# Chapter 4
# A Survey of Link Mining Tasks for Analyzing Noisy and Incomplete Networks

**Galileo Mark Namata, Hossam Sharara, and Lise Getoor**

**Abstract** Many data sets of interest today are best described as networks or graphs of interlinked entities. Examples include Web and text collections, social networks and social media sites, information, transaction and communication networks, and all manner of scientific networks, including biological networks. Unfortunately, often the data collection and extraction process for gathering these network data sets is imprecise, noisy, and/or incomplete. In this chapter, we review a collection of link mining algorithms that are well suited to analyzing and making inferences about networks, especially in the case where the data is noisy or missing.

## 4.1 Introduction

A key emerging challenge for data mining is tackling the problem of mining richly structured, heterogeneous data sets. These kinds of data sets are best described as networks or graphs, where the nodes can be of different types, and the edges (or hyperedges) can represent different kinds of links. As evidenced by this volume, there has been a growing interest in methods which can mine and make inferences about such data (see also an earlier survey article and special issue issue of KDD Explorations [41]).

In the context of network data, statistical inference can be used in a variety of ways. Two of the most common are for inferring missing information and identifying (and correcting) incorrect network data. Furthermore, one way of understanding the different inference tasks in network data is according to whether they predict (or correct) information associated with nodes, edges, or larger subgraphs of the network. The inference task may be about inferring missing values (such as the label or attribute values for a node or edge), reasoning about the existence of nodes and edges (such as predicting whether two nodes should be merged because they refer to the

L. Getoor (✉)
Department of Computer Science, University of Maryland, College Park, MD, USA
e-mail: getoor@cs.umd.edu

same underlying entity, predicting whether a relationship exists), or reasoning about the existence of groupings of nodes and edges (group or community detection).

Examples of work applying statistical inference to infer missing or incorrect network data can be found in various domains. For example, in the social sciences, there is interest in studying human interaction from large online social networks [69, 113]. In these large online networks, individuals may own multiple accounts which need to be resolved to get an accurate count of the individuals in the network. Furthermore, the relationships (e.g., unspecified friends), attributes (e.g., gender), and membership in social groups (e.g., political affiliation) of the individuals of interest may not be given and need to be inferred. Similarly, in biology, there is interest in gaining new insight into biological processes by studying protein–protein interaction (PPI) networks [50, 107, 118]. The high-throughput methods typically used to create and annotate these networks are notoriously noisy and incomplete. Even the proteins of the most studied organisms, yeast, are not completely annotated with their functions and complex memberships and it is estimated that up to 52% the interactions for the current yeast PPI are spurious [50]. Analysis of these PPI networks requires applying statistical inference to infer the missing and correct function, interaction, and complex membership of proteins. As a final example, in computer networks, there is work in creating a map of the Internet to understand its vulnerabilities and limitations [105]. While some ISPs and research networks publish high-level topologies, in general the information about the topology and attributes of a large part of the Internet are privately owned and rarely published. Consequently, research in mapping the Internet mainly relies on inexact techniques which can only give a partial view of the global picture. Inference needs to be applied to the noisy and incomplete map to resolve IP addresses to routers and autonomous systems (AS), predict the existence and type of links between AS, and discover well-connected (and poorly connected) parts of the Internet.

All of the above examples require data mining and machine learning algorithms which can help to clean and improve the quality of the networks, before they are analyzed. In this chapter, we survey a subset of the inference tasks that are particularly useful in dealing with noisy and incomplete network data. We begin with some notation and then describe methods for *collective classification* (Section 4.3), *link prediction* (Section 4.4), *entity resolution* (Section 4.5), and *group detection* (Section 4.6).

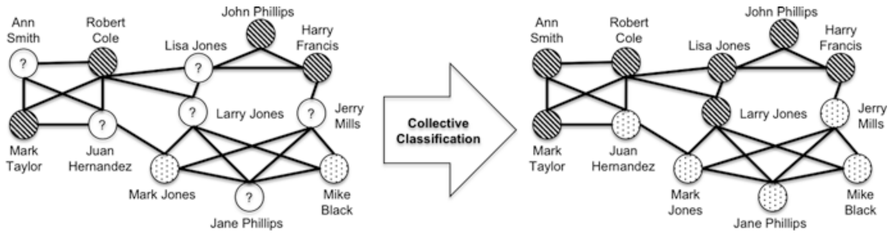

**Fig. 4.1** Example of a collective classification problem. Nodes with a *question mark* are nodes whose labels are unknown. Collective classification uses the attributes and labels of neighboring nodes. Ann Smith, for example, is likely to have the same research area as her co-authors, Robert Cole and Mark Taylor

## 4.2 Terminology and Notation

We begin by introducing some general notation and terminology used through this chapter. First, let $G(V, E)$ denote a graph $G$ with nodes $v \in V$ and edges $e \in E$. $|V|$ and $|E|$ are used to denote the size of the node and edge sets in the graph, respectively. We describe an edge and the nodes on that edge as *incident* to each other. Also, we refer to nodes which share an edge as *adjacent* to each other. For this document, whenever we use the term graph, we normally refer to either a directed graph (where each edge, $e \in E$, consists of an ordered pair of vertices) or undirected graph (where each edge, $e \in E$, consists of an unordered pair of vertices); in both cases, the edges are incident to exactly two nodes (i.e., $e = (v_i, v_j)$). In some cases, we refer to a *bipartite graph*, where the nodes can be divided into two disjoint sets, $V_1, V_2 \subset V$, so that every edge has one node in each of the two sets (i.e., $v_i \in V_i$, $v_j \in V_j$). Although we mainly use the terms graph, nodes, and edges in this chapter, we note that graphs are often used to represent networks, and the terms edges, link, and relationships are often used interchangeably.

Finally, throughout this chapter, we use a simple author collaboration network to illustrate the different inference tasks (shown in Figs. 4.1, 4.2, 4.3, and 4.4). In the collaboration network figures, the nodes represent authors and the edges between the authors indicate that the authors have co-authored at least one paper together. The shading of the nodes indicates the research area of the authors; to make it simple, here we assume there are just two areas, shown either in white (i.e., theory) or gray (i.e., systems), if observed, and as a "?" if it is unobserved. The bounded areas (as shown in Fig. 4.4) indicate group structure.

## 4.3 Collective Classification

A traditional problem in machine learning is to classify objects: given a corpus of documents classify each according to its topic label; given a collection of email communications determine which are not spam; given individuals in a collaboration network determine a characteristic of that individual; given a sentence, determine the part of speech for each word, etc. In networks, the problem of inferring labels has traditionally been applied to the nodes of the graph. Initial work in classification makes an independent and identically distributed (IID) assumption where the class label of each object is made in isolation. In graphs, however, studies have shown that predicting the labels of nodes can benefit by using autocorrelations between the node label and the attributes of related nodes. For example, in the collaboration network in Fig. 4.1, nodes with a question mark represent authors whose research areas are unknown. While we can use attributes of the author (e.g., titles of their papers) to predict the label, we can also use the research areas of the other authors they share a co-authorship edge with. The author, Ann Smith, for one is likely to work in theory given she has only co-authored with individuals in the theory field.

In the past decade there have been a number of approaches proposed which attempt to classify nodes in a joint or collective manner instead of treating each in isolation. In the following sections, we formally define the problem of collective classification and introduce several types of approaches that have been proposed to address it.

## *4.3.1 Definition*

Collective classification is an optimization problem where we are given the set of nodes, $V = \{v_1, v_2, ..., v_n\}$, over a graph $G(V, E)$, with a set of pre-defined labels, $L = \{l_1, l_2, ..., l_q\}$. Each node $v \in V$ can take exactly one value from the set of labels in $L$, denoted as $v.L$. Moreover, $V$ is divided into two sets of nodes: $V_k$, the nodes for which we know the correct labels and $V_u$, the nodes whose labels need to be determined. We are also given a neighborhood function, $N$, over the nodes where $N_i \subseteq V \setminus v_i$, which captures the relationships of a node, $v_i$. The task of collective classification is to infer the values of the labels $v.L$ for the nodes $v \in V_u$.

## *4.3.2 Approaches*

In this section, we describe the three main categories of collective classification algorithms which vary based on their mathematical underpinnings as well as how they exploit the relationships between the nodes.

### 4.3.2.1 Relational Classifiers

Traditional classification concentrates on classifying a given node using only the observed attributes of that node. Relational classifiers [104] go beyond that by also considering the observed attributes of related nodes. For instance, when classifying authors, not only would we use the words present in their papers, we would also look at the authors who they have co-authored with and their word usage and research area (if known) to arrive at the correct class label. One relational classifier, popular due to its simplicity, is the relational classifier proposed by Macskassy and Provost [73]. Their classifier makes two assumptions: some node labels are known and related nodes are likely to have the same labels. The classifier assigns a label to a node, $v_i$, by looking at the labels of related nodes whose label values are known, $N_i \cap V_u$, and taking the weighted proportion of neighbors for each possible label. The label with largest weighted proportion among neighbors is the predicted label of $v_i$. Although relational classifiers have been shown to perform well in some domains, overall the results have been mixed. For instance, although there have been reports of classification accuracy gains using such techniques over traditional classification, in certain cases, these techniques can harm classification accuracy [22].

### 4.3.2.2 Approaches Based on Local Conditional Classifiers

A source of information in collective classification is to use not only the attributes and the known labels of related nodes, but also the predicted labels of other nodes whose labels are unobserved. For instance, going back to the classification example in Fig. 4.1, authors which share a co-authorship edge to other authors *predicted* to have a certain research area, are likely to work in the same area. In this section, we look at this source of information and exploit it using local conditional classifiers. Chakrabarti et al. [22] illustrated the use of this approach and reported impressive classification accuracy gains for labeling Web pages. Neville and Jensen [81] further developed the approach as an iterative classification algorithm (ICA) and studied the conditions under which it improved classification performance [57].

We provide pseudocode for a simple variant of ICA in Algorithm 1. The basic premise behind ICA is simple. Consider a node $v_i \in V$ whose label needs to be determined. Suppose we know the attributes and labels of related nodes, $N_i$, ICA assumes that we are given a local classifier $f$ that takes the attributes and labels of the nodes in $N_i$ and returns the most likely value of $v_i.L$. This makes the local classifier $f$ an extremely flexible function and we can use popular classifiers like decision trees [95] and SVM [58] in its place. However, since $N_i$ may contain nodes whose labels we also need to predict, we need to repeat the process iteratively where in each iteration, we label each $v_i.L$ using the current best estimates of $N_i$ and classifier $f$. We continue to do so until the assignments to the labels stabilize or some stopping criterion is met.

**Algorithm 1** Iterative Classification Algorithm

```
Iterative Classification Algorithm (ICA)
  for each node $v_i \in V$ do {bootstrapping}
    {c}ompute label using only observed nodes in $N_i$
    compute $\mathbf{a}_i$ using only $V_k \cap N_i$
    $v_i.L \leftarrow f(\mathbf{a}_i)$
  end for
  repeat {iterative classification}
    generate ordering $O$ over nodes in $V_u$
    for each node $v_i \in O$ do
      {c}ompute new estimate of $v_i.L$
      compute $\mathbf{a}_i$ using current assignments to $N_i$
      $v_i.L \leftarrow f(\mathbf{a}_i)$
    end for
  until all class labels have stabilized or a threshold number of iterations have elapsed
```

A number of aspects of the iterative approach have been studied. An important aspect is how to use the values provided by $N_I$ in $f$ [70]. Most classifiers are defined as functions with a fixed-length vector of attribute values as arguments while the number of nodes in $N_i$ may vary for different $v_i$. A common approach to address this is to use an aggregation operator such as count, mode, or prop, which measures the proportion of neighbors with a given label. In Algorithm 1, we use $\mathbf{a}_i$ to denote the vector encoding the values in $N_i$ obtained after aggregation. Another aspect to consider is the choice of the local classifier $f$. Classifiers used include naive Bayes [22, 81], logistic regression [70], decision trees [57], and weighted-vote [73].

There is some evidence to indicate that discriminately trained local classifiers such as logistic regression tend to produce higher accuracies than others [101].

Previous work has also looked at different ways of ordering and updating the labels in ICA. While there is some evidence which shows ICA is fairly robust to simple ordering strategies such as random ordering, visiting nodes in ascending order of diversity of its neighborhood class labels or labeling nodes in descending order of label confidence [40], strategies which vary what labels are updated at each iteration have been shown to improve accuracies [76].

Extensions have also been proposed for the ICA algorithm. Researchers in collective classification [73, 76, 82] have extended the simple algorithm described in Algorithm 1 and developed a version of Gibbs sampling that is easy to implement and faster than traditional Gibbs sampling approaches. The basic idea behind this algorithm is to assume, just like in the case of ICA, that we have access to a local classifier $f$ that can sample for the best label estimate for $v_i.L$ given all the values for the nodes in $N_i$. We keep doing this repeatedly for a fixed number of iterations (a period known as burn-in). After that, not only do we sample for labels for each $v_i \in V_u$, but we also maintain count statistics as to how many times we sampled a give label for node $v_i$. After collecting a predefined number of such samples, we output the best label assignment for node $v_i$ by choosing the label that was assigned the maximum number of times to $v_i$ during the sampling.

#### 4.3.2.3 Approaches Based on Global Formulations

In addition to the local conditional classifier approaches discussed in Section 4.3.2.2, another approach to collective classification is to represent the problem with a high-level global graphical model and then using the learning and inference techniques for the graphical modeling approach to arrive at the correct classification. Graphical models which have been used include both directed [43] and undirected [62, 109] models. While these techniques can use both the labels and attributes of related nodes, we note that these techniques tend to be less efficient and scalable than the iterative collective classification techniques.

A common way of defining such a global model is by using a pairwise Markov random field (pairwise MRF) [109]. Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ denote a random variable graph where $\mathcal{V}$ consists of the two types of random variables: the unobserved, $\mathcal{Y}$, which need to be assigned from a label set $\mathcal{L}$, and observed variables, $\mathcal{X}$, whose labels are known. Let $\Psi$ denote a set of *clique potentials* which contain three distinct types of functions. First, for each $\mathcal{Y}_i \in \mathcal{E}$, $\psi_i \in \Psi$ is a mapping $\psi_i : L \rightarrow \mathbb{R} \geq 0$, where $\mathbb{R} \geq 0$ is the set of non-negative real numbers. Next, for each $(\mathcal{Y}_i, \mathcal{X}_j) \in \mathcal{E}$, $\psi_{ij} \in \Psi$ is a mapping $\psi_{ij} : L \rightarrow \mathbb{R} \geq 0$. The last type of function is for each $(\mathcal{Y}_i, \mathcal{Y}_j) \in \mathcal{E}$, $\psi_{ij} \in \Psi$ is a mapping $\psi_{ij} : L \times L \rightarrow \mathbb{R} \geq 0$.

Let $x$ denote the values assigned to all the observed variables in $\mathcal{G}$ and let $x_i$ denote the value assigned to $\mathcal{X}_i$. Similarly, let $y$ denote any assignment to all the unobserved variables in $\mathcal{G}$ and let $y_i$ denote a value assigned to $\mathcal{Y}_i$. For brevity of notation we will denote by $\phi_i$ the clique potential obtained by computing $\phi_i(y_i) = \psi(y_i) \prod_{(\mathcal{Y}_i, \mathcal{X}_j) \in \mathcal{E}} \psi ij(y_i)$. A pairwise MRF can then be defined as follows:

**Definition 1** A pairwise Markov random field (pairwise MRF) is given by a pair $\langle \mathcal{G}, \Psi \rangle$ where $\mathcal{G}$ is a graph and $\Psi$ is a set of clique potentials with $\phi_i$ and $\psi_{ij}$ as defined above. Given an assignment $y$ to all the unobserved variables $\mathcal{Y}$ the pairwise MRF is associated with the probability distribution:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(x)} \prod_{Y_i \in Y} \phi_i(y_i) \prod_{(Y_i, Y_j) \in E} \psi_{ij}(y_i, y_j)$$

where $\mathbf{x}$ denotes the observed values of $X$ and

$$Z(x) = \sum_{\mathbf{y}'} \prod_{Y_i \in Y} \phi_i(y'_i) \prod_{(Y_i, Y_j) \in E} \psi_{ij}(y'_i, y'_j).$$

Given a pairwise MRF, it is conceptually simple to extract the best assignments to each unobserved variable in the network. For instance, we may adopt the criterion that the best label value for $\mathcal{Y}_i$ is simply the one corresponding to the highest marginal probability obtained by summing over all other variables from the probability distribution associated with the pairwise MRF. Computationally, however, this is difficult to achieve since computing one marginal probability requires summing over an exponentially large number of terms. Hence, approximate inference algorithms are typically employed, the two most common being loopy belief propagation (LBP) and mean-field relaxation labeling. A comparison of these two approaches are given in [90, 101].

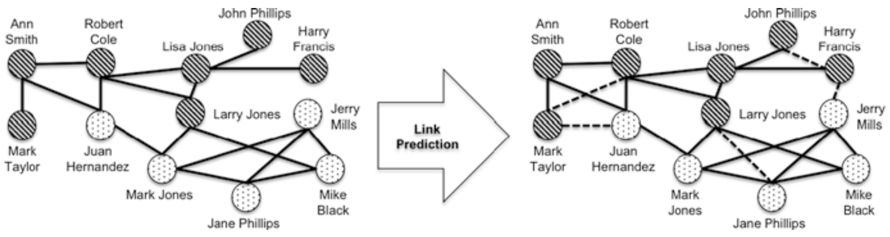

**Fig. 4.2** Example of a link prediction problem. The graph on the *left* represents a collaboration network at time $t$, and the graph on the *right* represents the predicted collaboration network at time $t + 1$. Predicted collaboration edges are highlighted using a *dashed line*

## 4.4 Link Prediction

In this section, we change our focus from inferring information about the nodes of a network to inferring information about the links or edges between them. Inferring the existences of edges between nodes has traditionally been referred to as *link* *prediction* [69, 110]. We provide a formal definition of the problem of link prediction, as well as discuss variants and closely related problems in Section 4.4.1.

Link prediction is a challenging problem that has been studied in various guises in different domains. For example, in social network analysis, there is work on predicting friendship links [119], event participation links (i.e., co-authorship [89]), communication links (i.e., email [89]), and links representing semantic relationships (i.e., advisor of [110] and subordinate manager [30]). In bioinformatics, there is interest in predicting the existence of edges representing physical protein–protein interactions [50, 107, 118], domain–domain interactions [29], and regulatory interactions [4]. Similarly, in computer network systems there is work in inferring unobserved connections between routers, and inferring relationships between autonomous systems and service providers [105]. There is also work on using link prediction to improve recommender systems [36, 51], Web site navigation [120], surveillance [52], and automatic document cross-referencing [77].

## 4.4.1 Definition

We begin with some basic definitions and notation. We refer to the set of possible edges in a graph as *potential edges*. The set of potential edges depends on the graph type and how the edges for the graph are defined. For example, in a directed graph, the set of potential edges consists of all edges $e = (v_1, v_2)$ where $v_1$ and $v_2$ are any two nodes $V$ in the graph (i.e., the number of potential edges is $|V| \times |V|$). In an undirected bipartite graph with two subsets of nodes ($V_1, V_2 \in V$), while the edges still consist of a pair of nodes, $e = (v_1, v_2)$, there is an added condition such that one node must be from $V_1$ and the other node must be from $V_2$; this results in $|V_1| \times |V_2|$ potential edges. Next, we refer to set of "true" edges in a graph as *positive edges*, and we refer to the "true" non-edges in a graph (i.e., pairs of nodes without edges between them) as *negative edges*. For a given graph, typically we only have information about a subset of the edges; we refer to this set as the *observed* edges. The observed edges can include both positive and negative edges, though in many formulations there is an assumption of positive-only information. We can view link prediction as a probabilistic inference problem, where the evidence includes the observed edges, the attribute values of the nodes involved in the potential edge, and possibly other information about the network, and for any unobserved, potential edge, we want to compute the probability of it's existing. This can be reframed as a binary classification problem by choosing some probability threshold and concluding that potential edges with existence probability above the threshold are true edges, and those below the threshold are considered false edges (more complex schemes are possible as well).

The earliest and most cited formulation of the link prediction problem was proposed by Liben-Nowell and Kleinberg [69]. Liben-Nowell and Kleinberg [69] proposed a temporal formulation defined over a dynamic network where given a graph $G_t(V_t, E_t)$ at time $t$, infer the set of edges at the next time step $t + 1$. More formally, the objective is to infer a set of edges $E_{\text{new}}$ where $E_{t+1} = E_t \bigcup E_{\text{new}}$. In this chapter, we use a more general definition of link prediction proposed by Taskar et al. [110] where given a graph $G$ and the set of potential edges in $G$, denoted $P(G)$, the problem of link prediction is to predict for all $p \in P(G)$ whether $p$ exists or does not exists, remaining agnostic on whether $G$ is a noisy graph with missing edges or a snapshot of a dynamic graph at a particular time point.

In addition to the definition of link prediction discussed above, it is also important to mention four closely related problems: *random graph models*, *link completion*, *leak detection*, and *anomalous link discovery*, whose objectives are different but very similar to link prediction. The first related research area, random graph models, is the problem of defining models for generating random graphs which capture the properties of graphs found in real networks [11, 33, 65, 66, 86]. Properties include scale-free degree distributions [1, 11, 35], the small-world phenomenon [11, 114], and densification and shrinking diameters of dynamic networks over time [66]. An important aspect of these models is modeling how to randomly generate edges between the nodes of the graph to capture these properties. The preferential attachment model [11], for example, creates edges based on the degree of nodes (i.e., higher degree nodes are more likely to be incident to more edges). The Forest Fire model [66], on the other hand, generates edges for nodes in an epidemic fashion, growing outward from some initial set of neighboring nodes.

The next two related problems, link completion [10, 21, 45] and leak detection [10, 20, 60], are a variation of link prediction over hypergraphs. A hypergraph is a graph where the edges (known as hyperedges) can connect any number of nodes. For example, in a hypergraph representing an email communication networks, a hyperedge may connect nodes representing email addresses that are recipients of a particular email communication. In link completion, given the set of nodes that participate in a particular hyperedge, the objective is to infer nodes that are missing. For our email communication network example, link completion may involve inferring which email address nodes need to be added to the hyperedge representing the recipients list of an email communication. Conversely, in leak detection, given the set of nodes participating in a particular hyperedge, the objective is to infer which nodes should not be part of that hyperedge. For example, in email communications, leak detection will attempt to infer which email address nodes are incorrectly part of the hyperedge representing the recipient list of the email communication.

The last problem, anomalous link discovery [53, 96], has been proposed as an alternate task to link prediction where the existence of the edges are assumed to be observed, and the objective is to infer which of the observed links are anomalous or unusual. Specifically, anomalous link discovery identifies which links are statistically improbable with the idea that these may be of interest for those analyzing the network. Rattigan and Jensen [96] show that some methods which perform poorly for link prediction can still perform well for anomalous link discovery.

### *4.4.2 Approach*

In this section, we discuss the two general categories of the current link prediction models: topology-based approaches and node attribute-based approaches. Topology-based approaches are methods which rely solely on the topology of the network to infer edges. Node attribute-based approaches make predictions based on the attribute values of the nodes incident to the edges. In addition, there are models which make use of both structure and attribute values.

#### 4.4.2.1 Topology-Based Approaches

A number of link prediction models have been proposed which rely solely on the topology of the network. These models typically rely on some notion of structural proximity, where nodes which are close are likely to share an edge (e.g., sharing common neighbors, nodes with a small shortest path distance between). The earliest topological approach for link prediction was proposed by [69]. In this work, Liben-Nowell and Kleinberg proposed various structure-based similarity scores and applied them over the unobserved edges of an undirected graph. They then use a threshold $k$ and only predict edges with the top $k$ scores as existing. A variety of similarity scores were proposed, given two nodes $v_1$ and $v_2$, including graph distance (the negated shortest path between $v_1$ and $v_2$), common neighbors (the size of the intersection of the sets of neighbors of $v_1$ and $v_2$), and more complex measures such as the Katz measure (the sum of the lengths of the paths between $v_1$ and $v_2$ exponentially damped by length to count short paths more heavily). Evaluating over a co-authorship network, the best performing proximity score measure was the Katz measure; however, the simple measures, which rely only on the intersection of the set of nodes adjacent to both nodes, performed surprisingly well. A related approach was proposed by [118] which applies the link prediction problem to predicting missing protein–protein interactions (PPI) from PPI networks generated by high-throughput methods. This work assumes that interacting proteins tend to form a clique. Thus, missing edges can be predicted by predicting the existence of edges which will create cliques in the network. More recent work by [24] has tried to go beyond predicting edges between neighboring nodes. In their problem domain of food webs, for example, pairs of predators often prey on a shared prey species but rarely prey on each other. Thus, in these networks, predicting "predator–prey" edges need to go beyond proximity. For this, they propose a "hierarchical random graph" approach which fits a hierarchical model to all possible dendrograms of a given network. The model is then used to calculate the likelihood of an edge existing in the network.

#### 4.4.2.2 Node Attribute-Based Approaches

Although topology has been shown useful in link prediction, topology-based approaches ignore an important source of information in networks, the attributes of nodes. Often there are correlations in the attributes of nodes which share an edge with each other. One approach which exploits this correlation was proposed by Taskar et al. [110]. In this approach, a relational Markov network (RMN) framework was applied to predicting the existence and class of edges between Web sites. They exploit the fact that certain links can only exist between nodes of the appropriate type. For example, an “advisor” edge can only exist between a student and a faculty nodes. Another approach which uses node attributes was proposed by [94]. In that approach, they used a structured logistic regression model over learned relational features to predict citation edges in a citation network. Their relational features are built over attributes such as the words used in the paper nodes. O’Madadhain et al. [89] also proposed an attribute based approach, constructing local conditional probability models based on the attributes such as node attribute similarity, topic distribution, and geographical location in predicting “co-participation” edges in an email communication network. More recently, there is work on exploiting other node attributes like the group membership of the nodes. Zheleva et al. [119] showed that membership in family groups are very useful in predicting friendship links in social networks. Similarly, [106] showed that using protein complex information can be useful in predicting protein–protein interactions. Finally, we note that in link prediction, as in classification, the quality of predictions can be improved by making the predictions collectively. Aside from the relational Markov network approach by [110] mentioned earlier, Markov Logic networks [98] and Probabilistic Relational models [42] have also been proposed for link prediction and are capable of performing joint inference.

### *4.4.3 Issues*

There are a number of challenges which make link prediction very difficult. The most difficult challenge is the large class skew between the number of edges which exist and the number of edges which do not. To illustrate, consider directed graph denoted by $G(V, E)$. While the number of edges $|E|$ is often $O(|V|)$, the number of edges which do not exist is often $O(|V|^2)$. Consequently, the prior probability edge existence is very small. This causes many supervised models, which naively optimize for accuracy, to learn a trivial model which always predicts that a link does not exist. A related problem in link prediction is the large number of edges whose existence must be considered. The number of potential edges is $O(|V|^2)$ and this limits the size of the data sets which can be considered.

In practice, there are general approaches to addressing these issues either prior to or during the link prediction. With both large class skew and number of edges to contend with, the general approach is to make assumptions which reduce the number of edges to consider. One common way to do this is to partition the set of nodes where we only consider potential edges between nodes of the same partition; edges between partitions are not explicitly modeled and are assumed not to exist [2, 118]. This is useful in many domains where there is some sort of natural partition among the nodes available (e.g., geography in social networks, location of proteins in a cell) which make edges across partitions unlikely. Another way is to define some simple, computationally inexpensive distance measure such that only edges whose nodes are within some distance are considered [30, 69].

Another practical issue in link prediction is that while real-world data often indicates which edges exist (positive examples), the edges which do not exist (negative examples) are rarely annotated for use by link prediction models. In bioinformatics, for example, the protein–protein interaction network of yeast, the most and annotated studied organism, is annotated with thousands of observed edges (physical interactions) between the nodes (proteins) gathered from numerous experiments [13]. There are currently, however, no major data sets available which indicate which proteins definitely do not physically interact. This is an issue not only in creating and learning models for link prediction but also an issue with evaluating them. Often, it is unclear whether a predicted edge which is not in our ground truth data is an incorrectly predicted edge or an edge resulting from incomplete data.

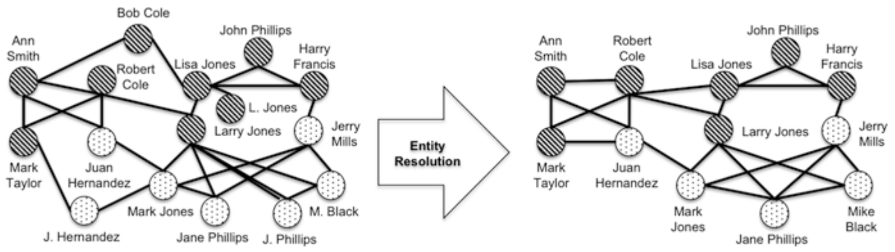

**Fig. 4.3** Example of a entity resolution problem. In this example, the nodes on the *left* are ambiguous due to variations in the spelling of their names. While attributes may suffice to resolve the entities in some cases (e.g., Juan Hernandez and J. Hernandez are likely the same person due to the similarity in their names), some cases (e.g., J. Phillips can refer to either Jane or John Phillips) it may not. However, if we use the edges (i.e., both Jane Phillips and J. Phillips have collaborated with Larry Jones), we are able to improve our predictions

## 4.5 Entity Resolution

Many networks have uncertain and imprecise references to real-world entities. The absence of identifiers for the underlying entities often results in noisy networks which contain multiple references to the same underlying entity. In this section, we look at the problem of resolving which references refer to the same entity, a problem known as *entity resolution*.

Examples of entity resolution problems can be found in many domains, often under different names. The earliest applications of entity resolution is on medical data [37, 83, 84, 117]. In this work, in a problem they referred to as record linkage, the goal was to identify which medical records refer to the same individual or family. Later, in computer vision, entity resolution was applied in identifying which regions in the same image are part of the same object (the correspondence problem). Also, in natural language processing, there is interest in determining which noun phrases refer to the same underlying entity (coreference resolution, object consolidation). The problems of deduplication and data integration, determining when two tuples in or across databases refer to the same entity, can also be seen as entity resolution.

### 4.5.1 Definition

We begin by introducing some additional notation. For a graph $G(V, E)$ we are given a set of reference nodes $R \subseteq V$ where the reference nodes correspond to some set of unknown entity nodes $E$. We introduce the notation $r.E$ to refer to the entity to which $r$ corresponds. Formally, the general goal of entity resolution is to recover the hidden set of entities $E$ and the entity labels $r.E$ for all the reference nodes.

We note that there are two commonly used interpretations of entity resolution and which is more natural depends on the algorithm chosen. First, entity resolution can be viewed as a pairwise classification problem, where for each pair of references, $r_i, r_j \in R$, we are interested in determining whether $r_i$ and $r_j$ are co-referent (i.e., $r_i.E = r_j.E$). Note the similarity here with link prediction; in fact, many of the challenges of link prediction (class skew and scaling) are issues in entity resolution as well. The second view is as a clustering problem, where the goal is to assign the reference nodes to clusters $C \in \mathcal{C}$. The subset of reference nodes in each cluster are assumed to be co-referent to each other (i.e., $\forall r_i, r_j \in C,\ r_i.E = r_j.E$).

### 4.5.2 Approach

In this section, we survey existing entity resolution approaches. We distinguish between three categories of approaches: attribute-based, naive relational, and collective relational. Attribute-based approaches are the traditional approaches to entity resolution which rely solely on the attributes of the reference nodes. More recently, naive and collective relational approaches have been proposed which take the edges between these nodes into consideration. The naive relational approaches consider the attribute similarity of related reference node. The collective relational approaches, on the other hand, use the edges to make decisions jointly.

#### 4.5.2.1 Attribute-Based Entity Resolution

The attribute-based approach to entity resolution typically uses the pairwise formulation of the entity resolution problem [26, 37, 48]. Given two reference nodes, $r_i, r_j \in R$, the attribute-based approaches generally make use of a similarity measure, $sim_A(r_i, r_j)$, or a weighted combination of multiple similarity measures, over the attributes of the reference nodes. Several sophisticated similarity measures have been proposed for use in entity resolution based on the types of features and domain knowledge. For example, there are string similarity measures used commonly over the names of an entity such as

- Jaccard [54]: the size of the intersection among the characters divided by the size of the union of the characters occurring.
- Jaro and Jaro-Winker [56, 117]: string similarity scores which attempt to take into account typical spelling deviation by looking at the similarity within a certain neighborhood of the string characters; the Jaro-Winkler score is based on Jaro and weights matches at the beginning more highly.
- Levenshtein (edit distance) [67]: the minimum number of insertions, deletions, and substitutions required to transform one string to the other.
- Monge-Elkan [78]: recursive subcomponent matching algorithm which looks at matching subcomponents of the strings; it is good at finding swapped fields, such as first and last names.

Approaches have also been proposed which learn a string similarity measure from labeled data [18]. Pairs of nodes whose similarity is above a certain threshold are predicted as co-referent. Transitivity may also be enforced such that if $r_i$ and $r_j$ are predicted co-referent and $r_j$ and $r_k$ predicted co-referent, $r_i$ and $r_k$ are also predicted co-referent.

#### 4.5.2.2 Naive Relational Entity Resolution

While attribute-based approaches have been shown to do well in some domains, work in relational data has focused on incorporating links, in particular, co-occurrence information. The earliest work using links for entity resolution was explored in the database community. Ananthakrishna et al. [6] introduce a method for deduplication using edges in data warehouse applications where there is a dimensional hierarchy over the link relations. Kalashnikov et al. [59] proposed the Relationship-based Data Cleaning (RelDC) approach which uses graph theoretic techniques to discover and analyze relationships, such as affiliation and co-authorship, that exist between reference nodes.

#### 4.5.2.3 Collective Relational Entity Resolution

Although the approaches in Section 4.5.2.2 consider the edges for entity resolution, only the attributes of linked references are considered and the different resolution decisions are still taken independently. Work in collective relational entity resolution addresses this by using the edges between nodes to establish dependencies in the resolution decisions. In databases, for example, approaches have been proposed [14, 32] where one resolution decision affects another if they are linked. Bhattacharya and Getoor [14, 17] propose different measures for edge similarity and show how those can be combined with attribute similarity iteratively to perform entity resolution on collaboration networks. Dong et al. [32] collectively resolve entities of multiple types by propagating evidence along the links in a dependency graph. In machine learning, probabilistic models have also been proposed to consider the interactions between the different entity resolution decisions. McCallum and Wellner [75] use conditional random fields for noun coreference and use clique templates with tied parameters to capture repeated relational structure. Singla and Domingos [103] use the idea of merging evidence to allow the flow of reasoning between different pairwise decisions over multiple entity types. Markov logic networks have also been applied for collective entity resolution [93, 103]. Pasula et al. [92] propose a generic probabilistic relational model framework for performing entity resolution on citations. Li et al. [68] propose a probabilistic generative model which captures a joint distribution over pairs of entities in terms of co-mentions in documents. Similarly, Bhattacharya and Getoor [16] proposed a generative group model by extending the Latent Dirichlet Allocation model for documents and topics.

### *4.5.3 Issues*

A major issue in entity resolution is that it is a known hard problem computationally; a naive algorithm is $O(N^2)$, which for very large data sets is not feasible. For many networks, it is infeasible to compare all pairs of references for approaches which use expensive similarity measures. Similarly, for many probabilistic models, it is infeasible to explicitly represent all the variables required for the inference. Thus, efficiencies have long been a focus for research in entity resolution. One mechanism for doing this involves computing the matches efficiently and employing techniques commonly called "blocking" to place nodes into disjoint "blocks" using cheap and index-based similarity computations [49, 79]. The number of potential pairs is greatly reduced by assuming that only pairs of nodes in the same block can be co-referent pairs. Another mechanism, proposed by McCallum et al. [74], relaxes the use of disjoint blocks and places nodes into possibly overlapping subsets called "canopies". Potential co-referent pairs are then restricted only to pairs of nodes which share at least one common canopy.

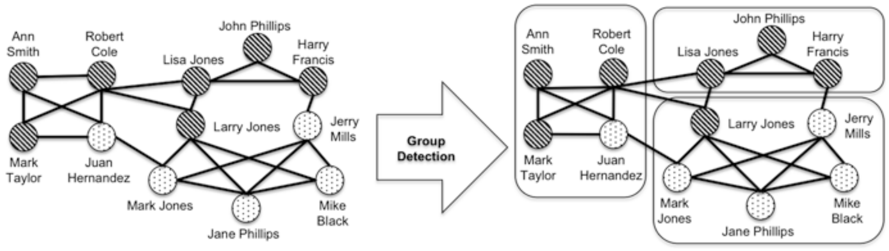

**Fig. 4.4** Example of a group detection problem. The goal of group detection is to predict the underlying groups which the nodes, and/or edges, participate in. The three regions surrounded with a *rounded rectangle* represent the affiliations of our authors

Another issue in entity resolution is referred to a “canonicalization” [27, 116]. Once the reference nodes have been resolved to their corresponding entities, there is the problem of constructing a standard representation of the entity from the attributes of those references. In particular, canonicalization resolves the inconsistencies in the attributes among the reference nodes. Simple heuristics for determining the appropriate values for the attributes and edges of an entity based on the attributes of the references are possible; often these amount to choosing the longest string, or the most recently updated value. Such approaches, however, are not robust to noisy and incomplete attributes. Another approach is, instead of returning a single value for an attribute, keeping all the values, returning a ranked list of the possible values and edges [7, 111]. When there are a large number of references, however, the ranked list may be too long. Culotta et al. [27] addresses this by using adaptive similarity measures to select values in order to create a standard representation most similar to each of the different records. A unified approach was also proposed by Wick et al. [116] which performs entity resolution and canonicalization jointly using discriminatively trained model.

## 4.6 Group Detection

Another common problem that often occurs in reasoning about network data is inferring the underlying hidden groups or community structures of the nodes in the network. This problem is strongly related to data clustering; a traditional unsupervised learning problem in data mining. In cluster analysis, data points are organized in different groups based on the similarity of their feature values [55], where points in the same cluster are more similar to each other than points in different clusters according to a specific similarity measure. Similarly, a community in a network can be defined as a group of nodes that share dense connections among each other, while being less tightly connected to nodes in different communities in the network.

The importance of identifying the communities in networks lies in the fact that they can often be closely related to functional units of the system, e.g., groups of individuals interacting with each other in a society [8, 44, 71], WWW pages related to similar topics [38], compartments in food webs [61], or proteins responsible for similar biological functions [23]. Furthermore, analyzing the community structure itself provides insight into understanding the various roles of different nodes in their corresponding groups. For instance, by studying the structural properties of communities, one can distinguish between the functions of the central nodes in the group and the ones at the periphery.

In this section, we review some of basic methods for group detection and community discovery in network settings.

### *4.6.1 Definition*

As before, we consider a graph $G = (V, E)$; in the case of weighted networks, $w(v_i, v_j)$ denotes the weight of the edge connecting nodes $v_i$ and $v_j$. A community or a group $C$ is a subgraph $C(V', E')$ of the original graph $G(V, E)$ whose nodes and edges are subsets of the original graph's nodes and edges, i.e., $V' \subset V$ and $E' \subset E$. For each node $v'$ in group $C$ of $G$, we define an *internal* and an *external* degree as $d_{\text{int}}(v') = |e'(v', v_t)|$; $v_t \in V'$ and $d_{\text{ext}}(v') = |e'(v', v_t)|$; $v_t \notin V'$, where the internal degree of a node with respect to a certain group is the number of edges connecting it to other nodes of the group, while its external degree is the number of edges connecting it to nodes in the graph other than those in the corresponding group. Intuitively, nodes with relatively high internal degree and low external degree for a specific group are potentially good candidates to be included in that group. The opposite is also true, where nodes with low internal degree and high external degree for a specific group are candidates for removal. Throughout the discussion, the terms group, community, and cluster are used exchangeably.

To identify communities in networks, a basic set of properties that is capable of distinguishing a true community structure from a randomly selected set of nodes and edges is needed. One of the important properties that can be utilized is the graph density, which is the number of edges present in the network relative to the total number possible. Similarly, the density of a group of nodes in the network can be defined as the ratio between the number of edges connecting pairs of nodes within that group and the maximum number of possible edges within the same group:

$$\delta(C) = \frac{|E'|}{|V'| \times (|V'| - 1)/2}. \tag{4.1}$$

A randomly selected set of nodes from a network is likely to have a density similar to that of the global network structure. However, for community structures, the density of a group is expected to be higher than that of the overall graph. Formally, for any community $C$ in a graph $G$, it is expected that $\delta(C) > \delta(G)$, where $\delta(G)$ is the overall graph density. Similarly, the average density of sets of nodes belonging to different communities, calculated using the ratio between the number of edges emanating from a group and terminating in another, and the maximum number possible of such edges, should generally be low. This basic idea is exploited in many of the group detection methods described next.

### *4.6.2 Approaches*

Beyond the intuitive definition above, precisely defining what constitutes a community involves a number of aspects: whether the definition relies on global or local network properties, whether nodes can simultaneously belong to several communities, whether link weights are utilized, and whether the definition allows for hierarchical community structure. Global methods utilize the whole network structure for defining the communities. This can be achieved in several ways, such as global optimization methods [87, 97], algorithms based on different global centrality measures [39, 44], spectral methods [9, 31], or information-theoretic methods

[99, 100]. Local methods, on the other hand, define communities based on purely local network structure, such as detecting cliques of different sizes [34], clique percolation method [91], and subgraph fitness method [63].

As mentioned above, another important aspect is whether nodes are allowed to belong simultaneously to several communities. In general, overlapping communities do commonly occur in natural settings, especially in social networks. Currently, only a few methods are able to handle overlapping communities [88, 91]. Another difficulty in community detection is that networks may contain hierarchical structures, which means that communities may be parts of even larger communities. This leads to the problem of evaluating the best partitioning among different alternatives. One solution for evaluating the quality of a given community structure was suggested by Girvan and Newman [87], who introduced the concept of modularity as a measure for the goodness of a partitioning.

The methods used for community detection with respect to different perspectives are briefly reviewed in the following sections.

### 4.6.2.1 Clique-Finding Techniques

Cliques are graph structures that are frequently used in local techniques for community detection. A clique is defined as a complete subgraph $\{C(V', E') : \forall v_1, v_2 \in V', \exists (v_1, v_2) \in E'\}$, where there exists an edge between every pair of nodes belonging to it. In this context, communities can be considered as maximal clique, which cannot be extended with the addition of any new nodes or edges.

One of the problems of using this approach for group detection is the fact that finding cliques in a graph is an NP-complete problem. Another problem arises from the interpretation of communities, especially in social networks, where we expect different individuals to have different centrality in their corresponding groups, contradicting with the degree symmetry of nodes in cliques. To overcome these drawbacks, the notion of cliques is often relaxed to $k$-clique, which is a maximal subgraph where the distance between each pair of its nodes is not larger than $k$ [3].

Recently, Palla et al. [91] introduced a local method for community detection called the clique percolation method. The method is based on the observation that, due to the high density of community structures, it is more likely that nodes within a given community form more small-sized cliques than nodes belonging to different communities. The clique percolation algorithm defines communities by considering overlapping chains of small cliques, which are likely to explore a significant fraction of each community, without crossing the boundary between different communities. Specifically, a community of size $k$ is obtained by "rolling" a clique of size $k$ over cliques of the same size that share at least $k - 1$ nodes with the current clique.

### 4.6.2.2 Clustering Techniques

Data clustering is one of the earliest techniques for group detection, where data points are grouped according to a specific similarity measure over their features.

The main objective of traditional clustering methods is to obtain clusters or groups of data points possessing high intra-cluster similarity and low inter-cluster similarity. Classical data clustering techniques can be divided into partition-based methods such as $k$-means clustering [72], model-based methods such as *Expectation-Maximization* algorithm [28], spectral clustering algorithms [5, 115] and hierarchical clustering methods [47] which are very popular and commonly used in many fields.

One advantage of the hierarchical clustering techniques is that they provide the ability to look at the groups at multiple resolutions. Hierarchical techniques are further divided into agglomerative and divisive algorithms. The agglomerative algorithm is a greedy bottom-up algorithm which starts with individual data points, then successively merge pairs with highest similarity. At each iteration, the similarities between the new cluster and each of the old clusters are recomputed and again the maximally similar pair of clusters merged. Divisive algorithms work in a reverse manner, where initially the whole set of points is regarded as one cluster which is successively divided into smaller ones by splitting nodes of lowest similarity. In both algorithms, clusters are represented as a dendrogram (see Fig. 4.5), whose depths indicate the steps at which two clusters are joined. This representation provides insight into the formed groups, where it is clear which communities are built up from smaller modules, and how these smaller communities are organized.

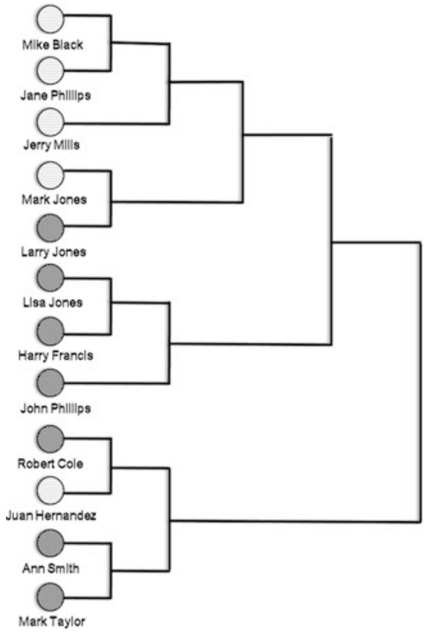

**Fig. 4.5** A dendrogram resulting from a hierarchical clustering technique. Different levels in the tree correspond to partitions of the graph into clusters

Hierarchical clustering techniques can easily be adapted to network domains, where data points are replaced by individual nodes in the network, and the similarity is based on edges between them. In addition, there are other divisive algorithms based on spectral methods and other community detection techniques, which are discussed in the following sections.

#### 4.6.2.3 Centrality-Based Techniques

Girvan and Newman introduced several community detection algorithms that have received much attention. The first method [44] uses a divisive algorithm based on the betweenness centrality of edges to be able to recover the group structure within the network. Betweenness centrality is a measure of centrality of nodes in networks, defined for each node as the number of shortest paths between pairs of nodes in the network that run through it. The Girvan–Newman algorithm extended this definition for edges in the network as well, where the betweenness centrality of an edge is defined as the number of shortest paths between pairs of nodes that include this edge.

The algorithm is also based on the fact that there exists denser connections between nodes belonging to the same group structure than those in different groups. Thus, all shortest paths between nodes from different communities should pass along one of these sparse set of edges, increasing their edge betweenness centrality measure. By following a divisive approach and removing edges with highest betweenness centrality from the network successively, the underlying community structure is revealed.

One of the drawbacks of the algorithm is its time complexity which is $O(|E|^2|V|)$ generally, and $O(|V|^3)$ for sparse networks. However, by limiting the re-calculations of the edge betweenness for only those affected by the prior edge removal can be factored in, making the algorithm efficient in sparse networks with strong community structure, but still not very efficient on dense networks. Following the same approach, other methods based on different notions of centrality have been introduced [64, 112].

#### 4.6.2.4 Modularity-Based Techniques

The concept of modularity was introduced by Newman and Girvan [87] as a measure to evaluate the quality of a set of extracted communities in a network and has become one of the most popular quality functions used for community detection. The basic idea is utilizing a *null model*; a randomly rewired version of the original network preserving the node degrees, which is expected to contain no community structure. Modularity is then calculated by comparing the number of edges within the extracted communities against the expected number of edges in the same communities from the random network. More specifically, the modularity $Q$ is defined as follows:

$$Q = \frac{1}{2|E|} \sum_{ij} \left[ A_{ij} - \frac{k_i.k_j}{2|E|} \right] \delta(c_i, c_j), \tag{4.2}$$

where $A_{ij}$ is the element of the adjacency matrix of the network denoting the number of edges between nodes $i$ and $j$, $k_i$ and $k_j$ are the degrees of nodes $i$ and $j$ respectively, $c_i$ and $c_j$ are the communities to which nodes $i$ and $j$ belong respectively. The summation runs over all pairs of nodes within the same community.

Clearly, a higher modularity value indicates that the average density of the extracted community is larger than that of the random network where no community structure is present. Thus, modularity maximization can be used as the objective for producing high-quality community structure. However, modularity maximization is an NP-hard problem [19]. Nevertheless, there has been several heuristics for approximate modularity maximization with reasonable time complexity.

An efficient greedy modularity maximization algorithm was introduced by Newman [85]. The algorithm starts with individual nodes and merges them agglomeratively, by choosing the pair that gives the largest increase in modularity. The time complexity of this greedy algorithm is $O(|V|(|E| + |V|))$ or $O(|V|^2)$ for sparse networks, which enables users to run community detection on large networks in a reasonable amount of time. A further speedup was achieved by Clauset et al. [25] by utilizing specialized data structures for sparse matrices.

### *4.6.3 Issues*

Because the majority of work on group detection in relational setting has focused on the structural properties of the nodes and the edges in the underlying network, the resulting communities often lack a correspondence with the actual functional communities in the network [102]. Recently, relational clustering methods have been introduced for combining structural information with node characteristics to obtain better communities that are more related to the functional units in the network [15, 80]. However, more work is needed for tying the information about the target function with the group detection process to obtain different community structures from the network according to the specific function that needs to be highlighted.

One of the issues that has attracted more attention lately is the fact that most group detection methods works on single-mode networks, with less work focused on finding groups in more complex, multi-mode settings [12, 46]. Most algorithms deal with these types of networks by projecting them onto a series of individual graphs for each mode, thus losing some of the information that could have been retained by operating collectively on the original multi-modal setting.

Another issue that is gaining more interest is developing new methods for group detection in dynamic network settings [108], where the underlying network structure changes over time. Most of the previous work on group detection mainly focused on static networks, and handles the dynamic case by either analyzing a snapshot of the network at a single point in time, or aggregating all interactions over the whole time period. Both approaches do not capture the dynamics of change in the network structure, which can be an important factor in revealing the underlying communities.

## 4.7 Conclusion

In this chapter, we have surveyed some of the common inference tasks that can be applied to graph data. The algorithms we have presented are especially well suited to the situation where we have noisy and incomplete observations. Some of the methods focus on predicting attribute values, some focus on inferring the existence of edges, and some focus on grouping nodes, either for entity resolution or for community detection. There are many other possibilities and combinations still to be explored, and this research area is likely to expand as we gather more and more graph and network data from a wider variety of sources.

**Acknowledgments** The work was supported by NSF Grant #0746930.

## References

1. J. Abello, A. L. Buchsbaum, and J. R. Westbrook. A functional approach to external graph algorithms. In *Proceedings of the 6th Annual European Symposium on Algorithms*, Venice, Italy, 1998.
2. S. F. Adafre and M. de Rijke. Discovering missing links in wikipedia. In *Proceedings of the 3rd International Workshop on Link Discovery*, Chicago, IL, 2005.
3. R. D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3:113–126, 1973.
4. R. Albert, B. DasGupta, R. Dondi, S. Kachalo, E. Sontag, A. Zelikovsky, and K. Westbrook. A novel method for signal transduction network inference from indirect experimental evidence. *Journal of Computational Biology*, 14:407–419, 2007.
5. C. Alpert, A. Kahng, and S. Yao. Spectral partitioning: The more eigenvectors, the better. *Discrete Applied Math*, 90:3–26, 1999.
6. R. Ananthakrishna, S. Chaudhuri, and V. Ganti. Eliminating fuzzy duplicates in data warehouses. In *Proceedings of the 28th International Conference on Very Large Databases*, Hong Kong, China, 2002.
7. P. Andritsos, A. Fuxman, and R. J. Miller. Clean answers over dirty databases: A probabilistic approach. In *Proceedings of the 22nd International Conference on Data Engineering*, Hong Kong, China, 2006.
8. A. Arenas, L. Danon, A. Daz-Guilera, P. M. Gleiser, and R. Guimer. Community analysis in social networks. *The European Physical Journal B*, 38(2):373–380, 2004.
9. A. Arenas, A. Daz-Guilera, and C. J. Prez-Vicente. Synchronization reveals topological scales in complex networks. *Physical Review Letters*, 96(11):114102, 2006.
10. R. Balasubramanyan, V. R. Carvalho, and W. Cohen. Cutonce- recipient recommendation and leak detection in action. In *Workshop on Enhanced Messaging*, Chicago, IL, 2009.
11. A.-L. Barabasi and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286(5439):509–512, 1999.
12. J. Barber. Modularity and community detection in bipartite networks. *Physical Review E*, 76:066102, 2007.

13. A. Ben-Hur and W. Noble. Choosing negative examples for the prediction of protein-protein interactions. *BMC Bioinformatics*, 7:S2, 2006.
14. I. Bhattacharya and L. Getoor. Iterative record linkage for cleaning and integration. In *Data Mining and Knowledge Discovery*, Paris, France, 2004.
15. I. Bhattacharya and L. Getoor. Relational clustering for multi-type entity resolution. In *ACM SIGKDD Workshop on Multi Relational Data Mining*, Chicago, Illinois, 2005.
16. I. Bhattacharya and L. Getoor. A latent dirichlet model for unsupervised entity resolution. In *SIAM Conference on Data Mining*, Bethesda, MD 2006.
17. I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data*, 1:1–36, 2007.
18. M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, D.C., 2003.
19. U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. N. Z, and D. Wagner. On finding graph clusterings with maximum modularity. In *Proceedings of 33rd International Workshop on Graph-Theoretical Concepts in Computer Science*, Dornburg, Germany, 2007.
20. V. R. Carvalho and W. W. Cohen. Preventing information leaks in email. In *SIAM Conference on Data Mining*, Minneapolis, MN, 2007.
21. P. Chaiwanarom and C. Lursinsap. Link completion using prediction by partial matching. In *International Symposium on Communications and Information Technologies*, Vientiane, Lao, 2008.
22. S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *ACM SIGMOD International Conference on Management of Data*, Seattle, WA, 1998.
23. J. Chen and B. Yuan. Detecting functional modules in the yeast protein-protein interaction network. *Bioinformatics*, 22(18):2283–2290, 2006.
24. A. Clauset, C. Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453:98, 2008.
25. A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review*, 70(6):066111, 2004.
26. W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *Proceedings of the International Joint Conference on Artificial Intelligence Workshop on Information Integration*, Acapulco, Mexico, 2003.
27. A. Culotta, M. Wick, R. Hall, M. Marzilli, and A. McCallum. Canonicalization of database records using adaptive similarity measures. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Jose, CA, 2007.
28. A. P. Dempster, N. M. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society Series B*, 39(1):1 – 38, 1977.
29. M. Deng, S. Mehta, F. Sun, and T. Chen. Inferring domain-domain interactions from protein-protein interactions. *Genome Research*, 12(10):1540–1548, October 2002.
30. C. Diehl, G. M. Namata, and L. Getoor. Relationship identification for social network discovery. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, Vancouver, Canada, 2007.
31. L. Donetti and M. A. Muoz. Detecting network communities: A new systematic and efficient algorithm. *Journal of Statistical Mechanics*, 10:10012, 2004.
32. X. Dong, A. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Baltimore, MD, 2005.
33. P. Erdos and A. Renyi. On the evolution of random graphs. *Mathematics Institute Hungarian Academy of Science*, 5:17–61, 1960.
34. M. G. Everett and S. P. Borgatti. Analyzing clique overlap. *Connections*, 21(1):49–61, 1998.
35. M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, Cambridge, MA, 1999.
36. S. Farrell, C. Campbell, and S. Myagmar. Relescope: an experiment in accelerating relationships. In *Extended Abstracts on Human Factors in Computing Systems*, 2005.

37. I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.
38. G. W. Flake, S. Lawrence, C. L. Giles, and F. Coetzee. Self-organization and identification of web communities. *IEEE Computer*, 35:66–71, 2002.
39. S. Fortunato, V. Latora, and M. Marchiori. Method to find community structures based on information centrality. *Physical Review E*, 70(5):056104, 2004.
40. L. Getoor. *Advanced Methods for Knowledge Discovery from Complex Data*, chapter Link-based classification. Springer, London, 2005.
41. L. Getoor and C. P. Diehl. Link mining: a survey. *SIGKDD Explorations Newsletter*, 7:3–12, 2005.
42. L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of link structure. *Machine Learning*, 3:679–707, 2003.
43. L. Getoor, E. Segal, B. Taskar, and D. Koller. Probabilistic models of text and link structure for hypertext classification. In *International Joint Conferences on Artificial Intelligence Workshop on Text Learning: Beyond Supervision*, 2001.
44. M. Girvan and M. E. J. Newman. Community structure in social and biological networks. In *Proceedings of National Academy of Science*, 2002.
45. A. Goldenberg, J. Kubica, P. Komarek, A. Moore, and J. Schneider. A comparison of statistical and machine learning algorithms on the task of link completion. In *Conference on Knowledge Discovery and Data Mining, Workshop on Link Analysis for Detecting Complex Behavior*, Washington, D.C., 2003.
46. R. Guimera, M. Sales-Pardo, and L. A. N. Amaral. Module identification in bipartite and directed networks. *Physical Review E*, 76:036102, 2007.
47. J. A. Hartigan. *Clustering Algorithms*. Wiley, New York NY, 1975.
48. O. Hassanzadeh, M. Sadoghi, and R. J. Miller. Accuracy of approximate string joins using grams. In *5th International Workshop on Quality in Databases at VLDB*, Vienna, Austria, 2007.
49. M. A. Hernández and S. J. Stolfo. The merge/purge problem for large databases. In *Proc. of the ACM Sigmod International Conference on Management of Data*, San Jose, CA, 1995.
50. H. Huang and J. S. Bader. Precision and recall estimates for two-hybrid screens. *Bioinformatics*, 25(3):372–378, 2009.
51. Z. Huang, X. Li, and H. Chen. Link prediction approach to collaborative filtering. In *ACM/IEEE-CS Joint Conference on Digital Libraries*, 2005.
52. Z. Huang and D. K. J. Lin. The Time-Series Link Prediction Problem with Applications in Communication Surveillance. *Informs Journal On Computing*, 21:286–303, 2008.
53. Z. Huang and D. D. Zeng. A link prediction approach to anomalous email detection. In *IEEE International Conference on Systems, Man, and Cybernetics*, Taipei, Taiwan, 2006.
54. P. Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.
55. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.
56. M. A. Jaro. Probabilistic linkage of large public health data files. *Statistics in Medicine*, 14:491–498, 1995.
57. D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, 2004.
58. T. Joachims. *Learning to Classify Text Using Support Vector Machines*. PhD thesis, University of Dortmund, 2002.
59. D. V. Kalashnikov, S. Mehrotra, and Z. Chen. Exploiting relationships for domain-independent data cleaning. In *SIAM International Conference on Data Mining*, Newport Beach, CA, 2005.
60. C. Kalyan and K. Chandrasekaran. Information leak detection in financial e-mails using mail pattern analysis under partial information. In *Proceedings of the 7th Conference on WSEAS*

*International Conference on Applied Informatics and Communications*, Athens, Greece, 2007.
61. A. E. Krause, K. A. Frank, D. M. Mason, R. E. Ulanowicz, and W. W. Taylor. Compartments revealed in food-web structure. *Nature*, 426(6964):282–285, 2003.
62. J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, Williamstown, MA, 2001.
63. A. Lancichinetti, S. Fortunato, and J. Kertesz. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11:033015, 2009.
64. V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Physical Review Letters*, 87(19):198701, 2001.
65. J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, Nevada, 2008.
66. J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1):2, 2007.
67. V. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707, 1966.
68. X. Li, P. Morie, and D. Roth. Semantic integration in text: From ambiguous names to identifiable entities. *AI Magazine Special Issue on Semantic Integration*, 26(1):45–58, 2005.
69. D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *International Conference on Information and Knowledge Management*, New Orleans, LA, 2003.
70. Q. Lu and L. Getoor. Link-based classification. In *Proceedings of the International Conference on Machine Learning*, 2003.
71. D. Lusseau and M. E. J. Newman. Identifying the role that animals play in their social networks. In *Proceedings of the Royal Society of London*, 2004.
72. J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.
73. S. A. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8:935–983, 2007.
74. A. McCallum, K. Nigam, and L. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the 6th International Conference On Knowledge Discovery and Data Mining*, Boston, MA, 2000.
75. A. McCallum and B. Wellner. Toward conditional models of identity uncertainty with application to proper noun coreference. In *International Workshop on Information Integration on the Web*, 2003.
76. L. McDowell, K. M. Gupta, and D. W. Aha. Cautious inference in collective classification. In *Association for the Advancement of Artificial Intelligence*, 2007.
77. D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proceedings of the 17th ACM conference on Information and Knowledge Management*, Napa Valley, CA, 2008.
78. A. E. Monge and C. P. Elkan. The field matching problem: Algorithms and applications. In *Proceedings of the 2nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Portland, Oregon, 1996.
79. A. E. Monge and C. P. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *Proceedings of the Special Interest Group on Management of Data Workshop on Research Issues on Data Mining and Knowledge Discovery*, Tucson, AZ, 1997.
80. J. Neville, M. Adler, and D. Jensen. Clustering relational data using attribute and link information. In *Proceedings of the Text Mining and Link Analysis Workshop, 18th International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, 2003.
81. J. Neville and D. Jensen. Iterative classification in relational data. In *Association for the Advancement of Artificial Intelligence Workshop on Learning Statistical Models from Relational Data*, 2000.

82. J. Neville and D. Jensen. Relational dependency networks. *Journal of Machine Learning Research*, 8:653–692, 2007.
83. H. B. Newcombe and J. M. Kennedy. Record linkage: making maximum use of the discriminating power of identifying information. *Communications ACM*, 5(11):563–566, 1962.
84. H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James. Automatic linkage of vital records. *Science*, 130:954–959, October 1959.
85. M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):066133, 2004.
86. M. E. J. Newman, A. L. Barabasi, and D. J. Watts. *The Structure and Dynamics of Networks*. Princeton University Press, Princeton, NJ, 2006.
87. M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:026113, 2004.
88. M. E. J. Newman and E. A. Leicht. Mixture models and exploratory analysis in networks. In *Proceedings of National Academy of Science*, 2007.
89. J. O'Madadhain, J. Hutchins, and P. Smyth. Prediction and ranking algorithms for event-based network data. *SIGKDD Explorations Newsletter*, 7(2):23–30, 2005.
90. M. Opper and D. Saad, editors. *Advanced Mean Field Methods*. Neural Information Processing Series. MIT Press, Cambridge, MA, 2001. Theory and practice, Papers from the workshop held at Aston University, Birmingham, 1999, A Bradford Book.
91. G. Palla, I. Dernyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.
92. H. Pasula, B. Marthi, B. Milch, S. Russell, and I. Shpitser. Identity uncertainty and citation matching. In *Neural Information Processing Systems*, Vancouver, Canada, 2003.
93. H. Poon and P. Domingos. Joint unsupervised coreference resolution with markov logic. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Honolulu, HI, 2008.
94. A. Popescul and L. H. Ungar. Statistical relational learning for link prediction. In *International Joint Conferences on Artificial Intelligence Workshop on Learning Statistical Models from Relational Data*, Acapulco, Mexico, 2003.
95. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA, USA, 1993.
96. M. J. Rattigan and D. Jensen. The case for anomalous link discovery. *SIGKDD Explorations Newsletter*, 7:41–47, 2005.
97. J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006.
98. M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.
99. M. Rosvall and C. T. Bergstrom. An information-theoretic framework for resolving community structure in complex networks. In *Proceedings of National Academy of Science*, 2007.
100. M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. In *Proceedings of National Academy of Science*, 2008.
101. P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.
102. C. R. Shalizi, M. F. Camperi, and K. L. Klinkner. Discovering functional communities in dynamical networks. *Statistical Network Analysis: Models, Issues, and New Directions*, pages 140–157, 2007.
103. P. Singla and P. Domingos. Entity resolution with markov logic. *IEEE International Conference on Data Mining*, 21:572–582, Hong Kong, China, 2006.
104. S. Slattery and M. Craven. Combining statistical and relational methods for learning in hypertext domains. In *Proceedings of the 8th international Conference on Inductive Logic Programming*, Madison, Wisconsin, 1998.
105. N. Spring, D. Wetherall, and T. Anderson. Reverse engineering the internet. *SIGCOMM Computer Communication Review*, 34(1):3–8, 2004.

106. E. Sprinzak, Y. Altuvia, and H. Margalit. Characterization and prediction of protein-protein interactions within and between complexes. *Proceedings of the National Academy of Sciences*, 103(40):14718–14723, 2006.
107. A. Szilagyi, V. Grimm, A. K. Arakaki, and J. Skolnick. Prediction of physical protein-protein interactions. *Physical Biology*, 2(2):S1–S16, 2005.
108. C. Tantipathananandh and T. Y. Berger-Wolf. Algorithms for identifying dynamic communities. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Paris, France, 2009.
109. B. Taskar, A. Pieter, and D. Koller. Discriminative probabilistic models for relational data. In *Conference on Uncertainty in Artificial Intelligence*, Alberta, Canada, 2002.
110. B. Taskar, M.-F. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2003.
111. S. Tejada, C. A. Knoblock, and S. Minton. Learning object identification rules for information integration. *Information Systems*, 26:2001, 2001.
112. I. Vragovic and E. Louis. Network community structure and loop coefficient method. *Physical Review E*, 74(1):016105, 2006.
113. S. Wasserman, K. Faust, and D. Iacobucci. *Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences)*. Cambridge University Press, Cambridge November 1994.
114. D. J. Watts and S. H. Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393(6684):440–442, June 1998.
115. Y. Weiss. Segmentation using eigenvectors: A unifying view. In *Proceedings of International Conference on Computer Vision*, 1999.
116. M. L. Wick, K. Rohanimanesh, K. Schultz, and A. McCallum. A unified approach for schema matching, coreference and canonicalization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, Nevada, 2008.
117. W. E. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Census Bureau, 1999.
118. H. Yu, A. Paccanaro, V. Trifonov, and M. Gerstein. Predicting interactions in protein networks by completing defective cliques. *Bioinformatics*, 22(7):823–829, 2006.
119. E. Zheleva, L. Getoor, J. Golbeck, and U. Kuter. Using friendship ties and family circles for link prediction. In *2nd ACM SIGKDD Workshop on Social Network Mining and Analysis*, Las Vegas, Nevada, 2008.
120. J. Zhu. *Mining Web Site Link Structure for Adaptive Web Site Navigation and Search*. PhD thesis, University of Ulster at Jordanstown, UK, 2003.