x, y) = \frac{|\Gamma_x \cap \Gamma_y|}{\min(|\Gamma_x|, |\Gamma_y|)}.$$

The computation complexity of this method is, once more, $O(vk^3)$.

3.1.10. The Hub Depressed Index (HDI). This index is based on the hub promoted index but has an opposite goal [Ravasz et al. 2002]. The hub depressed index promotes link formation between hubs and between low-degree nodes, but not between hubs and low-degree nodes. This similarity function can be defined as

$$s(x, y) = \frac{|\Gamma_x \cap \Gamma_y|}{\max(|\Gamma_x|, |\Gamma_y|)}.$$

This method has the same computational complexity as the hub promoted index, which is $O(vk^3)$.

3.1.11. The Local Leicht-Holme-Newman Index (LLHN). This index is defined as the ratio of actual paths of length two between two nodes and a value proportional to the expected number of paths of length two between them [Leicht et al. 2006]. Its own authors proclaim that this index is a more sensitive measure of structural equivalence than others like the Salton index or the Jaccard index. The similarity function defined by this index can be computed as

$$s(x, y) = \frac{|\Gamma_x \cap \Gamma_y|}{|\Gamma_x||\Gamma_y|}.$$

As with almost all local methods, this one has a time complexity of $O(vk^3)$.

3.1.12. The Individual Attraction Index (IA). This index is similar to the resource allocation index but also takes into account how connected the shared neighbors are [Dong et al. 2011]. It makes sense to assume that two nodes with the same number of shared neighbors are more likely to be connected if their neighborhoods are also highly connected among them. These links are called inner links. This similarity is computed as

$$s(x, y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{|e_{z,\Gamma_x \cap \Gamma_y}| + 2}{|\Gamma_z|},$$

where $|e_{z,\Gamma_x \cap \Gamma_y}|$ is the number of links between node z and nodes from the set $\Gamma_x \cap \Gamma_y$. Dong et al. [2011] also proposed an alternative version with a lower computational complexity, which they called the simple individual attraction index:

$$s(x, y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{|e_{\Gamma_x \cap \Gamma_y}| + 2}{|\Gamma_z||\Gamma_x \cap \Gamma_y|},$$

where $|e_{\Gamma_x \cap \Gamma_y}|$ is the number of edges in the network that connect common neighbors of nodes x and y. The computational complexity of the more complex version, which we will call IA1, is $O(vk^4)$, whereas the computation complexity of the more efficient version, which we will call IA2, is $O(vk^3)$.

3.1.13. Mutual Information (MI). This method treats the problem from the perspective of information theory by computing the conditional self-information of the existence of a link given the set of common neighbors [Tan et al. 2014]. The similarity among two nodes is computed as

$$s(x, y) = -I(e_{x,y}|\Gamma_x \cap \Gamma_y) = -I(e_{x,y}) + \sum_{z \in \Gamma_x \cap \Gamma_y} I(e_{x,y}; z),$$

where $I(e_{x,y})$ is the self-information of x and y being connected, which is computed as

$$I(e_{x,y}) = -\log_2 \left(1 - \prod_{i=1}^{|\Gamma_y|} \frac{|E| - |\Gamma_x| - i + 1}{|E| - i + 1}\right),$$

and $I(e_{x,y}; z)$ is the mutual information of the existence of a link between x and y and the set of shared neighbors of these nodes, which is computed as

$$I(e_{x,y}; z) = \frac{1}{|\Gamma_z|(|\Gamma_z| - 1)} \sum_{u,v \in \Gamma_z : u \neq v} (I(e_{u,v}) - I(e_{u,v}|z)).$$

Finally, the conditional self-information of x and y being connected is computed as

$$I(e_{x,y}|z) = -\log_2 \frac{|\{e_{x,y} : x, y \in \Gamma_z, e_{x,y} \in E\}|}{\frac{1}{2}|\Gamma_z|(|\Gamma_z| - 1)}.$$

The complexity of evaluating a single link is $O(k^4)$, as shown by its authors. Therefore, applying it to nodes at distance two leads to a computational complexity $O(nk^6)$ when used as a local link prediction technique.

3.1.14. Local Naïve Bayes (LNB). This method assumes that each shared neighbor has a different role or degree of influence, which can be estimated using probability theory [Liu et al. 2011]. This method estimates the similarity of two nodes as

$$s(x, y) = \sum_{z \in \Gamma_x \cap \Gamma_y} f(z) \log(oR_z),$$

where o is a constant for the network, which is computed as

$$o = \frac{p_{unconnected}}{p_{connected}} = \frac{\frac{1}{2}|V|(|V| - 1)}{|E|} - 1.$$

R_z is the role or influence of the node, which is computed as

$$R_z = \frac{2|\{e_{x,y} : x, y \in \Gamma_z, e_{x,y} \in E\}| + 1}{2|\{e_{x,y} : x, y \in \Gamma_z, e_{x,y} \notin E\}| + 1}$$

and $f(z)$ is a function that measures the influence of the node. The authors suggest $f(z) = 1$ from common neighbors, $f(z) = \frac{1}{\log |\Gamma_z|}$ from the Adamic-Adar index, or $f(z) = \frac{1}{|\Gamma_z|}$ from the resource allocation method. The computational complexity of this method is, therefore, $O(vO(f(z)) + vk^3)$.

3.1.15. CAR-Based Indices (CAR). CAR-based methods are proposed under the hypothesis that two nodes are more likely to be linked if their common neighbors are members of a strongly inner-linked cohort, named a local community (LC) [Cannistraci et al. 2013]. This assumption allows us to give more importance to nodes interlinked with other neighbors. A CAR-based version of common neighbors is defined as

$$s(x, y) = \sum_{z \in \Gamma_x \cap \Gamma_y} 1 + \frac{|\Gamma_x \cap \Gamma_y \cap \Gamma_z|}{2}.$$

In a similar way, a CAR-based variation of the resource allocation can be computed as

$$s(x, y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{|\Gamma_x \cap \Gamma_y \cap \Gamma_z|}{|\Gamma_z|}.$$

The computational complexity of this approach relies on the underlying technique. Both examples shown previously have a computational complexity of $O(vk^4)$.

3.1.16. Functional Similarity Weight (FSW). This method is derived from the Sørensen index considering that the probability of a node x interacting with a node y is independent of the probability of the node y interacting with the node x in directed networks [Chua et al. 2006]. However, this score can be also applied to undirected networks as

$$s(x, y) = \left(\frac{2|\Gamma_x \cap \Gamma_y|}{|\Gamma_x - \Gamma_y| + 2|\Gamma_x \cap \Gamma_y| + \lambda} \right)^2,$$

where λ is computed as $\lambda = \max(0, \langle \Gamma \rangle - (|\Gamma_x - \Gamma_y| + |\Gamma_x \cap \Gamma_y|))$. This parameter is included to penalize the similarity between nodes when any of them has a small degree. The computational complexity of this method is $O(vk^3)$.

3.1.17. Local Interacting Score (LIT). This score is an iterative variation of the functional similarity weight [Liu et al. 2008]. Initially, weights are assigned as $s^{x,y}(0) = 1$ for connected pairs of nodes and $s^{x,y}(0) = 0$ for the rest of pairs. Then, weights are iteratively updated as

$$s^{x,y}(t) = \frac{\sum_{u \in \Gamma_x \cap \Gamma_y} s^{z,x}(t-1) + \sum_{v \in \Gamma_x \cap \Gamma_y} s^{z,y}(t-1)}{\sum_{u \in \Gamma_x} s^{z,x}(t-1) + \sum_{v \in \Gamma_y} s^{z,y}(t-1) + \lambda(x) + \lambda(y)}.$$

where $\lambda(x)$ is computed as $\lambda(x) = \max(0, \sum_{u \in V} \sum_{v \in \Gamma_u} s^{u,x}(t)/|V| - \sum_{z \in \Gamma_x \cap \Gamma_y} s^{z,x}(t-1))$. $\lambda(x)$ plays the role of the λ penalization in functional similarity weight. The computational complexity of each iteration is $O(vk^3)$. When the process is limited to l iterations, the final computational complexity is $O(lvk^3)$.

3.2. Global Approaches

Global similarity-based indices use the whole network topological information to score each link. These methods are not limited to measuring similarity between distance-two nodes. However, their computational complexity can make them unfeasible for large networks and their parallelization can be very complex, especially in distributed environments where the complete topology of the network may not be known by every computational agent. Although they exhibit very diverse time complexities, their spatial complexity is $O(v^2)$, since they have to store a score for every pair of nodes.

3.2.1. Negated Shortest Path (NSP). Negated shortest path [Liben-Nowell 2005] is a basic graph similarity measure that requires one to compute the shortest path between a pair of nodes. Shortest paths can be efficiently computed with Dijkstra's algorithm,

which has a $O(e \log v)$ complexity using an adjacency list representation of the network and a heap data structure for its priority queue. Given the shortest path between a pair of nodes x and y, their similarity can be computed as

$$s(x, y) = -|shortest\ path_{x,y}|.$$

Since shortest paths must be computed for each node in the network, the overall time complexity of this method is $O(ev \log v)$. Negated path prediction accuracy is poor even when compared to most local methods. Other methods described later, based on multiple paths, obtain significantly better results. This fact illustrates the importance of considering indirect paths in link prediction techniques.

3.2.2. The Katz Index (KI). This index sums the influence of all possible paths between two pairs of nodes, incrementally penalizing paths by their length [Katz 1953]. This index is defined as

$$s(x, y) = \sum_{l=1}^{\infty} \beta^l |paths^l_{x,y}| = \sum_{l=1}^{\infty} \beta^l (A^l)_{x,y},$$

where $paths^l_{x,y}$ is the set of paths of length l between nodes x and y, and A is the adjacency matrix of the network. It should be noted that the lth power of the matrix has each of its entries equal to the count of paths of length l between the corresponding pair of nodes. The parameter β is a damping factor where $0 < \beta < 1$. Giving a larger value to this parameter increases the influence of longer paths. If 1 is added to each element of the diagonal of the resulting similarity matrix S, this expression can be written in matrix terms as $S = \beta AS + I$. The similarity between all pairs of nodes can be directly computed using the closed form by rearranging for S in the previous expression and subtracting the previously added 1s to the elements in the diagonal:

$$S = (I - \beta A)^{-1} - I,$$

where I is the identity matrix. The similarity for each pair of nodes x and y is $s(x, y) = S_{x,y}$, where $S_{x,y}$ is the (x, y)-element of the matrix S. The Katz index has a great predictive power but the high algorithmic complexity required to compute the inverse of a matrix limits its applicability to small networks. The time complexity of this method is $O(vk + v^3 + v)$, where the $O(vk)$ term is due to matrix subtraction and scalar multiplication, $O(v^3)$ is due to matrix inversion, and $O(v)$ comes from the subtraction of the diagonal elements in the identity matrix. Therefore, the time complexity of this method is $O(v^3)$.

3.2.3. The Global Leicht-Holme-Newman Index (GLHN). The global version of the Leicht-Holme-Newman index is based on the same fundamentals as the Katz index [Leicht et al. 2006]. This index assigns a similarity proportional to the number of paths between nodes. Similarity between all pairs of nodes is defined as

$$S = I + \sum_{l=1}^{\infty} \phi^l A^l,$$

where the identity matrix term indicates maximal self-similarity. Parameter ϕ is similar to the damping factor used in the Katz index. The number of actual paths of length l between each pair of nodes can be replaced by their expected value, which is computed as

$$Expected\left((A^l)_{x,y}\right) = \frac{|\Gamma_x||\Gamma_y|}{2|E|}\lambda_1^{l-1},$$

where λ_1 is the largest or dominant eigenvalue of the network adjacency matrix A. Replacing A^l by $Expected(A^l)$, this similarity can be finally expressed as

$$S = 2|E|\lambda_1 D^{-1}\left(I - \frac{\beta}{\lambda_1}A\right)^{-1} D^{-1},$$

where D is a diagonal matrix with each element set to $D_{i,i} = |\Gamma_i|$ and β is a free parameter related to ϕ. If the constant factor is removed, this expression can be iteratively computed avoiding the matrix inversions as

$$S(t) = \frac{\beta}{\lambda_1}AS(t-1) + I$$

starting with $S(0)$ having all its elements set to zero. This iterative process is performed until convergence. Iterative methods require a threshold parameter ϵ with a value close to zero. When the absolute difference between two iterations is below this threshold, the process is considered to have converged.

In order to compute the required dominant eigenvalue, different approaches can be followed. One of the most efficient is power iteration, or the Von Mises iteration method. This technique performs a series of iterative steps that converge to the largest eigenvector. It can be expressed as

$$b(t) = \frac{Ab(t-1)}{\|Ab(t-1)\|},$$

where A is the adjacency matrix of the network and b is a vector of length $|V|$, which can be initialized with random values. This process is repeated until convergence. Finally, it has been shown that the dominant eigenvalue is equal to the norm of this vector, so $\lambda_1 = \|b(c)\|$, where c is the step of convergence.

The similarity between two nodes is defined as $s(x, y) = S(c)_{x,y}$, where c is the time step when convergence is reached. The computational complexity of this method is analyzed as follows, considering c as the number of iterations required. The largest eigenvalue can be computed in $O(cvk)$ time using the power iteration method. The complexity of the iterative process to obtain $S(t)$ is $O(cv^2k)$. In summary, the complexity of the Global Leicht-Holme-Newman method is $O(cv^2k + cvk) = O(cv^2k)$.

3.2.4. Random Walks (RW). Given a graph and a starting node, let us suppose that we randomly select a neighbor of this node and move to it; then, we repeat this process for each reached node. This Markov chain of randomly selected nodes is known as a random walk on the graph [Liu and Lü 2010]. Random walks were introduced by the mathematician Karl Pearson and have been applied to describe lots of stochastic processes in many fields such as economics, physics, or biology [Pearson 1905]. If we define $\overrightarrow{p^x}$ as the probability vector of reaching any node starting a random walk from node x, the probability of reaching each node can be iteratively approximated by

$$\overrightarrow{p^x}(t) = M^T\overrightarrow{p^x}(t-1),$$

where M is the transition probability matrix defined by the adjacency matrix A normalized by rows, with $M_{i,j} = A_{i,j}/\sum_k A_{i,k}$, and $\overrightarrow{p^x}(0)$ has all its elements set to 0 except $\overrightarrow{p^x_x}(0)$, which is set to 1. The transition probability matrix must satisfy some properties to ensure convergence. These constraints are satisfied for undirected simple networks. The previous equation is iteratively applied until a stop condition is met such as $\sum_{i\in V}(\overrightarrow{p^x_i}(t) - \overrightarrow{p^x_i}(t-1))^2 < \epsilon$, where ϵ is a threshold value close to zero. Finally, the similarity for each pair of nodes x and y is defined as $s(x, y) = \overrightarrow{p^x_y}$. Assuming that

the network is sparse and c is the number of iterations until convergence, the time complexity of this method is $O(cv^2k)$, since a sparse multiplication between the adjacency matrix and the probability vector ($O(vk)$) is repeated for each node in each iteration. If we also take into account the matrix normalization, the final time complexity is $O(cv^2k + vk) = O(cv^2k)$.

3.2.5. Random Walks with Restart (RWR). Let us consider a model based on the definition of random walk where we pick a node and move following a random walk with probability α or we return to the starting node with probability $(1 - \alpha)$. This model is known as a random walk with restart [Tong et al. 2006]. It is almost equivalent to the rooted version of the popular Google's PageRank algorithm [Page et al. 1999]. This problem can be defined as an optimization problem:

$$\min_{\overrightarrow{p^x}} \alpha \sum_{i,j \in V} M^T_{i,j} \left(\overrightarrow{p^x_i} - \overrightarrow{p^x_j}\right)^2 + (1-\alpha) \sum_{i \in V} \left(\overrightarrow{p^x_i} - \overrightarrow{s^x_i}\right)^2,$$

where M is the transition probability matrix computed for random walks and $\overrightarrow{s^x}$ is the seed vector of length $|V|$ with all its elements set to 0 except for $\overrightarrow{s^x_x} = 1$. The closed-form solution of this equation is

$$\overrightarrow{p^x} = (1-\alpha)(I - \alpha M^T)^{-1} \overrightarrow{s^x}.$$

Although this computation may be unfeasible for large networks, it can be iteratively approximated by the following iterative equation:

$$\overrightarrow{p^x}(t) = \alpha M^T \overrightarrow{p^x}(t-1) + (1-\alpha) s^x,$$

where $\overrightarrow{p^x}(0)$ has all its elements initially set to zero. This expression is iteratively applied, as in the random walk method. Since this measure is not symmetric, the final similarity for each pair of nodes is defined as $s(x, y) = \overrightarrow{p^x_y} + \overrightarrow{p^y_x}$. Like the random walk method, if we assume that the network is sparse, its complexity is $O(vc(vk + k) + vk) \approx O(cv^2k)$, where the $O(vc(vk + k))$ term is the time complexity of random walk including vector addition and the $O(vk)$ term is the computational complexity of the normalization process.

3.2.6. Flow Propagation (FP). Despite the fact that the random walk with restart method converges to a vector of probabilities, its iterative definition corresponds to a propagation process. Alternative normalizations can be used for the adjacency matrix. For example, Vanunu and Sharan [2008] proposed to apply RWR replacing the normalized adjacency matrix with the normalized Laplacian matrix, which can be computed as

$$M = D^l A D^r,$$

where D^l and D^r are diagonal matrices whose elements are respectively defined as $D^l_{i,i} = 1/\sqrt{\sum_j A_{i,j}}$ and $D^r_{i,i} = 1/\sqrt{\sum_j A_{j,i}}$. The computational complexity of this method is the same as the random walk with restart method complexity, since the transition matrix can be computed by the multiplication of each adjacency matrix entry by a scalar value.

3.2.7. Maximal Entropy Random Walk (MERW). Nodes tend to be linked to central nodes in structured networks. The maximal entropy random walk method incorporates the centrality of nodes in order to model this behavior [Burda et al. 2009]. This method

aims to maximize the entropy rate of a walk μ that is defined as

$$\mu = \lim_{l \leftarrow \infty} \frac{-\sum_{path^l_{x,y} \in paths^l} p(path^l_{x,y}) \ln p(path^l_{x,y})}{l},$$

where $p(path^l_{x,y}) = M_{x,h} M_{h,i} \dots M_{i,j} M_{j,y}$. In order to maximize the entropy, each element of the transition matrix is computed as

$$M_{i,j} = \frac{A_{i,j}}{\lambda} \frac{\psi_j}{\psi_i},$$

where λ is the largest eigenvalue of the adjacency matrix and ψ is the normalized eigenvector with respect to λ satisfying $\sum_{x \in V} \psi_x^2 = 1$. Following this theory, Li et al. [2011] proposes entropy maximizations of other global techniques described in this survey. The computational complexity of the normalization process is $O(cvk + vk)$ since computing the largest eigenvalue and the associated vector is $O(cvk)$ using the power iteration method as we have described. The final computational complexity of this method is $O(cvk + 2vk + vc(vk + k)) = O(cv^2k)$.

3.2.8. SimRank (SR). SimRank is a method that computes how soon two random walkers starting from nodes x and y are expected to meet [Jeh and Widom 2002]. This method is recommended for directed or mixed networks. However, as the number of undirected edges increases, it becomes impractical. It is recursively defined as

$$s(x, y) = \beta \frac{\sum_{i \in \Gamma_x} \sum_{j \in \Gamma_y} s(i, j)}{|\Gamma_x||\Gamma_y|},$$

where $s(z, z) = 1$ and $0 < \beta < 1$ is a damping factor. The high algorithmic complexity of this method makes it necessary to apply optimization techniques for its computation. Its original authors suggest pruning recursive branches beyond a radius l. The authors of SimRank suggest a near-linear complexity with a large constant factor in directed networks. However, this recursive expansion process has a complexity $O(k^{2l})$. Since two nested summations must be performed, the complexity for each pair of nodes is $O(k^{2l+2})$. This score must be computed for each pair of nodes so the final algorithmic time complexity is $O(v^2k^{2l+2})$. It should be noted that, if pruning is applied with a radius lower than the network diameter, this method can be considered as a quasi-local method (quasi-local methods are described in Section 3.3).

3.2.9. Pseudoinverse of the Laplacian Matrix (PLM). The Laplacian matrix provides an alternative representation of a graph, and it is widely used in spectral graph theory [Spielman 2009]. It can be defined as $L = D - A$, where D is a diagonal matrix of size $|V|$ with each element $D_{i,i} = |\Gamma_i|$ and A is the adjacency matrix of the graph. The Moore-Penrose pseudoinverse of the Laplacian matrix is noted as L^+. This matrix is a kernel and can be considered as a similarity matrix [Fouss et al. 2007]. The Moore-Penrose pseudoinverse can be computed in many different ways. One of the most popular approaches, implemented in most important mathematical packages such as MATLAB, GNU Octave, or NumPy, is based on singular value decomposition (SVD). Detailed steps to compute the SVD are included in the description of the low-rank approximation preprocessing method (see Section 6). Given the Laplacian matrix decomposition $L = U \Sigma V^T$, the Moore-Penrose pseudoinverse is computed as $L^+ = V \Sigma^+ U^T$, where Σ^+ is the matrix obtained from replacing each nonzero element of Σ by its inverse. Similarity can be computed using an inner-product-based measure such as the cosine

distance:

$$s(x, y) = \frac{L^+_{x,y}}{\sqrt{L^+_{x,x} L^+_{y,y}}}.$$

The complexity of the Moore-Penrose pseudoinverse of the Laplacian matrix computation is dominated by the cost of computing the SVD, which is $O(v^3)$. Once this matrix has been computed, the complexity of computing the similarity for each pair of nodes is $O(v^2)$. The overall complexity of this global link prediction method is, therefore, $O(v^3 + v^2) = O(v^3)$.

3.2.10. Average Commute Time (ACT). The average commute time is defined as the average number of steps that a random walker starting from node x takes to reach a node y for the first time and go back to x [Liu and Lü 2010]. If the number of steps needed to reach node y starting from node x in a random walk is denoted by $m(x, y)$ (also known as hitting time), the ACT value $c(x, y)$ between both nodes can be defined as

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

The average commute time can also be computed in terms of the pseudoinverse of the Laplacian matrix L:

$$n(x, y) = |E|(L^+_{x,x} + L^+_{y,y} - 2L^+_{x,y}),$$

where $\sqrt{n(x, y)}$ defines a distance measure called the Euclidean commute time distance (ECTD) between nodes x and y [Fouss et al. 2007]. Finally, the similarity between two nodes can be computed as the inverse of the squared ECDT between both nodes, ignoring the constant $|E|$:

$$s(x, y) = \frac{1}{L^+_{x,x} + L^+_{y,y} - 2L^+_{x,y}}.$$

The complexity of this method is the same we obtained for the pseudoinverse of the Laplacian matrix method, which was $O(v^3 + v^2) = O(v^3)$.

3.2.11. Random Forest Kernel Index (RFK). In graph theory, a spanning tree of a graph G is defined as a connected undirected subgraph with no cycles that includes all the vertices and some or all the edges of G. The matrix-tree theorem [Chebotarev and Shamis 2006] states that the number of spanning trees in G is equal to any cofactor of an entry of its Laplacian representation. A cofactor is the determinant of the matrix obtained by removing the row and column of a given element. A rooted forest is defined as the union of disjoint rooted spanning trees. It can be proved that the cofactor of $(I + L)_{x,y}$ is equal to the number of spanning rooted forests in which x and y are contained in the same x-rooted spanning tree. The inverse of this number can be considered as a measure of accessibility between x and y. Therefore, a similarity measure can be defined as

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

Given this similarity matrix, the similarity between a pair of nodes is $s(x, y) = S_{x,y}$. The algorithmic time complexity of this method is $O(v^3 + vk + v) = O(v^3)$ if we take into account the Laplacian matrix computation, $O(vk)$; the addition of the diagonal elements of the identity matrix, $O(v)$; and the matrix inversion, $O(v^3)$.

3.2.12. The Blondel Index (BI). The Blondel index was initially proposed to measure similarity for a pair of vertices in different graphs [Blondel et al. 2004]. However, it

can be adapted to work in a single graph. It is iteratively computed as

$$S(t) = \frac{AS(t-1)A^T + A^T S(t-1)A}{\|AS(t-1)A^T + A^T S(t-1)A\|_F},$$

where $S(0) = I$ and $\|M\|_F$ is the Frobenius matrix norm. The Frobenius norm for a matrix is computed as

$$\|M_{m\times n}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}(M_{i,j})^2}.$$

This measure is iteratively computed as in random-walk-based methods. The final similarity between a pair of nodes is defined as $s(x, y) = S_{x,y}(c)$, where c is the odd time step when convergence is reached. It is important to remark that this method reaches convergence only in odd iterations. The difference between the similarity matrices obtained in odd iterations must be computed in order to test for convergence.

The complexity of this method is $O(cv^2k + cv^2) = O(cv^2k)$, which can be derived from sparse matrix multiplications, $O(v^2k)$, matrix addition, matrix division by a scalar, and the Frobenius norm, $O(v^2)$.

3.3. Quasi-Local Approaches

Quasi-local methods have recently emerged to strike a balance between local and global measures. Quasi-local approaches are almost as efficient to compute as local methods but also consider additional topological information, as global methods do. They do not take into account the similarity between any arbitrary pair of nodes in the network, but neither are they limited to neighbors of neighbors. Some quasi-local methods have access to the whole network, but their algorithmic time complexity is still below the time complexity of global methods. Their spatial complexity is $O(vk^{2+s})$, where s depends on specific parameters that set the number of iterations or the length of the paths considered.

3.3.1. The Local Path Index (LPI). This index is strongly based on the Katz index but it only considers a finite number of path lengths [Lü et al. 2009]. The similarity matrix can be computed as

$$S = \sum_{i=2}^{l} \beta^{i-2} A^i,$$

where $l > 2$ is the maximal path length and β is a damping factor. It should be noted that, when $l = 2$, it would be equivalent to the common neighbors method. Similarity for each pair of nodes x and y is defined as $s(x, y) = S_{x,y}$. This measure is typically used with $l = 3$ due to its algorithmic complexity. When the damping factor is set to a low value, this measure obtains very similar results to the Katz index but avoids the matrix inversion computation. If the power of each adjacency matrix from the previous summation term is reused, the complexity of this method $O(lv^2k + lv^2) = O(lv^2k)$.

3.3.2. Local Random Walks (LRW). Local random walks exploit the concept of random walks but limit the number of iterations to a fixed a priori small number l [Liu and Lü 2010]. This method does not focus on the stationary state when convergence is reached like other random-walk-based approaches. It is defined as

$$s^{x,y}(t) = \frac{|\Gamma_x|}{2|E|}\overrightarrow{p_y^x}(t) + \frac{|\Gamma_y|}{2|E|}\overrightarrow{p_x^y}(t),$$

where $\overrightarrow{p_y^x}(t)$ is the probability vector obtained by the random walk at iteration t. This method only requires us to compute a limited number of random walk steps for all the nodes in the network, so its computational complexity is $O(lv^2k)$, where usually $l < i$, with i being the number of steps the random walk method would need to converge.

3.3.3. Superposed Random Walks (SRW). Random-walk-based methods are too sensitive to the topology of the network in distant zones. The superposed random walk method, which is based on the local random walk method, has been proposed to counteract this issue by continuously releasing the walker at the starting node [Liu and Lü 2010]. This behavior can be obtained by superposing each walker contribution as

$$s^{x,y}(t) = \sum_{i=1}^{t} \left(\frac{|\Gamma_x|}{2|E|} \overrightarrow{p_y^x}(i) + \frac{|\Gamma_y|}{2|E|} \overrightarrow{p_x^y}(i) \right).$$

The final similarity is computed as $s(x, y) = s^{x,y}(l)$. The computational complexity of this method is $O(lv^2k + lvk) = O(lv^2k)$.

3.3.4. Third-Order Resource Allocation Based on Common Neighbor Interactions (ORA-CNI). This measure is an extension of resource allocation based on common neighbor interactions that also takes into consideration distance-three paths [Zhang et al. 2014]. It redefines resource allocation for nodes at distance three. It is computed as

$$s(x, y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{|\Gamma_z|} + \sum_{e_{i,j} \in E, |\Gamma_i| < |\Gamma_j|, i \in \Gamma_x, j \in \Gamma_y} \left(\frac{1}{|\Gamma_i|} - \frac{1}{|\Gamma_j|} \right) + \beta \sum_{[x,p,q,y] \in paths^3_{x,y}} \frac{1}{|\Gamma_p||\Gamma_q|},$$

where β is a damping factor to adjust the influence of the three-hop resource allocation term. This method starts from the resource allocation based on common neighbors' interactions complexity, which it must also compute for distance-three nodes. Adding the third term to reach three-hop nodes increases its computational complexity to $O(vk^3(k^2 + k^3)) = O(vk^6)$.

3.3.5. FriendLink (FL). FriendLink is a quasi-local measure based on the path count between nodes of interest, like the local path index [Papadimitriou et al. 2012]. However, this method uses a normalization and other path length penalization mechanisms. The similarity between two nodes x and y is computed as

$$s(x, y) = \sum_{i=2}^{l} \frac{1}{i-1} \frac{(A^i)_{x,y}}{\prod_{j=2}^{i}(|V| - j)},$$

where $\frac{1}{i-1}$ is an attenuation factor similar to the Katz or the local path index damping factors, and the count of paths of length i between x and y is normalized by the count of paths of length i between x and y that would exist in a complete version of the network. The algorithmic complexity of this method is $O(lv^2k)$.

3.3.6. PropFlow Predictor (PFP). PropFlow is a similarity-based method that computes the probability that a restricted random walk starting from x ends at y in l steps or fewer [Lichtenwalter et al. 2010]. Given a source node x and maximal path length l, this algorithm returns an array S_x where each element $S_{x,y}$ is the score assigned between the pair of nodes x and y. New links can be predicted, as in other similarity-based methods, by setting the similarity score to $s(x, y) = S_{x,y}$.

The pseudocode of the algorithm can be found in Algorithm 1. This method is similar to the random walk with restart or rooted PageRank algorithms, but it employs the localized method of propagation and, therefore, is more insensitive to noise far from

ALGORITHM 1: PropFlow Predictor (Unweighted Version).

Input: Network $G = (V, E)$, node x and max path length l.
Output: Score $S_{x,y}$ for all $n \leq l$-degree neighbors of y from x.

```
Found = {x};
NewSearch = {x};
S_{x,x} = 1;
for each z in V − {x} do
    S_{x,z} = 0;
end
for CurrentDegree from 0 to l do
    OldSearch = NewSearch;
    NewSearch = ∅;
    for each i in OldSearch do
        for each j in Γ_i do
            S_{x,j} ← S_{x,j} + S_{x,i} / |Γ_i|;
            if j is not in Found then
                Found = Found ∪ {j};
                NewSearch = NewSearch ∪ {j};
            end
        end
    end
end
```

the source node x. It is also more efficient to compute, since it employs a breadth-first search limited to l steps. The analysis of this method, given that the maximum size of set *OldSearch* is k^l and the process must be repeated $|V|$ times, results in a computational complexity of $O(vlk^l)$.

3.4. Summary

Our survey has shown that very different approaches can be used to perform link prediction. A large number of methods using diverse approaches have been described in this chapter. These methods have been catalogued using the taxonomy shown in Figure 1. This taxonomy has two levels, which allow us to categorize each method based on its main features.

Similarity-based methods are the most studied category in link prediction and they compose the core of this survey. These methods are usually applied in massive networks so their time complexity is a fundamental feature. A summary of the similarity-based techniques that have been studied in this survey and their time complexity is shown in Table I, which groups methods by their taxonomic classification.

As it has been seen in this chapter, most link prediction techniques are heuristics based on some coherent assumptions. The heuristic nature of the link prediction problem allows a rich variety of approaches that might work better or worse depending on the particular context, since network formation is a complex process and some fundamental factors can vary among networks. This implies that it is impossible to devise a method that works better than all other methods for all networks.

3.5. Experimentation

In practice, link prediction strongly relies on similarity-based techniques, since most of them are efficient enough to be applied to massive networks. Furthermore, similarity-based scores are usually used as features when more complex approaches are applied using other algorithmic techniques. We have performed an extensive comparison of

Table I. Computational Complexity and References for Similarity-Based Link Prediction Methods

Type	Name	Time Complexity	Reference
Local	CN	$O(vk^3)$	Liben-Nowell and Kleinberg [2007]
	AA	$O(vk^3)$	Adamic and Adar [2003]
	RA	$O(vk^3)$	Zhou et al. [2009]
	RA-CNI	$O(vk^4)$	Zhang et al. [2014]
	PA	$O(vk^2)$	Barabási and Albert [1999]
	JA	$O(vk^3)$	Jaccard [1901]
	SA	$O(vk^3)$	Salton and McGill [1983]
	SO	$O(vk^3)$	Sørensen [1948]
	HPI	$O(vk^3)$	Ravasz et al. [2002]
	HDI	$O(vk^3)$	Ravasz et al. [2002]
	LLHN	$O(vk^3)$	Leicht et al. [2006]
	IA1	$O(vk^4)$	Dong et al. [2011]
	IA2	$O(vk^3)$	Dong et al. [2011]
	MI	$O(nk^6)$	Tan et al. [2014]
	LNB	$O(O(f(z)) + vk^3)$	Liu et al. [2011]
	CAR	$O(vk^4)$	Cannistraci et al. [2013]
	FSW	$O(vk^3)$	Chua et al. [2006]
	LIT	$O(lvk^3)$	Liu et al. [2008]
Global	NSP	$O(ev \log v)$	Liben-Nowell [2005]
	KI	$O(v^3)$	Katz [1953]
	GLHN	$O(cv^2k)$	Leicht et al. [2006]
	RW	$O(cv^2k)$	Pearson [1905]
	RWR	$O(cv^2k)$	Tong et al. [2006]
	FP	$O(cv^2k)$	Vanunu and Sharan [2008]
	MERW	$O(cv^2k)$	Li et al. [2011]
	SR	$O(v^2k^{2l+2})$	Jeh and Widom [2002]
	PLM	$O(v^3)$	Fouss et al. [2007]
	ACT	$O(v^3)$	Fouss et al. [2007]
	RFK	$O(v^3)$	Chebotarev and Shamis [2006]
	BI	$O(cv^2k)$	Blondel et al. [2004]
Quasi-local	LPI	$O(lv^2k)$	Lü et al. [2009]
	LRW	$O(lv^2k)$	Liu and Lü [2010]
	SRW	$O(lv^2k)$	Liu and Lü [2010]
	ORA-CNI	$O(vk^6)$	Zhang et al. [2014]
	FL	$O(lv^2k)$	Papadimitriou et al. [2012]
	PFP	$O(vlk^l)$	Lichtenwalter et al. [2010]

Columns (from left to right): type of similarity-based link prediction technique, name of the technique, computational complexity of the technique, and original reference to the technique.

these techniques in order to study how they behave in different kinds of networks. As far as we know, the most complete existing comparison was Lü and Zhou [2011]. We have implemented and applied to different networks all the similarity-based techniques described previously. Other methods described in the following sections have not been considered due to their computational complexity and their need to tune different parameters, which would lead to unrepresentative results. In the following section, we describe the set of networks used in our experiments, how they were preprocessed, and how their structural properties were measured. In addition, results obtained are presented and discussed.

3.5.1. Datasets. Seven networks with different backgrounds and different topological properties were collected: a protein-protein interaction network of budding yeast (YST, Bu et al. [2003]), a neural network of Caenorhabditis elegans (CEL, Watts and Strogatz [1998]), a network describing face-to-face contacts of people during the exhibition Infectious: Stay Away in 2009 at the Science Gallery in Dublin (INF, Isella et al. [2011]), a frequent copurchasing network of books about US politics published in 2004 during the presidential election campaign and sold by Amazon (BCK, Krebs [2008]), a network of friendships among users of the hamsterster.com website (HMT, Kon [2014]), North American Transportation Atlas Data (NORTAD) flights from a 1997 network (USA, Batagelj and Mrvar [2006]), and a coauthorship network of scientists working on network theory and experiments (NSC, Newman [2006]). Each network was preprocessed to make its links undirected, isolated nodes were deleted, and duplicated links and self-loops were removed. Some important topological properties of these networks were computed and included in the supplementary material.

3.5.2. Evaluation and Results. To evaluate each technique, we conducted a fivefold cross-validation as described in the supplement. We report the precision (see Table II) and the AUC (see Table III) for each possible method and network combination. Methods with parameters were evaluated with different reasonable hand-picked values. The local interacting score has been tested with two, four, and six iterations and, as suggested by its authors, its best results were obtained with only two iterations. The Katz index (β), global Leicht-Holme-Newman index (ϕ), SimRank and local path index (β) with l fixed to 3 in both cases, and third-order RA-CNI (β) were tested with values 0.1, 0.01, and 0.001. The best results for Katz, SimRank, and local path were obtained with $\beta = 0.001$. The best results for the Leicht-Holme-Newman index were obtained with $\phi = 0.1$. ORA-CNI was set to $\beta = 0.001$. All random-walk-based methods with an α parameter were tested for 0.5, 0.7, and 0.9, finally selecting 0.5 in all cases since the results were very stable for all the tested parameter values. Finally, local random walk, superposed random walk, and PropFlow predictors were tested with parameters t and l set to 3, 5, and 7. The best average precision was obtained choosing 3 in all cases.

The number of times that each method appears in the top five for each network is summarized in Table IV for precision and AUC. Different conclusions can be extracted from our empirical results.

Global techniques obtain the worst results on average. Their poorer performance, and the fact that tuning their parameters strongly penalizes indirect paths, suggests that global methods are worse because they tend to consider too much noise. It can be seen that global methods are less present in the top five. Only RWR, FP, MERW, and RFK reach the top five on a few occasions in networks with a low average clustering coefficient.

Local techniques work surprisingly well. Some local methods reach the top five a few times, including AA, CAR-based methods, and the mutual information technique; however, RA and its variation RA-CNI, the LNB-RA method, and IA techniques are present in the top five a reasonable number of times when compared to other methods. These results suggest that most of the information that can be used to perform link prediction is of a local nature.

Quasi-local techniques also obtain good results on average. ORA-CNI, PFP, LRW, and SRW appear among the best for almost all networks. The SRW technique is the method that has been shown to obtain the bests results on average in our experiments.

Finally, the variety of methods in the top of the ranking shows that the performance of each technique strongly depends on the structural properties of the network. This highlights the importance of analyzing the properties of the network before choosing a particular link prediction technique. As we observe in our results, the quality of

Table II. Precision Results

Method	YST	CEL	INF	BCK	HMT	USA	NSC
CN	0.0876	0.1119	0.3484	0.2101	0.2453	0.4091	0.3561
AA	0.1080	0.1532	0.4080	0.2608	0.3267	0.4558	0.6217
RA	0.0876	0.1448	0.4163	0.2563	0.3959	0.5160	**0.6652**
RA-CNI	0.0951	0.1490	0.4242	0.2630	**0.4406**	0.4965	0.6306
PA	0.0310	0.1051	0.0848	0.1820	0.0787	0.3819	0.1932
JA	0.0030	0.0373	0.4104	0.1297	0.2472	0.1081	0.4884
SA	0.0026	0.0340	0.4188	0.1395	0.2409	0.0866	0.4997
SO	0.0030	0.0373	0.4104	0.1297	0.2472	0.1081	0.4884
HPI	0.0000	0.0000	0.2764	0.1463	0.0000	0.0000	0.0005
HDP	0.0104	0.0364	0.4017	0.1053	0.2461	0.0810	0.4208
LLHN	0.0002	0.0023	0.1271	0.0897	0.0817	0.0104	0.2334
IA1	0.1079	0.1569	0.4195	0.2619	0.4093	0.4638	0.6170
IA2	0.0855	0.1493	**0.4268**	0.2427	0.4199	0.4927	0.6596
MI	0.1228	0.1676	0.4040	0.2313	0.3324	0.4365	0.5125
LNB-CN	0.1189	0.1569	0.3964	0.2494	0.2996	0.4440	0.5414
LNB-AA	0.1190	0.1555	0.4036	0.2534	0.3574	0.4685	0.6362
LNB-RA	0.0954	0.1462	0.4105	0.2540	0.4019	0.5169	0.6610
CAR-CN	0.1073	0.1240	0.3942	0.2029	0.2816	0.4228	0.3914
CAR-AA	0.1241	0.1480	0.4047	0.2222	0.3191	0.4322	0.5074
CAR-RA	0.1245	0.1560	0.4148	0.2517	0.3873	0.4487	0.5074
FSW	0.0504	0.0792	0.3637	0.1349	0.2266	0.2158	0.4849
LIT	0.1049	0.1220	0.4253	0.1814	0.4097	0.3984	0.5022
NSP	0.0000	0.0001	0.0001	0.0007	0.0000	0.0001	0.0003
KI	0.1186	0.1513	0.3924	0.2471	0.2595	0.4332	0.4300
GLHN	0.0876	0.1119	0.3484	0.2101	0.2453	0.4091	0.3561
RW	0.0891	0.1276	0.2555	0.1927	0.2639	0.1599	0.4282
RWR	0.1175	**0.1983**	0.4090	0.2268	0.3751	0.3316	0.5292
FP	0.1013	0.1643	0.3508	0.2449	0.2845	0.4059	0.4303
MERW	0.0129	0.0666	0.2575	0.1565	0.2618	0.0927	0.5036
SR	0.0009	0.0033	0.1309	0.1066	0.0877	0.0127	0.2918
PLM	0.0176	0.1066	0.1703	0.2494	0.0017	0.3782	0.0149
ACT	0.0284	0.0889	0.1826	0.2199	0.0807	0.3791	0.0543
RFK	0.0260	0.0847	0.2329	0.1746	0.2381	0.0814	0.4712
BI	0.0710	0.1359	0.1483	0.2221	0.0988	0.3923	0.1940
LPI	0.1069	0.1438	0.3852	0.2404	0.1723	0.4200	0.4194
LRW	**0.1857**	0.1839	0.3819	0.2177	0.4232	0.4972	0.4927
SRW	0.1380	0.1657	0.4123	0.2540	0.4265	**0.5230**	0.6580
ORA-CNI	0.0993	0.1490	0.4242	**0.2630**	0.4405	0.4967	0.6601
FL	0.1069	0.1443	0.3848	0.2358	0.2328	0.4191	0.4194
PFP	0.0405	0.1331	0.1834	0.2110	0.2456	0.1449	0.4776

Average precision of the five iterations of the cross-validation performed for each method and network pair.

the results is related to the average clustering coefficient of the nodes with degree above one. This is reasonable since most link prediction techniques are variations of counting shared neighbors, and the count of shared neighbors increases as the clustering coefficient does. Another variable that seems to play an important role is the average degree. This makes sense since as we know the more neighbors there are of a node, the more information we have to predict new links for it. However, revealing

Table III. AUC Results

Method	YST	CEL	INF	BCK	HMT	USA	NSC
CN	0.6850	0.8274	0.9264	0.8691	0.9523	0.9278	0.9056
AA	0.6855	0.8450	0.9300	0.8782	0.9553	0.9391	0.9061
RA	0.6854	0.8485	0.9305	0.8801	0.9561	0.9439	0.9061
RA-CNI	0.6854	0.8495	0.9307	0.8803	0.9564	0.9433	0.9061
PA	0.6846	0.8091	0.8991	0.8505	0.9386	0.9177	0.9043
JA	0.6837	0.7766	0.9285	0.8558	0.9492	0.8926	0.9059
SA	0.6837	0.7831	0.9285	0.8621	0.9501	0.8995	0.9059
SO	0.6837	0.7766	0.9285	0.8558	0.9492	0.8926	0.9059
HPI	0.6834	0.7933	0.9255	0.8671	0.9484	0.8666	0.9058
HDP	0.6836	0.7680	0.9277	0.8475	0.9483	0.8878	0.9057
LLHN	0.6828	0.7276	0.9195	0.8314	0.9411	0.7821	0.9056
IA1	0.6854	0.8490	0.9305	0.8791	0.9562	0.9418	0.9061
IA2	0.6853	0.8483	0.9306	0.8795	0.9562	0.9434	0.9061
MI	0.6527	0.8325	0.9083	0.8426	0.9379	0.9089	0.7765
LNB-CN	0.6858	0.8411	0.9263	0.8717	0.9541	0.9337	0.9057
LNB-AA	0.6858	0.8445	0.9285	0.8765	0.9557	0.9403	0.9059
LNB-RA	0.6857	0.8451	0.9295	0.8772	0.9561	0.9440	0.9059
CAR-CN	0.6850	0.8272	0.9264	0.8682	0.9525	0.9269	0.9056
CAR-AA	0.5792	0.7130	0.8254	0.7224	0.8832	0.8971	0.7554
CAR-RA	0.5792	0.7140	0.8255	0.7230	0.8838	0.9001	0.7554
FSW	0.6839	0.7822	0.9256	0.8535	0.9473	0.8949	0.9058
LIT	0.6730	0.8313	0.9206	0.8550	0.9501	0.9164	0.8424
NSP	0.7887	0.7443	0.9121	0.8456	0.9423	0.8135	0.9142
KI	0.8044	0.8507	0.9528	0.8946	0.9630	0.9180	0.9147
GLHN	0.6850	0.8274	0.9264	0.8691	0.9523	0.9278	0.9056
RW	0.7811	0.8354	0.9494	0.8851	0.9551	0.8796	0.9151
RWR	0.8029	**0.8967**	**0.9637**	**0.9183**	0.9681	0.9362	0.8892
FP	0.8113	0.8873	0.9599	0.9018	0.9674	0.9382	0.9154
MERW	0.7806	0.8519	0.9454	0.8906	0.9575	0.8809	0.9151
SR	0.6828	0.7310	0.9200	0.8334	0.9414	0.7856	0.9056
PLM	0.7725	0.8480	0.9439	0.8908	0.6435	0.9407	0.5263
ACT	0.7659	0.7370	0.7969	0.7456	0.8793	0.8749	0.5654
RFK	0.7964	0.8614	0.9549	0.8984	0.9593	0.9106	0.9150
BI	0.7784	0.7159	0.7730	0.8166	0.8800	0.8674	0.9081
LPI	0.8025	0.8345	0.9501	0.8882	0.9591	0.9136	0.9137
LRW	0.8164	0.8874	0.9514	0.8928	0.9647	0.9291	0.8536
SRW	**0.8210**	0.8936	0.9608	0.9139	**0.9726**	**0.9463**	0.9164
ORA-CNI	0.8135	0.8539	0.9515	0.8977	0.9682	0.9386	0.9164
FL	0.8025	0.8342	0.9500	0.8882	0.9619	0.9112	0.9137
PFP	0.8159	0.8645	0.9568	0.9081	0.9636	0.8899	**0.9171**

Average AUC of the five iterations of the cross-validation performed for each method and network pair.

which specific properties play such an important role in link prediction is still an unsolved problem that requires further work.

4. PROBABILISTIC AND STATISTICAL APPROACHES

Many network formation models have been successfully described in terms of statistical and probabilistic concepts [Goldenberg et al. 2010]. These studies have opened the door to link prediction techniques based on statistical analysis and probability theory. These

Table IV. Summary of Methods Appearing in the Top Five of the Rank

Method	1st	2nd	3rd	Top 5
AA				1
RA	1		1	3
RA-CNI	1	1		3
IA1			1	2
IA2	1			3
MI			1	2
LNB-RA		2		2
CAR-AA				1
CAR-RA			1	1
LIT		1		1
RWR	1			1
FP				1
LRW	1	1		4
SRW	1	1	1	5
ORA-CNI	1	1	2	5

Method	1st	2nd	3rd	Top 5
RA			1	1
RA-CNI				1
IA2				1
LNB-RA		1		1
RWR	3		1	4
FP			1	6
MERW				1
RFK				2
LRW		1	1	3
SRW	3	3	1	7
ORA-CNI		2		3
PFP	1		2	5

Number of times that each method appears in the first, second, and third position and in the top five for all networks according to precision (left table) and AUC (right table). The rows of methods that never are in the top five of the rank are omitted.

approaches usually assume that the network has a known structure. They build a model that fits the structure and estimate model parameters using statistical methods. These parameters are used to compute the formation probability of each nonobserved link. These probability values can be used to rank potential links as we did in similarity-based methods.

4.1. The Hierarchical Structure Model

Some studies show that many real networks are hierarchically organized, including metabolic networks, protein interaction networks, Internet domains, and some social networks like actor networks [Ravasz and Barabási 2003]. In hierarchical networks, nodes with higher degree are expected to have a lower clustering coefficient than lower-degree nodes. Hub nodes weakly connect isolated communities of highly clustered nodes. This way, a hierarchical structure is formed.

The method proposed by Clauset et al. [2008] represents a hierarchically structured network by a dendrogram with $|V|$ leaves and $|V| - 1$ internal nodes. Each leaf represents a node from the network and each internal node represents a relationship among its descendant nodes in the dendrogram. Each internal node n has an associated probability p_n, which is equal to the probability of a link between nodes of both branches descending from it. Each network has multiple representations based on dendrograms depending on how internal nodes are set. Given a dendrogram representation D of the network, let e_n be the number of links in the network connecting nodes that have internal node n as their lowest common ancestor in D. The likelihood of dendrogram D together with the set of internal node probabilities can be estimated as

$$\mathcal{L}(D, \{p_n\}) = \prod_{n \in D} p_n^{e_n} (1 - p_n)^{l_n r_n - e_n},$$

where l_n and r_n are, respectively, the numbers of leaves in the left and the right subtrees with root n. If the dendrogram D is fixed, its likelihood can be maximized by a set of probabilities $\overline{p}_n$ computed as

$$\overline{p}_n = \frac{e_n}{l_n r_n}.$$

ALGORITHM 2: Link Prediction Based on the Hierarchical Structure Model.

Input: Network $G = (V, E)$, number n of dendrograms to sample.
Output: Probability $P_{x,y}$ for all unconnected pairs of nodes.
$Samples = \emptyset$;
for i ***from*** 1 ***to*** n **do**
    Initialize the Markov chain with a random dendrogram;
    Run Monte Carlo algorithm until equilibrium is reached;
    Insert resulting dendrogram D into $Samples$;
end
for ***each*** $e_{x,y}$ ***in*** $U_G - E$ **do**
    $avg\_prob = 0$;
    for ***each*** $sample$ ***in*** $Samples$ **do**
        $r \leftarrow$ lower common ancestor of x and y in $sample$;
        $avg\_prob \leftarrow avg\_prob + \frac{\overline{p}_r}{|Samples|}$;
    end
    $P_{x,y} = avg\_prob$;
end

$\overline{p}_n$ represents the ratio of the number of actual edges with respect to the number of potential ones. Based on this result, the e_n term can be removed from the likelihood formula to estimate the likelihood of a dendrogram at its maximum as

$$\mathcal{L}(D) = \prod_{n \in D} [\overline{p}_n^{\overline{p}_n} (1 - \overline{p}_n)^{1-\overline{p}_n}]^{l_n r_n}.$$

Once the theoretical background is set, these results can be used to perform link prediction. A Markov chain Monte Carlo method [Geyer 1992] is used to sample a set of dendrograms with a probability proportional to their likelihood. The transitions between dendrograms consist of rearranging subtrees of the current dendrogram in another order. The complete procedure, which computes the probability of link formation between each pair of unconnected nodes, is described in Algorithm 2.

4.2. The Stochastic Block Model

Most networks do not fit in a hierarchical schema. A more general approach is to consider that nodes are distributed in communities or blocks. The probability of link formation between two nodes directly depends on the block they belong to [Guimerà and Sales-Pardo 2009]. In this model, we are required to compute a partition $\mathcal{M}$ of the network where each node is assigned to one group or block $m \in \mathcal{M}$. Given a partition, the likelihood of the network structure can be estimated as

$$\mathcal{L}(G|\mathcal{M}) = \prod_{a,b \in \mathcal{M}} p_{a,b}^{l_{a,b}} (1 - p_{a,b})^{r_{a,b} - l_{a,b}},$$

where $l_{a,b}$ is the number of edges between nodes in groups a and b, $|\{e_{x,y} : x \in a, y \in b\}|$; and $r_{a,b}$ is the number of pairs between nodes of both groups, which are $|a||b|$ when $a \neq b$ and $|a|(|a| - 1)$ when $a = b$. This likelihood is maximized for

$$\overline{p}_{a,b} = \frac{l_{a,b}}{r_{a,b}}.$$

Applying the Bayes theorem, the probability of a link with maximum likelihood can be computed as

$$P_{x,y} = \frac{\sum_{\mathcal{M} \in \omega} \mathcal{L}(e_{x,y} \in E|\mathcal{M}) \mathcal{L}(G|\mathcal{M}) p(\mathcal{M})}{\sum_{\mathcal{M}' \in \omega} \mathcal{L}(G|\mathcal{M}') p(\mathcal{M}')},$$

ALGORITHM 3: Link Prediction Based on the Cycle Formation Model.

Input: Network $G = (V, E)$, model degree k.
Output: Probability $P_{x,y}$ for all unconnected pairs of nodes.
Compute Generalized Clustering Coefficients $C(2), \ldots, C(k)$;
$c_1 =$ Connecting probability in random graph with same degree distribution that G;
$c_2 = \frac{(1-c_1)C(2)}{c_1 - 2c_1C(2)+C(2)}$;
for i ***from*** 3 ***to*** k **do**
    $c_i \leftarrow 0.5$;
end
for i ***from*** 3 ***to*** k **do**
    $c_i \leftarrow \arg\min_{c_i} |C(i) - f(c_1, \ldots, c_k)|$;
end
for ***each*** $e_{x,y}$ ***in*** $U_G - E$ **do**
    $P_{x,y} \leftarrow p_{x,y}(c_1, \ldots, c_k)$;
end

where ω is the set of possible partitions. It should be noted that ω grows fast as the number of nodes in the network increases ($\omega \in O(2^{|V|})$), which makes this approach impractical for large networks. The Metropolis algorithm can be used to sample partitions, but this process is still computationally expensive.

4.3. The Cycle Formation Model

The cycle formation model is based on the assumption that networks have the tendency to close cycles in their formation process [Huang 2006]. This assumption matches with other techniques like the common neighbors method, which counts the number of cycles of length three that would be formed if the evaluated link existed. This method tries to capture longer cycles by extending the overall clustering coefficient to a generalized clustering coefficient. This generalized clustering coefficient is defined as

$$C(k) = \frac{\text{number of cycles of length } k}{\text{number of paths of length } k},$$

where k is the length of the cycles under analysis.

A cycle formation model of degree k, denoted as $CF(k)$ with $k > 0$, characterizes each order formation mechanism, $g(1), \ldots, g(k)$, by a single coefficient, $c_1, \ldots, c_k$, which describes the conditional probability of k-order cycle formation. The expected clustering coefficient of degree k based on this model can be estimated as

$$f(c_1, \ldots, c_k) = \sum_i |G_i| Pr(G_i) Pr(e_{1,k+1} \in E | G_i),$$

where G_i is the set of possible connected graphs with i nodes for each order in this model. Finally, given the coefficients, the probability of the existence of a link is computed as

$$p_{x,y}(c_1, \ldots, c_k) = \frac{c_1 \prod_{i=2}^{k} c_i^{|paths^i_{x,y}|}}{c_1 \prod_{i=2}^{k} c_i^{|paths^i_{x,y}|} + (1-c_1) \prod_{i=2}^{k} (1-c_i)^{|paths^i_{x,y}|}}.$$

The whole link prediction method based on this cycle formation model is reproduced in Algorithm 3.

4.4. The Local Co-Occurrence Model

The probabilistic methods presented previously are prohibitive for large networks due to their high computational complexity. The local co-occurrence model proposes a

ALGORITHM 4: Link Prediction Based on the Local Co-Occurrence Model.

Input: Network $G = (V, E)$, central neighborhood set max size t, max path length k.
Output: Probability $P_{x,y}$ for all unconnected pairs of nodes.

```
for each e_{x,y} in U − E do
    C_{x,y} = ∅;
    for l from 2 to k do
        p_i ← Compute and sort by length and frequency paths^l_{x,y};
        for each p in p_i do
            if |C_{x,y}| < t then
                Insert all nodes in p into C_{x,y};
            end
        end
    end
    NDI = Compute nonderivable itemsets from C_{x,y};
    R_{x,y} = ∅;
    for each ndi in NDI do
        if ndi in C_{x,y} then
            Insert ndi into R_{x,y};
        end
    end
    M = Initialize Markov Random Fields using C_{x,y} and R_{x,y};
    while not M satisfies all constrains in R_{x,y} do
        for each r in R_{x,y} do
            Update M to force satisfying r;
        end
    end
    P_{x,y} = Infer probability of e_{x,y} from M;
end
```

scalable probabilistic method based on local topological features of the network [Wang et al. 2007].

A set $C_{x,y}$ composed of relevant nodes, called the central neighborhood, is computed for each pair of nodes x and y. These sets can be obtained using different topological measures. The original paper proposes to compute these sets by obtaining all simple paths (without cycles) of length $1, \ldots, k$. The t nodes in the most frequent paths are selected as the central neighborhood set of a pair of nodes. In addition, a collection of nonderivable itemsets (NDIs) is efficiently computed for each of these pairs by a depth-first search [Calders and Goethals 2005]. Nonderivable itemsets are those itemsets whose occurrence statistics cannot be inferred from other itemset patterns. These itemsets provide nonredundant constraints that can be used to learn probabilistic models without losing information.

These sets are used to learn a Markov random field (MRF) undirected graph model. This model is iteratively built satisfying constraints associated to the NDI sets. Finally, the built model allows one to compute the probability of existence of each link. The final link prediction method appears in Algorithm 4.

5. ALGORITHMIC METHODS

All the approaches presented in the previous sections are based on computing a score for each nonobserved link by defining a similarity or a probability function. However, link prediction can also benefit from other algorithmic approaches, including supervised learning and optimization techniques. These approaches have been less explored in the literature of link prediction but present interesting properties.

5.1. Classifier-Based Methods

The link prediction problem can be approached by classical supervised learning techniques. It can be seen as a classification problem with two classes: existence and absence of link. This is a very powerful technique since it can use any topological property and measure, or even any other link prediction measure as a feature. This approach, however, has to deal with the well-known class imbalance problem [Kotsiantis et al. 2006], since almost all real networks are sparse; that is, the number of absent links is extremely higher than the number of existent links.

Several classifier-based approaches have been proposed. Almost any type of classifier can be used. Some works have compared different classifiers including decision trees, support vector machines (SVMs), k-nearest neighbors, multilayer perceptrons, radial basis function networks, naive Bayes, and different ensembles of these classifiers [Al Hasan et al. 2006]. Other authors have obtained good results using random forest classifiers [Cukierski et al. 2011]. Random forests are decision tree ensembles trained on the same training set but using different subsets of the available features.

Many classifier-based methods do not rank possible links like similarity-based or probabilistic methods. This property makes their comparison harder, since the number of predicted links in each class cannot be controlled in this case.

5.2. Metaheuristic-Based Methods

Link formation is a complex process with a large number of factors involved. All approaches are heuristic, in the sense that they try to outperform a random baseline predictor by making some assumptions about link formation in the studied network. Assuming that all links are formed by the same mechanism is an oversimplification of the problem.

Recently, Bliss et al. [2014] proposed an approach based on an evolutionary algorithm. Their method assumes that different link formation heuristics can coexist and cooperate in the same network. It uses an evolution strategy to optimize the influence of different base link predictors including local and global similarity-based indices and node similarity features in a Twitter reciprocal reply network. Each individual or candidate solution u in the population is a vector $w^{(u)}$ of as many real numbers as the number of heuristics being considered. Each of these values represents the weight or the influence of the heuristic in the network. Each candidate represents a similarity-based link predictor characterized by the similarity function

$$s(x, y) = \sum_{i=1}^{|w^{(u)}|} w_i^{(u)} s_i(x, y),$$

where $s_i(x, y)$ is the similarity function for the ith heuristic. The fitness of each candidate is defined as the precision obtained by applying it on a second training subset and a second test subset created from sampled links from the original training set. A covariance matrix adaptation evolution strategy (CMA-ES) is applied to generate new candidates with better fitness by creating new populations of candidates based on combinations and mutations of the previous ones [Hansen and Ostermeier 2001].

5.3. Factorization-Based Methods

Matrix factorization models have been widely used in recommender systems since they can extract latent features or use additional features to perform prediction [Koren et al. 2009]. For example, Menon and Elkan [2011] have suggested a latent feature learning method for link prediction composed of a latent vector $\overrightarrow{l_i}$ for each node i, a scaling factor $F_{x,y}$ for each link, weights for node features W_n, and a vector of weights for link

features $\overrightarrow{w_l}$. Furthermore, each node i has a vector of features $\overrightarrow{a_i}$ and each link has a vector of features $\overrightarrow{b_{x,y}}$.

In this model, given the latent vectors, the scaling factor, and the weights, a prediction score is computed for each pair of nodes x and y as

$$s(x, y) = \frac{1}{1 + \exp(-\overrightarrow{l_x}^T F \overrightarrow{l_y} - \overrightarrow{a_x}^T W_n \overrightarrow{a_y} - \overrightarrow{w_l}^T \overrightarrow{b_{x,y}})}.$$

Latent vectors, the scaling factor, and the weights are obtained in a training stage that optimizes the following function:

$$\min_{l,F,W_n,\overrightarrow{w_l}} \sum_{e_{x,y} \in E} \ell(A_{x,y}, s(x, y)) + \Omega(l, F, W_n, \overrightarrow{w_l})),$$

where ℓ is a loss function and Ω is a regularizer term to prevent overfitting. These terms can be selected to customize the model [Koren et al. 2009]. This training stage is performed using Stochastic Gradient Descent (SGD) until convergence.

6. PREPROCESSING METHODS

Preprocessing methods are also known as higher-level approaches or meta-approaches, since they are intended to be used in conjunction with other methods. The main goal of preprocessing approaches is to reduce the noise present in the networks as "weak" or "false" links, in order to improve the performance of the methods described previously.

6.1. Low-Rank Approximation

This method simplifies the structure of the network to reduce its noise using the adjacency matrix A representation of the graph by solving the low-rank approximation problem [Kunegis and Lommatzsch 2009]. This optimization problem tries to minimize a cost function that measures the fit between the original matrix and an approximation matrix of reduced rank. The rank of the approximated matrix is usually set to a relatively small number. This problem can be algorithmically solved in an efficient way using the SVD of the original matrix, which is a factorization of the form

$$A = U \Sigma V^T,$$

where U and V^T are unitary matrices and Σ is a diagonal matrix with no negative elements. There are different techniques to compute the SVD. The most basic approach relies on the fact that singular values are the square roots of the eigenvalues of AA^T. Indeed, given the expression of the decomposition, it can be stated that

$$AA^T = (U \Sigma V^T)(V \Sigma U^T) = U \Sigma^2 U^T.$$

Since the columns of U are eigenvectors of A, it can be obtained using a technique to compute the eigenvectors of a matrix, therefore allowing one to obtain the Σ matrix using simple linear algebra calculations. Unfortunately, this technique is not practical for large matrices due to lack of numerical accuracy. One of the most commonly used SVD algorithms, which shows a better accuracy, is given by Demmel and Kahan [1990].

Given the SVD of A, the low-rank matrix A' can be approximated by

$$A' = U_{1:|V|,1:k} \Sigma_{1:k,1:k} V^T_{1:k,1:|V|},$$

where k is the desired rank. The first eigenvectors explain most of the variance, so the low-rank approximated matrix maintains the overall structure of the graph but removes its less significant links.

6.2. Unseen Bigrams

A bigram is a sequence of two adjacent elements in a string composed of tokens or words. The frequency distribution of bigrams has been extensively studied in many applications such as linguistics, speech recognition, or cryptography. Unseen bigrams are valid bigrams not observed in a given string set. It has been observed that the same tokens in different bigrams with similar appearance distributions are likely to be interchangeable and to form unseen bigrams. For example, if we observe the bigrams "a house," "the house," "a tree," "the tree," and "a car," then we can infer that "the car" is an unseen bigram. The idea of "substitution" presented by unseen bigrams can be adapted to link prediction in order to reduce noise by replacing a node by its most similar nodes [Liben-Nowell 2005]. For example, the common neighbors similarity can be rewritten as

$$S(x, y) = |S_x^\delta \cap \Gamma_y|,$$

where S_x^δ is a set with the δ most similar nodes to x. The similarity method employed to obtain the set S_x^δ could also be the common neighbors similarity or any other similarity-based technique.

6.3. Filtering

Originally called clustering [Liben-Nowell 2005], we refer to it as filtering in order to avoid any ambiguity. Removing the weakest links (usually, those observed between nodes with a small number or no shared neighbors) could help improve the results obtained by link prediction methods due to the associated noise reduction. Most of the techniques presented in this survey assign a score $S(x, y)$ to nonobserved links, but they can be applied to observed links in order to estimate their strength. Therefore, the filtering approach consists of applying any link prediction technique that assigns a score to each pair of nodes between connected nodes to weigh the observed link and remove the fraction γ of weakest links to obtain a cleaned-up network.

7. LINK PREDICTION IN DIFFERENT KINDS OF NETWORKS

In this work, a large number of proposed techniques focusing only on undirected unweighed networks without attributes have been discussed. However, an increasing number of link prediction techniques for other types of networks and the consideration of nontopological additional information is also of interest in practice. The role of weak ties in weighted networks remains an open question [Lü and Zhou 2010]. Heterogeneous networks with different types of nodes and links require different approaches to those used in homogeneous networks [Sun et al. 2012]. Predicting not only a link but also its source and its destination in directed networks has recently started to be under study [Zhao et al. 2013]. Link prediction in signed networks, where links can be positive or negative, is another active area of research [Symeonidis and Tiakas 2014]. Time-aware methods can be proposed for networks with links labeled with their time of formation [Tylenda et al. 2009]. Aligned networks are sets of different networks partially matched by anchor nodes and links, and sets of networks representing the same agent in different domains are available, so specific techniques to exploit this additional information are being developed [Lu et al. 2010]. Most of the presented methods in our survey can be adapted to the characteristics of these real-world usage scenarios.

8. CONCLUSIONS

In this survey, we have performed, as far as we know, the most comprehensive study about the link prediction methods that have been proposed in terms of the number of methods and networks employed. These methods have been classified according to

a custom taxonomy based on their theoretical approach. The most important results that support specific link prediction techniques and network formation models have also been described.

A comprehensive experimentation has been performed for a large number of fundamental and state-of-the-art methods using a varied set of networks with different properties. It has been observed that new links can be better predicted using only local or quasi-local information in most networks. Considering indirect connections only adds noise and computational complexity to the link prediction problem.

Link prediction is a relatively young research area and many open challenges remain. Further studies are required in order to understand why some methods work better or worse than others depending on the network they are applied to. Studying which network structural properties lead to better performance for each technique is an open research problem. In addition, very few techniques adapt to the global structure of the network and no technique adapts to the local structure of networks. The main difficulty when dealing with complex networks in practice is their size, which limits the kinds of techniques that can be applied.

Link prediction remains an open research problem, given its importance in many applications. New techniques with better accuracy and performance tradeoffs are expected to be proposed in the forthcoming future.

REFERENCES

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

Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. 2006. Link prediction using supervised learning. In *Proceedings of the Workshop on Link Analysis, Counter-terrorism and Security (SDM'06)*.

Mohammad Al Hasan and Mohammed J. Zaki. 2011. A survey of link prediction in social networks. In *Social Network Data Analytics*. Springer, 243–275.

Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.

Vladimir Batagelj and Andrej Mrvar. 2006. Pajek datasets. Retrieved from http://vlado.fmf.uni-lj.si/pub/networks/data.

Indrajit Bhattacharya and Lise Getoor. 2007. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007), 5.

Catherine A. Bliss, Morgan R. Frank, Christopher M. Danforth, and Peter Sheridan Dodds. 2014. An evolutionary algorithm approach to link prediction in dynamic social networks. *Journal of Computational Science* 5, 5 (2014), 750–764.

Vincent D. Blondel, Anahí Gajardo, Maureen Heymans, Pierre Senellart, and Paul Van Dooren. 2004. A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM Review* 46, 4 (2004), 647–666.

Dongbo Bu, Yi Zhao, Lun Cai, Hong Xue, Xiaopeng Zhu, Hongchao Lu, Jingfen Zhang, Shiwei Sun, Lunjiang Ling, Nan Zhang, and others. 2003. Topological structure analysis of the protein–protein interaction network in budding yeast. *Nucleic Acids Research* 31, 9 (2003), 2443–2450.

Zdzislaw Burda, Jarek Duda, Jean-Marc Luck, and Bartek Waclaw. 2009. Localization of the maximal entropy random walk. *Physical Review Letters* 102, 16 (2009), 160602.

Toon Calders and Bart Goethals. 2005. Depth-first non-derivable itemset mining. In *Proceeding of the 2005 SIAM International Conference on Data Mining (SDM'05)*. 250–261.

Carlo Vittorio Cannistraci, Gregorio Alanis-Lobato, and Timothy Ravasi. 2013. From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. *Scientific Reports* 3 (2013), 1613.

Pavel Chebotarev and Elena Shamis. 2006. Matrix-forest theorems. *arXiv preprint math/0602575* (2006).

Hon Nian Chua, Wing-Kin Sung, and Limsoon Wong. 2006. Exploiting indirect neighbours and topological weight to predict protein function from protein–protein interactions. *Bioinformatics* 22, 13 (2006), 1623–1630.

Aaron Clauset, Cristopher Moore, and Mark E. J. Newman. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature* 453, 7191 (2008), 98–101.

William Cukierski, Benjamin Hamner, and Bo Yang. 2011. Graph-based features for supervised link prediction. In *Proceedings of the 2011 International Joint Conference on Neural Networks (IJCNN'11)*. IEEE, 1237–1244.

James Demmel and William Kahan. 1990. Accurate singular values of bidiagonal matrices. *SIAM Journal on Scientific and Statistical Computing* 11, 5 (1990), 873–912.

Yuxiao Dong, Qing Ke, Bai Wang, and Bin Wu. 2011. Link prediction based on local information. In *Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining (ASONAM'11)*. IEEE, 382–386.

Santo Fortunato. 2010. Community detection in graphs. *Physics Reports* 486, 3 (2010), 75–174.

Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. 2007. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering* 19, 3 (2007), 355–369.

Charles J. Geyer. 1992. Practical Markov chain Monte Carlo. *Statistical Science* 7, 4 (1992), 473–483.

Anna Goldenberg, Alice X. Zheng, Stephen E. Fienberg, and Edoardo M. Airoldi. 2010. A survey of statistical network models. *Foundations and Trends in Machine Learning* 2, 2 (2010), 129–233.

Roger Guimerà and Marta Sales-Pardo. 2009. Missing and spurious interactions and the reconstruction of complex networks. *Proceedings of the National Academy of Sciences* 106, 52 (2009), 22073–22078.

Lieve Hamers, Yves Hemeryck, Guido Herweyers, Marc Janssen, Hans Keters, Ronald Rousseau, and André Vanhoutte. 1989. Similarity measures in scientometric research: The Jaccard index versus Salton's cosine formula. *Information Processing & Management* 25, 3 (1989), 315–318.

Nikolaus Hansen and Andreas Ostermeier. 2001. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation* 9, 2 (2001), 159–195.

Zan Huang. 2006. Link prediction based on graph topology: The predictive value of the generalized clustering coefficient. In *Proceedings of the Workshop on Link Analysis (KDD'06)*.

Zan Huang, Xin Li, and Hsinchun Chen. 2005. Link prediction approach to collaborative filtering. In *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'05)*. ACM, 141–142.

Lorenzo Isella, Juliette Stehlé, Alain Barrat, Ciro Cattuto, Jean-François Pinton, and Wouter Van den Broeck. 2011. What's in a crowd? Analysis of face-to-face behavioral networks. *Journal of Theoretical Biology* 271, 1 (2011), 166–180.

Paul Jaccard. 1901. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la Société Vaudoise des Sciences Naturelles* 37 (1901), 547579.

Glen Jeh and Jennifer Widom. 2002. SimRank: A measure of structural-context similarity. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*. ACM, 538–543.

Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (1953), 39–43.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

Gueorgi Kossinets and Duncan J. Watts. 2006. Empirical analysis of an evolving social network. *Science* 311, 5757 (2006), 88–90.

Sotiris Kotsiantis, Dimitris Kanellopoulos, Panayiotis Pintelas, and others. 2006. Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering* 30, 1 (2006), 25–36.

Valdis Krebs. 2008. A network of books about recent US politics sold by the online bookseller amazon.com. Retrieved from http://www.orgnet.com.

Valdis E. Krebs. 2002. Mapping networks of terrorist cells. *Connections* 24, 3 (2002), 43–52.

Jérôme Kunegis. 2014. Hamsterster full network dataset – KONECT. Retrieved from http://konect.uni-koblenz.de/networks/petster-hamster.

Jérôme Kunegis and Andreas Lommatzsch. 2009. Learning spectral graph transformations for link prediction. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML'09)*. ACM, 561–568.

Elizabeth A. Leicht, Petter Holme, and Mark E. J. Newman. 2006. Vertex similarity in networks. *Physical Review E* 73, 2 (2006), 026120.

Rong-Hua Li, Jeffrey Xu Yu, and Jianquan Liu. 2011. Link prediction: The power of maximal entropy random walk. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM'15)*. ACM, 1147–1156.

David Liben-Nowell. 2005. *An Algorithmic Approach to Social Networks*. Ph.D. Dissertation. Massachusetts Institute of Technology.

David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology* 58, 7 (2007), 1019–1031.

Ryan N. Lichtenwalter, Jake T. Lussier, and Nitesh V. Chawla. 2010. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'10)*. ACM, 243–252.

Guimei Liu, Jinyan Li, and Limsoon Wong. 2008. Assessing and predicting protein interactions using both local and global network topological metrics. *Genome Informatics* 21 (2008), 138–149.

Weiping Liu and Linyuan Lü. 2010. Link prediction based on local random walk. *EPL (Europhysics Letters)* 89, 5 (2010), 58007.

Zhen Liu, Qian-Ming Zhang, Linyuan Lü, and Tao Zhou. 2011. Link prediction in complex networks: A local Naïve Bayes model. *EPL (Europhysics Letters)* 96, 4 (2011), 48007.

Linyuan Lü, Ci-Hang Jin, and Tao Zhou. 2009. Similarity index based on local paths for link prediction of complex networks. *Physical Review E* 80, 4 (2009), 046122.

Linyuan Lü and Tao Zhou. 2010. Link prediction in weighted networks: The role of weak ties. *EPL (Europhysics Letters)* 89, 1 (2010), 18001.

Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and Its Applications* 390, 6 (2011), 1150–1170.

Zhengdong Lu, Berkant Savas, Wei Tang, and Inderjit S. Dhillon. 2010. Supervised link prediction using multiple sources. In *Proceedings of the 2010 IEEE 10th International Conference on Data Mining (ICDM'10)*. IEEE, 923–928.

Víctor Martínez, Fernando Berzal, and Juan-Carlos Cubero. 2016. Adaptive degree penalization for link prediction. *Journal of Computational Science* 13 (2016), 1–9.

Víctor Martínez, Carlos Cano, and Armando Blanco. 2014. ProphNet: A generic prioritization method through propagation of information. *BMC Bioinformatics* 15, Suppl 1 (2014), S5.

Bruce McCune, James B. Grace, and Dean L. Urban. 2002. *Analysis of Ecological Communities*. Vol. 28. MjM Software Design, Gleneden Beach, Oregon.

Aditya Krishna Menon and Charles Elkan. 2011. Link prediction via matrix factorization. In *Machine Learning and Knowledge Discovery in Databases*. Springer, 437–452.

Michael Mitzenmacher. 2004. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics* 1, 2 (2004), 226–251.

Mark E. J. Newman. 2001. Clustering and preferential attachment in growing networks. *Physical Review E* 64, 2 (2001), 025102.

Mark E. J. Newman. 2003. The structure and function of complex networks. *SIAM Review* 45, 2 (2003), 167–256.

Mark E. J. Newman. 2006. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E* 74, 3 (2006), 036104.

Joshua O'Madadhain, Jon Hutchins, and Padhraic Smyth. 2005. Prediction and ranking algorithms for event-based network data. *ACM SIGKDD Explorations Newsletter* 7, 2 (2005), 23–30.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web.* Technical Report 1999-66. Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 7043 (2005), 814–818.

Alexis Papadimitriou, Panagiotis Symeonidis, and Yannis Manolopoulos. 2012. Fast and accurate link prediction in social networking systems. *Journal of Systems and Software* 85, 9 (2012), 2119–2132.

Milen Pavlov and Ryutaro Ichise. 2007. Finding experts by link prediction in co-authorship networks. *FEWS* 290 (2007), 42–55.

Karl Pearson. 1905. The problem of the random walk. *Nature* 72, 1865 (1905), 294.

Yanjun Qi, Ziv Bar-Joseph, and Judith Klein-Seetharaman. 2006. Evaluation of different biological data and computational classification methods for use in protein interaction prediction. *Proteins: Structure, Function, and Bioinformatics* 63, 3 (2006), 490–500.

Erzsébet Ravasz and Albert-László Barabási. 2003. Hierarchical organization in complex networks. *Physical Review E* 67, 2 (2003), 026112.

Erzsébet Ravasz, Anna Lisa Somera, Dale A. Mongru, Zoltán N. Oltvai, and A.-L. Barabási. 2002. Hierarchical organization of modularity in metabolic networks. *Science* 297, 5586 (2002), 1551–1555.

Matthew Richardson and Pedro Domingos. 2002. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*. ACM, 61–70.

Gerard Salton and Michael J. McGill. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York.

J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. 2007. Collaborative filtering recommender systems. In *The Adaptive Web*. Springer, 291–324.

Benno Schwikowski, Peter Uetz, and Stanley Fields. 2000. A network of protein–protein interactions in yeast. *Nature Biotechnology* 18, 12 (2000), 1257–1261.

Thorvald Sørensen. 1948. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. *Biologiske Skrifter* 5 (1948), 1–34.

Daniel Spielman. 2009. Spectral graph theory. *Lecture Notes, Yale University* (2009), 740–0776.

Yizhou Sun, Jiawei Han, Charu C. Aggarwal, and Nitesh V. Chawla. 2012. When will it happen? Relationship prediction in heterogeneous information networks. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM'12)*. ACM, 663–672.

Panagiotis Symeonidis and Eleftherios Tiakas. 2014. Transitive node similarity: Predicting and recommending links in signed social networks. *World Wide Web* 17, 4 (2014), 743–776.

Roberto Tamassia. 2013. *Handbook of Graph Drawing and Visualization*. CRC Press.

Fei Tan, Yongxiang Xia, and Boyao Zhu. 2014. Link prediction in complex networks: A mutual information perspective. *PloS One* 9, 9 (2014), e107056.

Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast random walk with restart and its applications. In *Proceedings of the 6th International Conference on Data Mining (ICDM'06)*. IEEE Computer Society, Washington, DC, 613–622. DOI:http://dx.doi.org/10.1109/ICDM.2006.70

Tomasz Tylenda, Ralitsa Angelova, and Srikanta Bedathur. 2009. Towards time-aware link prediction in evolving social networks. In *Proceedings of the 3rd Workshop on Social Network Mining and Analysis (SNA-KDD'09)*. ACM, 9.

Oron Vanunu and Roded Sharan. 2008. A propagation-based algorithm for inferring gene-disease associations. In *German Conference on Bioinformatics (GCB'08)*. Citeseer, 54–52.

Srinivas Virinchi and Pabitra Mitra. 2013. Similarity measures for link prediction using power law degree distribution. In *International Conference on Neural Information Processing (ICONIP'13)*. Springer, 257–264.

Chao Wang, Venu Satuluri, and Srinivasan Parthasarathy. 2007. Local probabilistic models for link prediction. In *Proceedings of the 7th IEEE International Conference on Data Mining, 2007 (ICDM'07)*. IEEE, 322–331.

Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou. 2015. Link prediction in social networks: The state-of-the-art. *Science China Information Sciences* 58, 1 (2015), 1–38.

Duncan J. Watts and Steven H. Strogatz. 1998. Collective dynamics of small-world networks. *Nature* 393, 6684 (1998), 440–442.

Jennifer Xu and Hsinchun Chen. 2008. The topology of dark networks. *Communications of the ACM* 51, 10 (2008), 58–65.

Jianpei Zhang, Yuan Zhang, Hailu Yang, and Jing Yang. 2014. A link prediction algorithm based on socialized semi-local information. *Journal of Computational Information Systems* 10, 10 (2014), 4459–4466.

Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. 2013. Link prediction for partially observed networks. *arXiv preprint arXiv:1301.7047* (2013).

Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. 2009. Predicting missing links via local information. *European Physical Journal B* 71, 4 (2009), 623–630.

Received May 2015; revised September 2016; accepted October 2016