Journal of Artificial Intelligence Research 45 (2012) 363-441 Submitted 03/12; published 10/12

Transforming Graph Data for Statistical Relational Learning

Ryan A. Rossi rrossi@purdue.edu
Department of Computer Science, Purdue University
West Lafayette, IN 47907 USA

Luke K. McDowell lmcdowel@usna.edu
Department of Computer Science, U.S. Naval Academy
Annapolis, MD 21402, USA

David W. Aha david.aha@nrl.navy.mil
Navy Center for Applied Research in Artificial Intelligence
Naval Research Laboratory (Code 5514)
Washington, DC 20375, USA

Jennifer Neville neville@purdue.edu
Department of Computer Science, Purdue University
West Lafayette, IN 47907 USA

Abstract

Relational data representations have become an increasingly important topic due to the recent proliferation of network datasets (e.g., social, biological, information networks) and a corresponding increase in the application of Statistical Relational Learning (SRL) algorithms to these domains. In this article, we examine and categorize techniques for transforming graph-based relational data to improve SRL algorithms. In particular, appropriate transformations of the nodes, links, and/or features of the data can dramatically affect the capabilities and results of SRL algorithms. We introduce an intuitive taxonomy for data representation transformations in relational domains that incorporates *link transformation* and *node transformation* as symmetric representation tasks. More specifically, the transformation tasks for both nodes and links include (i) predicting their existence, (ii) predicting their label or type, (iii) estimating their weight or importance, and (iv) systematically constructing their relevant features. We motivate our taxonomy through detailed examples and use it to survey competing approaches for each of these tasks. We also discuss general conditions for transforming links, nodes, and features. Finally, we highlight challenges that remain to be addressed.

1. Introduction

In this article, we examine and categorize techniques for transforming relational data to improve Statistical Relational Learning (SRL) algorithms. Below, Section 1.1 first introduces relational data and SRL. We summarize the primary types of representations for relational data, and explain that we focus on data represented as graphs. Section 1.1 also describes how transforming the *content* (rather than the type) of this representation can improve SRL analysis. For instance, predicting new links in a graph can increase accuracy for relational node classification. Section 1.2 then identifies the scope of this article. Finally, Section 1.3 summarizes the organization and approach of this article, and includes a description of our taxonomy for relational representation transformation.

©2012 AI Access Foundation. All rights reserved.

1.1 Relational Data, SRL, and Representation Choices

The majority of research in machine learning assumes independently and identically distributed data. This independence assumption is often violated in relational data, which encode dependencies among data instances. For instance, people are often linked by business associations, and information about one person can be highly informative for a prediction task involving an associate of that person. More generally, relational data can be described as a set of nodes, which can be connected by one or more types of relations (or "links"). Relational information is seemingly ubiquitous; it is present in domains such as the Internet and the world-wide web (Faloutsos, Faloutsos, & Faloutsos, 1999; Broder et al., 2000; Albert, Jeong, & Barabási, 1999), scientific citation and collaboration (McGovern et al., 2003; Newman, 2001b), epidemiology (Pastor-Satorras & Vespignani, 2001; Moore & Newman, 2000; May & Lloyd, 2001; Kleczkowski & Grenfell, 1999) communication analysis (Rossi & Neville, 2010), metabolism (Jeong, Tombor, Albert, Oltvai, & Barabási, 2000; Wagner & Fell, 2001), ecosystems (Dunne, Williams, & Martinez, 2002; Camacho, Guimerà, & Nunes Amaral, 2002), bioinformatics (Maslov & Sneppen, 2002; Jeong, Mason, Barabási, & Oltvai, 2001), fraud and terrorist analysis (Neville et al., 2005; Krebs, 2002), and many others. The links in these data may represent citations, friendships, associations, metabolic functions, communications, co-locations, shared mechanisms, or many other explicit or implicit relationships.

Statistical relational learning (SRL) methods have been developed to address the problems of reasoning and learning in domains with complex relations and probabilistic structure (Getoor & Taskar, 2007). In particular, SRL algorithms leverage relational information in an attempt to learn models with higher predictive accuracy. A key characteristic of many relational datasets is a correlation or statistical dependence between the values of the same attribute across linked instances (e.g., two friends are more likely to share political views than two randomly selected people). This *relational autocorrelation* provides a unique opportunity to increase the accuracy of statistical inferences (Jensen, Neville, & Gallagher, 2004). Similarly, relational information can be exploited for many other reasoning tasks such as identifying useful patterns or optimizing systems (Easley & Kleinberg, 2010).

Representation issues—including knowledge, model, and data representation—have been at the heart of the artificial intelligence community for decades (Amarel, 1968; Minsky, 1974; Russell & Norvig, 2009). All of these are important, but here we focus on *data representation* issues, simple examples of which include the choices of whether to discretize continuous features or to add higher-order polynomial features. Such decisions can have a significant effect on the accuracy and efficiency of AI algorithms. They are especially critical for the performance of SRL algorithms because, in relational domains, there is an even larger space of potential data representations to consider. The complex structure of relational data can often be represented in a variety of ways and the choice of specific data representation can impact both the applicability of particular models/algorithms and their performance. Specifically, there are two categories of decisions that need to be considered in the context of *relational data representation.*

First, we have to consider the *type* of data representation to use (cf., the hierarchy of De Raedt, 2008, ch. 4). For instance, relational data can be propositionalized for the application of standard, non-relational learning algorithms. More often, in order to fully exploit the relational information, SRL researchers have chosen to represent the data either using an attributed graph in a relational database (see e.g., Friedman, Getoor, Koller, & Pfeffer, 1999), or via logic programs (see e.g., Kersting & De Raedt, 2002).[1] Each choice has different strengths. In this article, we focus on the graph-based representation, which has been a common choice for addressing the growing interest in network data and applications for analyzing electronic communication and online social networks such as Facebook, Twitter, Flickr, and LinkedIn (Mislove, Marcon, Gummadi, Druschel, & Bhattacharjee, 2007; Ahmed, Berchmans, Neville, & Kompella, 2010). Specifically, we assume a graph-based data representation $G = \langle V, E, \mathbf{X}^V, \mathbf{X}^E \rangle$ where the nodes V are entities (e.g., people, places, events) and the links E represent relationships among those entities (e.g., friendships, citations). $\mathbf{X}^V$ is a set of features about the entities in V. Likewise, the set of features $\mathbf{X}^E$ provides information about the relation links in E.

Next, given the type of representation, we must consider the specific *content* of the data representation, for which there is a large space of choices. For instance, features for the nodes and links of a graph can be constructed using a wide range of aggregation functions, based on multiple kinds of links and paths. SRL researchers have already recognized the importance of such data representation choices (e.g., Getoor & Diehl, 2005), and many separate studies have examined techniques for feature construction (Neville, Jensen, Friedland, & Hay, 2003), node weighting (Tang, Musolesi, Mascolo, & Latora, 2009), link prediction (Taskar, Wong, Abbeel, & Koller, 2003), etc. However, this article is the first to comprehensively survey approaches to relational representation transformation for graph-based data.

Given a set of (graph-based) relational data, we define *relational representation transformation* as any change to the space of links, nodes, and/or features used to represent the data. Typically, the goal of this transformation is to improve the performance of some subsequent SRL application. For instance, in Figure 1 the original graph representation G is transformed into a new representation $\tilde{G}$ where links, nodes, and features (such as link weights) have been added, and some links have been removed. Some SRL algorithm or analysis is then applied to the new representation, for instance to classify the nodes or to identify anomalous links. The particular transformations that are used to produce $\tilde{G}$ will vary depending upon the intended application, but can sometimes substantially improve the accuracy, speed, or complexity of the final application. For instance, Gallagher, Tong, Eliassi-Rad, and Faloutsos (2008) found that adding links between similar nodes could increase node classification accuracy by up to 15% on some tasks. Similarly, Neville and Jensen (2005) demonstrated that adding nodes which represent underlying groups enabled both simpler inference and increased accuracy.

1.2 Scope of this Article

This article focuses on examining and categorizing various techniques for changing the representation of graph-based relational data. As shown in Figure 1, we typically view these changes as a pre-processing step that enables increased accuracy or speed for some other task, such as object classification. However, an output of these techniques can itself be valuable. For instance, the administrators of a social network may be interested in

1. In the latter case, the applicable SRL algorithms are often referred to as probabilistic inductive logic programming (ILP) (De Raedt & Kersting, 2008).

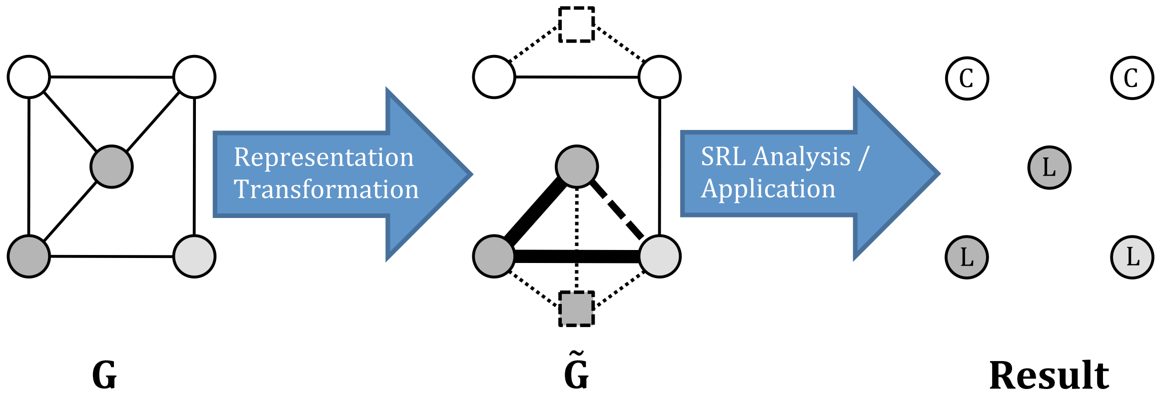


Figure 1: Example Transformation and Subsequent Analysis: The original relational representation G is transformed into $\tilde{G}$ where dotted lines represent predicted links, squares represent predicted nodes, and bold links represent link weighting. Changes may be based on link structure, link features, and node features (here, similar node shadings indicate similar feature values). Some SRL analysis is then applied to the new representation. In this example, the SRL analysis produces a label (C or L) for each node, as with the example task discussed in Section 2.1. This article focuses on the representation transformation (left side of the figure), not the subsequent analysis.

link prediction so that predicted links can be presented to their users as potential new "friendship" links. Alternatively, these techniques may also be applied to improve the comprehensibility of a model. For example, the prediction of protein-protein interactions provides insights into protein function (Ben-Hur & Noble, 2005). Thus, the techniques we survey may be used for multiple purposes, and relevant publications may have used them in different contexts. Regardless of the original context, we will examine the general applicability and benefits of each technique. After such techniques have been applied, the transformed data can be used as is (e.g., for friendship suggestions), examined for greater understanding, used for some other task (e.g., for object classification), or used recursively as the input for another representation change (e.g., as in object/node prediction followed by link prediction).

We do *not* attempt to survey the many methods that could be used for SRL analysis (e.g., the right side of Figure 1), although the relevant set of methods for such analysis overlaps with the set of methods that facilitate the transformations we consider. For instance, collective classification (Neville & Jensen, 2000; Taskar, Abbeel, & Koller, 2002) is an important SRL application that we define in Section 2 and use as a running example of an SRL analysis task. The output of such classification could also be used to create new attributes for the nodes (a data representation change). We discuss this possibility in Section 6.2, but focus on a few cases where such node labeling is particularly useful as a pre-processing step (e.g., before applying certain "stacked" algorithms), rather than surveying the wide range of possible classification algorithms, whether collective or not. Likewise, we do not survey issues in model and knowledge representation, such as whether the statistical dependencies between nodes, links, and features should be modeled with Structural Logistic Regression (Popescul, Popescul, & Ungar, 2003b) or with a Markov Logic Network (Domingos & Richardson, 2004). We consider such issues only briefly, in Section 8.4.

Furthermore, we focus on transformations that change the content of the graph data representation. In particular, we examine transformations to graph data that modify the set of links or nodes, or modify their features. We do not consider changing the graph data to a different *type* of representation, e.g., by propositionalizing the data or by changing to a logic program. However, some of the transformations we discuss, such as node or link feature aggregation, are a form of propositionalization. In addition, Section 6.3.3 describes a number of techniques for structure learning of logic programs, because these techniques are closely related to the analogous problem of feature construction for graph-based representations. Finally, many of the other techniques that we discuss are also applicable to logical representations. For instance, link weighting could be applied to weight the known relations before using a logic program to detect anomalous objects. We focus, however, on the methods most useful for transforming graph-based representations.

1.3 Approach and Organization of this Article

There are many dimensions of relational data transformation, which complicate the task of understanding and selecting the most appropriate techniques. To assist in this process, we introduce a simple and intuitive taxonomy for representation transformation that identifies *link transformation* and *node transformation* as symmetric representation tasks. More specifically, the transformation tasks for both nodes and links include (i) predicting their existence, (ii) predicting their label or type, (iii) estimating their weight or importance, and (iv) constructing their relevant features. In addition, we propose a taxonomy for constructing both link and node features that consists of non-relational features, topology features, relational node-value features, and relational link-value features. For each relational transformation task, we survey the applicable techniques, examine necessary conditions, and provide detailed examples and comparisons.

This article is organized as follows. The next section presents our taxonomy for relational representation transformation and discusses a motivating example. In Section 3, we review the algorithms for link prediction, while Section 4 examines the task of link interpretation (i.e., constructing link labels, link weights, and link features). Sections 5 and 6 consider the corresponding prediction and interpretation tasks for nodes instead of links. In Section 7, we summarize algorithms that jointly transform nodes and links. Section 8 discusses methods for evaluating representation transformations and challenges for future work, and Section 9 concludes.

2. Overview and Motivating Example

In this section we first introduce a running example based on the classification of data from Facebook, then describe how relational algorithms could be used to perform this task. Next, we introduce a taxonomy for relational representation transformation and explain how each type of transformation could aid the Facebook classification task. Finally, we formally define each type of relational representation transformation.

2.1 Motivating SRL Analysis Example: A Classification Task

As an example, we consider hypothetical data inspired by Facebook (www.facebook.com), one of the most popular online social networks. We assume that we are given a graph $G = \langle V, E, \mathbf{X}^V, \mathbf{X}^E \rangle$ where the nodes V are users [2] and the links E represent friendships in Facebook. $\mathbf{X}^V$ is a set of features about the users in V such as their gender, relationship status, school, favorite movies, or musical preference (though information may be missing for some users). Likewise, the set of features $\mathbf{X}^E$ provides information about the friendship links in E such as the time of formation or possibly the contents of the message that was sent when the link formation was requested by one of the users.

The example SRL analysis task (see Figure 1) is to predict the political affiliation (liberal, moderate, or conservative) of every node (person) in G. We assume that this affiliation, which we call the *class label* of a node, is known for some but not all of the people in G.[3] Moreover, we assume that a user's political affiliation is likely to be correlated with the characteristics of that user and (to a lesser degree) that user's friends. The next section summarizes how these correlations can be used for classification.

For this example, we assume that links are simple, binary friendship connections. However, other link types could be used to represent other kinds of relationships. For instance, a link might indicate that two people have communicated via a "wall-post" message, or that two people have chosen to join the same Facebook group. In addition, the notion of friendship in Facebook is very weak and thus a significant portion of a person's "friends" are often only casual acquaintances. Thus, representation changes such as link deletion or weighting may have a significant impact on classification accuracy. For notational purposes, we add a tilde to the top of each graph component's symbol to indicate that it has undergone some transformation (e.g., the modified link set E is denoted by $\tilde{E}$).

2.2 Background: Features and Methods for Classification

To predict the political affiliation of Facebook users, conventional classification approaches would ignore the links and classify each user using only information known about that user, such as their gender or location. We assume that such information is represented in the form of *non-relational features*, which are those features that can be computed directly from $\mathbf{X}^V$ without considering the links E. We refer to classification based only on these features as *non-relational classification*. Alternatively, in *relational classification*, the links are explicitly used to construct additional *relational features* to capture information about each user's friends. For instance, a relational feature could compute, for each user, the proportion of friends that are male or that live in a particular region. Using such relational information can potentially increase classification accuracy, though may sometimes decrease accuracy as well (Chakrabarti, Dom, & Indyk, 1998). Finally, even greater (and usually more reliable) increases can occur when the class labels (e.g., political affiliations) of the linked users are used instead to derive relevant features (Jensen et al., 2004). For instance,

2. In general, there may be more than one type of node. For instance, nodes in a citation network may represent papers or authors.
3. Later, we discuss the representation change of *node labeling*, which also constructs an estimated label for every node. As discussed in Section 1.2, representation changes can sometimes resemble the output of SRL analysis, but we focus on changes that are particularly useful as pre-processing before some subsequent SRL analysis.

a "class-label" relational feature could compute, for each user, the proportion of friends that have liberal views. However, using such features is challenging since some or all of the labels are initially unknown, and thus typically must be estimated and then iteratively refined in some way. This process of jointly inferring the labels of interrelated nodes is known as *collective classification* (CC).

CC requires both models and inference procedures that use inferences about one user to affect inferences about related users. Many such algorithms have been considered for CC, including Gibbs Sampling (Jensen et al., 2004), relaxation labeling (Chakrabarti, Dom, & Indyk, 1998), belief propagation (Taskar et al., 2002), ICA (Neville & Jensen, 2000; Lu & Getoor, 2003), and weighted neighbor techniques (Macskassy & Provost, 2007). See the work of Sen et al. (2008) for a survey.

As a concrete example of SRL analysis, we explain many of the techniques in this survey in terms of the Facebook classification task, with a special emphasis on CC. However, the features and the transformation techniques apply to many other SRL tasks and data sets such as relationship classification, anomalous link detection, entity resolution, or group discovery (Getoor & Diehl, 2005).

2.3 Representation Transformation Tasks for Improving SRL

Figure 2 shows our proposed taxonomy for relational representation transformation. The two main tasks in this taxonomy are **link transformation** and **node transformation**. We find that there is a powerful and elegant symmetry between these two tasks. In particular, the link and node representation transformation tasks can be decomposed into prediction and interpretation tasks. The former task involves predicting the existence of new nodes and links. The latter task of interpretation involves three parts: constructing the weights, labels, or features of nodes or links. Together, this yields eight distinct transformation tasks as shown in the leaves of the taxonomy in Figure 2. Underneath these eight tasks in the figure, we list the primary graph component that is modified by each task (i.e., $\tilde{V}$, $\tilde{E}$, $\tilde{\mathbf{X}}^V$, or $\tilde{\mathbf{X}}^E$), followed by an illustration of a possible representation change for that task. In the text below, we summarize Figure 2, organized around the four larger categories of link prediction, link interpretation, node prediction, and node interpretation.

First, **link prediction** adds new links to the graph. The sample graph for this task (Figure 2A) shows a link being predicted where the similarity between two nodes has been used to predict a new link between them. Intuitively, Facebook users that share the values of many non-relational features may also share the same political affiliation. Thus, adding links between such people should increase autocorrelation and improve the accuracy of collective classification. There are many simple link prediction algorithms based on similarity, neighbor properties, shortest path distances, infinite sums over paths (i.e. random walks), and other strategies. Section 3 provides more detail on these techniques.

Second, there are several types of **link interpretation**, which involves constructing weights, labels, or features for the existing links. For instance, in many graphs (including our Facebook data), not all links (or friendships) are of equal importance. Thus, Figure 2B shows the result of performing *link weighting*. In this case, weights are based on the similarity between the feature values of each pair of linked nodes, under the assumption that high similarity may indicate stronger relationships. (Link prediction techniques may also

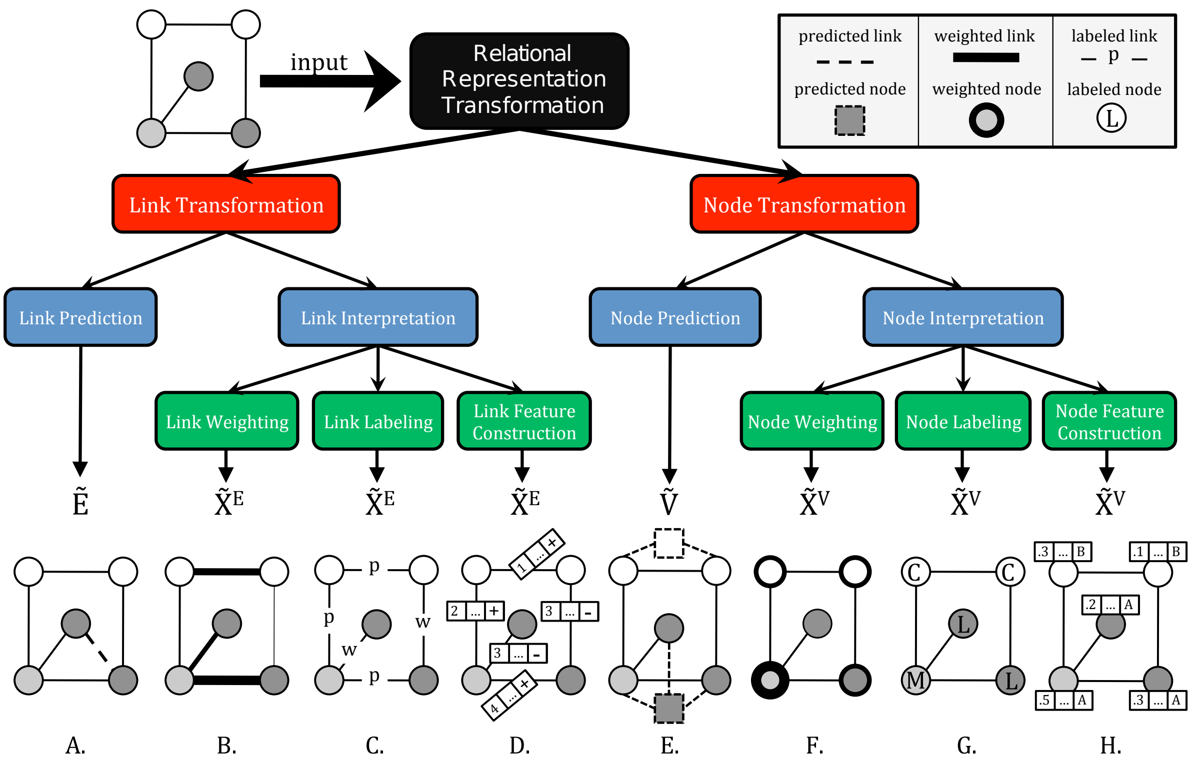


Figure 2: Relational Representation Transformation Taxonomy: Link and node transformation are formulated as symmetric tasks leading to four main transformation tasks: predicting links, interpreting links, predicting nodes, and interpreting nodes. Each task yields a modified graph component: $\tilde{E}$, $\tilde{\mathbf{X}}^E$, $\tilde{V}$, or $\tilde{\mathbf{X}}^V$, respectively. Interpretation is further divided into weighting, labeling, or constructing features. Examples of each of the tasks in relational representation transformation are shown under the leaves of the taxonomy. In these example graphs, nodes with similar shadings have similar feature values.

use such similarity measures, but for identifying probable new links, rather than weighting existing links.) Alternatively, *link labeling* may be used to assign some kind of discrete label to each link. For instance, Figure 2C shows how links might be labeled as either "personal" (p) or "work" (w) related, e.g., based on known feature values or an analysis of communication events between the linked users. On the other hand, links might instead be labeled as having positive or negative influence (i.e., labeled as $+/-$). Finally, Figure 2D shows how *link feature construction* can be used to add more general kinds of feature values to each link. For instance, a link feature might count the number of communication events that occurred between two people or the number of friends in common. Link weighting and labeling could perhaps be viewed as special cases of link feature construction, but we separate them because later sections will show how the most useful techniques for each task differ. All three of these link interpretation tasks could help with our example classification problem. In particular, a model learned to predict political affiliation might choose to place special emphasis on links that are highly weighted or that are labeled as personal. Other link features might be used to represent more complex dependencies, for instance modeling influence from a user's "work" friendships, but only for friendship links between nodes where there are a large number of friends in common. More details on these techniques are provided in Section 4.

Third, **node prediction** adds additional nodes (and associated links) to the graph. For instance, Figure 2E shows the result after relational clustering has been applied to discover two latent groups in the graph, where each user is now connected to one latent group node. A discovered node in Facebook might represent types of social processes, influences, or a tightly knit group of friends. The clustering or other techniques used to identify the new nodes could be designed to identify people that are particularly similar with respect to a relevant characteristic, such as their political affiliation. The new nodes and associated links could then be used in several ways. For instance, though not present in the small example of Figure 2E, some nodes that were far away (in terms of shortest path length) in the original graph may be much closer in the new graph. Thus, links to a latent node may allow influence to propagate more effectively when an algorithm such as CC is applied. Alternatively, identification of distinct latent groups may even enable more efficient or accurate algorithms to be applied separately to each group (Neville & Jensen, 2005). Node prediction is discussed further in Section 5.

Finally, there are several types of **node interpretation**, which involves constructing weights, labels, or feature values for existing nodes. For instance, as with links, some nodes may be more influential than others and thus should have more weight. Figure 2F demonstrates *node weighting*, where the weights might be assigned based on the numbers of friends or via the PageRank/eigenvector techniques. See Section 6.1 for more details. Alternatively, Figure 2G shows an example of *node labeling*. Here the graph represents a training graph, and each node has been given an estimated label of conservative (C), liberal (L), or moderate (M). Such labels might be estimated using only the non-relational features or via textual analysis. While most classification algorithms learn a model based on true labels in the training graph, some approaches instead first compute such estimated labels, then learn a model from this new representation (Kou & Cohen, 2007). Section 6.2 discusses how this can simplify inference. Finally, Figure 2H shows the result of *node feature construction,* where arbitrary feature values are added to each node. For instance, suppose we find that users with relatively few Facebook friends are often moderate while those with many friends are often liberal. In this case, a feature counting the number of friends for each node would be useful. To more directly exploit autocorrelation, a different feature might count the proportion of a user's friends that are conservative, or the most common political affiliation of a user's friends. Any feature that is correlated with political affiliation could be used to improve the performance of a classification algorithm for our example problem. Identifying and/or computing such features is essential to the performance of most SRL algorithms but can be very challenging; Section 6.3 considers this process.

In Table 2.3, we summarize some of the most prominent techniques for performing these tasks of link prediction, link interpretation, node prediction, and node interpretation. Sections 3-6 provide more detail about each category in turn.

2.4 Relational Representation Transformation: Definitions and Terminology

We assume that the initial relational data is represented as a graph $G = \langle V, E, \mathbf{X}^V, \mathbf{X}^E \rangle$ such that each $v_i \in V$ corresponds to node i and each edge $e_{ij} \in E$ corresponds to a (directed) link between nodes i and j. $\mathbf{X}^V$ is a set of features about the nodes in V, and $X_k^V \in \mathbf{X}^V$ is the k^{th} such feature. Likewise, $\mathbf{X}^E$ is a set of features about the links in E, and $X_k^E \in \mathbf{V}^E$ is the k^{th} such feature. The features $\mathbf{X}^E$ could refer to link weights, distances, or types, among other possibilities. The preceding notation lets us identify, for instance, the values of a particular feature X_k^V for all nodes. Alternatively, $\mathbf{x_i^v}$ refers to a vector containing all of the feature values for a particular node v_i, and $\mathbf{x_{ij}^e}$ contains all of the feature values for a particular edge e_{ij}. Table 2.3 summarizes this notation.

Relational representation transformation is the process of transforming the original graph G into some new graph $\tilde{G} = \langle \tilde{V}, \tilde{E}, \tilde{\mathbf{X}}^V, \tilde{\mathbf{X}}^E \rangle$ by an arbitrary set of transformation techniques. During this process, nodes, links, weights, labels, and general features may be added, and nodes and links may be removed. In theory, the transformation seeks to optimize some objective function (for instance, to maximize the autocorrelation), although in practice the objective function may not be completely specified or guaranteed to be improved by the transformation. We now define more specifically the four primary parts of relational representation transformation:

Definition 2.1 (Link Prediction) Given the nodes V, observed links E and/or the feature set $\mathbf{X} = (\mathbf{X}^E, \mathbf{X}^V)$, the link prediction task is defined as the creation of a modified link set $\tilde{E}$ such that $E \neq \tilde{E}$. Usually, this involves adding new links that were not present in E, but links may also be deleted.

Definition 2.2 (Link Interpretation) Given the nodes V, observed links E and/or the feature set $\mathbf{X} = (\mathbf{X}^E, \mathbf{X}^V)$, the link interpretation task is defined as the creation of a new link feature $\tilde{X}_k^E$ where $\tilde{X}_k^E \notin \mathbf{X}^E$. This task may estimate a feature value for every link. Alternatively, the values of $\tilde{X}_k^E$ may be only partially estimated, for example, if the original features have missing values or if additional links are also introduced during link prediction.

Definition 2.3 (Node Prediction) Given the nodes V, links E and/or the feature set $\mathbf{X} = (\mathbf{X}^E, \mathbf{X}^V)$, node transformation is defined as the creation of a modified node set $\tilde{V}$ such that $V \subset \tilde{V}$. In addition, many node prediction tasks simultaneously create new links,

Relational Representation Transformation		
	Links	**Nodes**
Prediction	⋆ **Adamic/Adar** (Adamic & Adar, 2001), **Katz** (Katz, 1953), **and others** (Liben-Nowell & Kleinberg, 2007) ⋆ **Text or Feature Similarity** (Macskassy, 2007) ⋆ **Classification via RMN** (Taskar et al., 2003) **or SVM** (Hasan, Chaoji, Salem, & Zaki, 2006)	⋆ **Spectral Clustering** (Neville & Jensen, 2005), **Mixed-Membership Relational Clustering** (Long et al., 2007) ⋆ **LDA** (Blei, Ng, & Jordan, 2003), **PLSA** (Hofmann, 1999), ⋆ **Hierarchical Clustering via Edge-betweenness** (Newman & Girvan, 2004)
Weighting	⋆ **Latent Variable Estimation** (Xiang, Neville, & Rogati, 2010) ⋆ **Linear Combination of Features** (Gilbert & Karahalios, 2009) ⋆ **Aggregating Intrinsic Information** (Onnela, Saramaki, Hyvonen, Szabo, Lazer, Kaski, Kertesz, & Barabasi, 2007)	⋆ **Betweenness** (Freeman, 1977), **Closeness** (Sabidussi, 1966) ⋆ **HITs** (Kleinberg, 1999), **Prob. HITs** (Cohn & Chang, 2000), **SimRank** (Jeh & Widom, 2002) ⋆ **PageRank** (Page, Brin, Motwani, & Winograd, 1999), **Topical PageRank** (Haveliwala, 2003; Richardson & Domingos, 2002)
Labeling	⋆ **LDA** (Blei et al., 2003), **PLSA** (Hofmann, 1999), ⋆ **Link Classification via Logistic Regression** (Leskovec, Huttenlocher, & Kleinberg, 2010), **Bagged Decision Trees** (Kahanda & Neville, 2009),	⋆ **LDA** (Blei et al., 2003), **PLSA** (Hofmann, 1999), ⋆ **Node Classification via Stacked Model** (Kou & Cohen, 2007) or **RN** (Macskassy & Provost, 2003)
Feature Construction	⋆ **Link Feature Similarity** (Rossi & Neville, 2010) ⋆ **Link Aggregations** (Kahanda & Neville, 2009) ⋆ **Graph Features** (Lichtenwalter, Lussier, & Chawla, 2010)	⋆ **MLN Structure Learning** (Kok & Domingos, 2009, 2010) ⋆ **Database Query Search** (Popescul et al., 2003b), **RPT** (Neville, Jensen, Friedland, et al., 2003) ⋆ **FOIL**, **nFOIL** (Landwehr, Kersting, & De Raedt, 2005), **kFOIL** (Landwehr, Passerini, De Raedt, & Frasconi, 2010), **Aleph** (Srinivasan, 1999),

Table 1: Summary of Techniques: A summary of prominent graph transformation techniques for the tasks of predicting the existence of nodes and links and interpreting them by weighting, labeling, and constructing general features.

Symbol	Description
G	Initial graph
$\tilde{G}$	Transformed graph
E	Initial link set
V	Initial node set
$\mathbf{X}^E$	Initial set of link features
$\mathbf{X}^V$	Initial set of node features
X_k^E	Initial link feature k ($X_k^E \in \mathbf{X}^E$) (for one feature, values for all links)
X_k^V	Initial node feature k ($X_k^V \in \mathbf{X}^V$) (for one feature, values for all nodes)
$\mathbf{x_{ij}^e}$	Initial feature vector for e_{ij} (for one link, values for all link features)
$\mathbf{x_i^v}$	Initial feature vector for v_i (for one node, values for all node features)
Other symbols	**Description**
$\mathbf{A}$	Adjacency matrix of the graph
$\Gamma(v_i)$	Neighbors of v_i
δ	Cut-off value

Table 2: Summary of Notation used in this Survey: The top half of the table shows symbols that are sometimes written with a tilde on top of the symbol, indicating the result of some transformation. For conciseness, the table demonstrates this notation only for G and $\tilde{G}$.

e.g., between an initial node $v_i \in V$ and a predicted node $\tilde{v}_j \in \tilde{V}$. Thus, this task may also produce a modified link set $\tilde{E}$.

Definition 2.4 (Node Interpretation) Given the nodes V, observed links E and/or the feature set $\mathbf{X} = (\mathbf{X}^E, \mathbf{X}^V)$, the node interpretation task is defined as the creation of a new node feature $\tilde{X}_k^V$ where $\tilde{X}_k^V \notin \mathbf{X}^V$. As with link interpretation, the values of $\tilde{X}_k^V$ may be estimated for only some of the nodes. The node feature $\tilde{X}_k^V$ could represent node weights, labels, or other general features.

Section 2.2 introduced the notion of a non-relational feature, which is a node feature $\tilde{X}_k^V$ that can be constructed without making use of the links (i.e., without using E or $\mathbf{X}^E$). Such features are sometimes referred to in other articles as *attributes* or *intrinsic features*. Other important terms can also be referred to in multiple different ways. To aid the reader, Table 2.4 summarizes the key synonyms for the terms that are found most often in the literature.

3. Link Prediction

This section focuses on predicting the existence of links while Section 4 considers link interpretation. Given the initial graph $G = \langle V, E, \mathbf{X}^V, \mathbf{X}^E \rangle$, we are interested in creating a modified link set $\tilde{E}$, usually through the prediction of new links that were not present

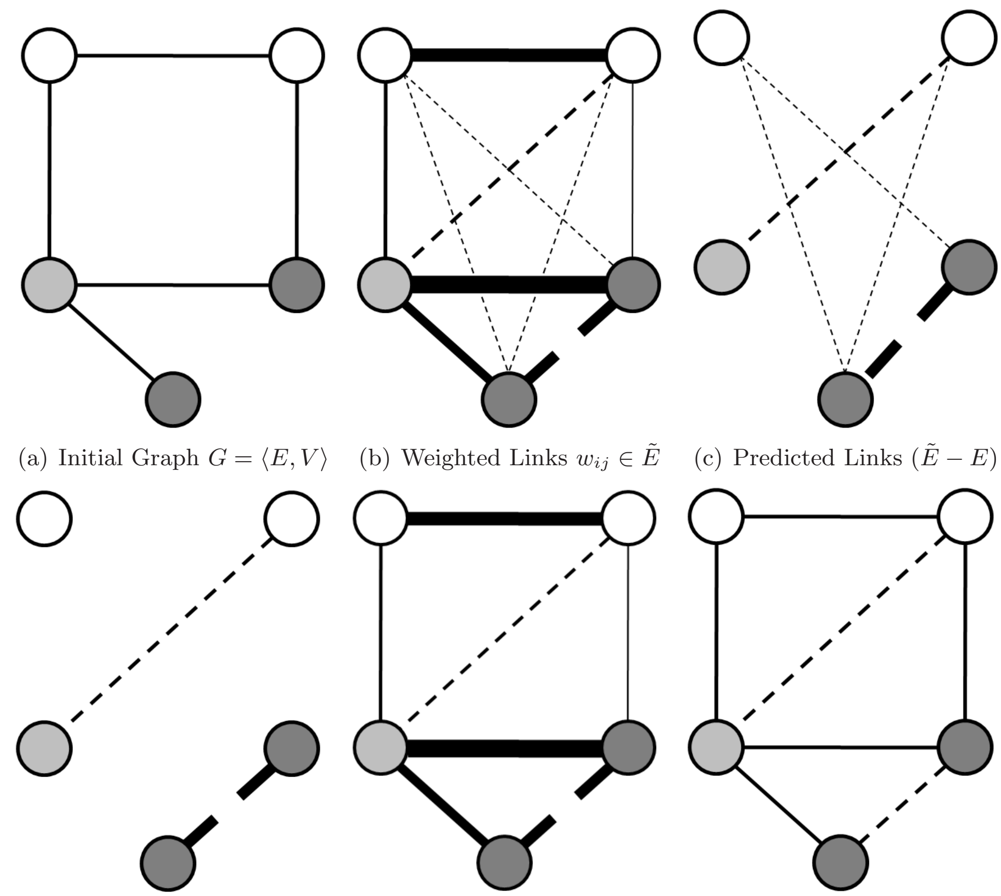
Term	Potential synonyms
Nodes	Vertices, points, objects, entities, individuals, users, constants, ...
Links	Edges, relationships, ties, arcs, events, interactions, predicates
Topology	Link/network/graph structure, relational information
Features	Attributes, variables, co-variates, queries, predicates, ...
Graph Measures	Topology-based metrics (such as proximity, centrality, betweenness, ...)
Similarity	Distance (the inverse of similarity), likeness
Clusters	Classes, communities, groups, roles, topics
Non-relational Features	Intrinsic attributes/features, local attributes/features, ...
Relational Features	Features, link-based features, graph features, aggregates, queries, ...
Structure Learning	Feature generation/construction, hypothesis learning
Parameter Learning	Model selection, function learning

Table 3: Synonyms in the Literature: A summary of possible synonyms found in the literature for important terms related to relational data.

in E. This task can be motivated in several ways. For instance, there may be a need to predict *missing links* that are not present in E because of incomplete data collection or other problems. Similarly, we may be interested in predicting *hidden links*, where we assume that there exists some unobservable interactions and the goal is to discover and model these interactions. For example, in a network representing criminals or terrorist activity, we may seek to predict a link between two people (nodes) that are not directly connected but whose actions share some common motivation or cause. For both missing and hidden links, predicting such links may improve the accuracy of a subsequent learned model. Alternatively, we may seek to predict *future links* in an evolving network, such as new friendships or connections that will be formed next year. We might also be interested in predicting links between objects that are spatially related. Finally, we may wish to predict *beneficial links*, for instance, predicting pairs of individuals that are likely to be successful working together.

Figure 3 summarizes one general approach that is often used for these link prediction tasks. In summary, scores or weights are computed for every pair of nodes in the graph, as shown in Figure 3(b). Predicted links with a weight greater than some threshold δ, along with the original links, are used to create the new link set $\tilde{E}^+$ (shown in Figure 3(e)). (At this step, original links with very low weight could also be deleted if appropriate.) As a final step, the weights of the predicted links are often discarded, yielding a new graph with uniform link weights as shown in Figure 3(f).

The key challenge in this approach is how to compute a weight or score for each possible link. The information used for this computation provides a natural way to categorize the link prediction techniques. Below, Section 3.1 describes techniques that use only the non-relational features of the nodes (ignoring the initial links), while Section 3.2 describes "topology-based" techniques that use only the graph structure (i.e., the links or relations).



(a) Initial Graph $G = \langle E, V \rangle$ (b) Weighted Links $w_{ij} \in \tilde{E}$ (c) Predicted Links $(\tilde{E} - E)$

(d) Pruning Predicted Links $(\tilde{E}^{>\delta})$ (e) $\tilde{E}^{+} := \tilde{E}^{>\delta} + E$ (f) $\tilde{E}^{+}$ with Uniform Link Weights

Figure 3: Example Demonstrating a General Approach to Link Prediction: The initial graph (a) is used as input to a link predictor, yielding a complete graph (b) where the weights w_{ij} are estimated between all pairs of nodes. The next step shows the removal of the initial (observed) links from consideration (c), followed by a pruning of all predicted links with a weight below some cut-off value δ (d). The remaining predicted links are then combined with the initial links (e). Often, the estimated weights on the initial and predicted links are then discarded, leaving a uniform weight graph (f).

Finally, Section 3.3 describes hybrid techniques that exploit both the node features and the graph structure.

3.1 Non-relational (Feature-Based) Link Prediction

In this section, we consider link predictors that do not exploit the graph structure or relational features derived using the graph structure. We are given an arbitrary pair of nodes

v_i and v_j from the graph such that each node is represented by a feature vector $\mathbf{x_i^v}$ and $\mathbf{x_j^v}$, respectively. *Feature-based link prediction* is defined as using an arbitrary similarity measure $S(x_i^v, x_j^v)$ as a means to estimate the likelihood that a link should exist between v_i and v_j. Typically, a link is created if the similarity exceeds some fixed cut-off value; another strategy is to predict links among the n% of all such node pairs with highest similarity.

A traditional approach is to simply *define* a measure of similarity between two objects, possibly based on knowledge of the application and/or problem-domain. There are many similarity metrics that have been proposed such as mutual information, cosine similarity, and many others (Lin, 1998). For instance, Macskassy (2007) represents the textual content of each node as a feature vector and uses cosine similarity to create new links between nodes in a graph. Macskassy showed that the combination of the initial links with the predicted text-based links increased classification accuracy compared to using only the initial links or the text-based links. In addition to leveraging textual information to predict links, we might use any arbitrary set of features combined with a proper measure of similarity for link prediction. For instance, many recommender systems implicitly predict a link between two users based on the similarity between their ratings of items such as movies or books (Adomavicius & Tuzhilin, 2005; Resnick & Varian, 1997). In this case, cosine similarity or correlation are commonly used as similarity metrics.

Alternatively, a similarity measure can be *learned* for predicting link existence. The link prediction problem can be transformed into a standard supervised classification problem where a binary classifier is trained to determine the similarity between two nodes based on their feature vectors. One such approach from the work of Hasan et al. (2006), who have used Support Vector Machines (SVMs) for link prediction and found that a non-relational feature (keyword match count) was most useful for predicting links in a bibliographic network. There are many link prediction approaches (Taskar et al., 2003; Getoor, Friedman, Koller, & Taskar, 2003) that apply traditional machine learning algorithms. However, most of them use features based on the graph structure as well as the non-relational features that are the focus of this section. Thus, we discuss such techniques further in Section 3.3.

Finally, variants of *topic models* can be used for link prediction. These types of models traditionally use only the text from documents (non-relational information) to infer a mixture of latent topics for each document. Inter-document topic similarity can then be used as a similarity metric for link prediction (Chang & Blei, 2009). However, because many topic models are capable of performing joint transformation of the nodes and links, we defer full discussion of such techniques to Section 7.

3.2 Topology-Based Link Prediction

Topology-based link prediction uses the local relational neighborhood and/or the global graph structure to predict the existence of unobserved links. Table 3.2 summarizes some of the most common metrics that have been used for this task. Below, we discuss many of these approaches, starting from the simplest local metrics and moving to the more complex techniques based on global measures and/or supervised learning. For a systematic study of many of these approaches applied to social network data, see the work of Liben-Nowell and Kleinberg (2007).

Local Node Metrics	**Description**
Common Neighbors	Number of common neighbors between x and y, $w(x,y) = \lvert\Gamma(x) \cap \Gamma(y)\rvert$ (Newman, 2001a)
Jaccard's Coefficient	Probability that x and y share common neighbors (normalized), $w(x,y) = \frac{\lvert\Gamma(x)\cap\Gamma(y)\rvert}{\lvert\Gamma(x)\cup\Gamma(y)\rvert}$ (Jaccard, 1901; Salton & McGill, 1983)
Adamic/Adar	Similar to common neighbors, but assigns more weight to rare neighbors, $w(x,y) = \sum_{z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{\log\lvert\Gamma(z)\rvert}$ (Adamic & Adar, 2001)
RA	Essentially equivalent to Adamic/Adar if $\lvert\Gamma(z)\rvert$ is small, $w(x,y) = \sum_{z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{\lvert\Gamma(z)\rvert}$ (Zhou, Lü, & Zhang, 2009)
Preferential Attachment	Probability of a link between x and y is the product of the degree of x and y, $w(x,y) = \lvert\Gamma(x)\rvert \cdot \lvert\Gamma(y)\rvert$ (Barabási & Albert, 1999)
Cosine Similarity	$w(x,y) = \frac{\lvert\Gamma(x)\cap\Gamma(y)\rvert}{\sqrt{\lvert\Gamma(x)\rvert\cdot\lvert\Gamma(y)\rvert}}$ (Salton & McGill, 1983)
Sorensen Index	$w(x,y) = \frac{2\times\lvert\Gamma(x)\cap\Gamma(y)\rvert}{\lvert\Gamma(x)\rvert+\lvert\Gamma(y)\rvert}$ (Green, 1972; Zhou et al., 2009)
Hub Index	Nodes with large degree are likely to be assigned a higher score, $w(x,y) = \frac{\lvert\Gamma(x)\cap\Gamma(y)\rvert}{\min\{\lvert\Gamma(x)\rvert,\lvert\Gamma(y)\rvert\}}$ (Ravasz, Somera, Mongru, Oltvai, & Barabási, 2002)
Hub Depressed Index	Analogous to Hub Index, $w(x,y) = \frac{\lvert\Gamma(x)\cap\Gamma(y)\rvert}{\max\{\lvert\Gamma(x)\rvert,\lvert\Gamma(y)\rvert\}}$ (Ravasz et al., 2002)
Leicht-Holme-Newman	Assigns large weight to pairs that have many common neighbors, normalized by the expected number of common neighbors, $w(x,y) = \frac{\lvert\Gamma(x)\cap\Gamma(y)\rvert}{\lvert\Gamma(x)\rvert\cdot\lvert\Gamma(y)\rvert}$ (Leicht, Holme, & Newman, 2006)
Global Graph Metrics	**Description**
Graph Distance	Length of the shortest path between x and y
Katz	Number of all paths between x and y, exponentially damped by length thereby assigning more weight to shorter paths, $w(x,y) = [(\mathbf{I} - \alpha\mathbf{A})^{-1}]_{xy}$ (Katz, 1953)
Hitting time	Number of steps required for a random walk starting at x to reach y (Brightwell & Winkler, 1990)
Commute Time	Expected number of steps to reach node y when starting from x and then returning back to x, defined as $w(x,y) = L^{+}_{xx} + L^{+}_{yy} - 2L^{+}_{xy}$ where $\mathbf{L}$ is the Laplacian matrix (Göbel & Jagers, 1974)
Rooted PageRank	Similar to Hitting time, but at each step there is some probability that the random walk will reset to the starting node x, $w(x,y) = [(\mathbf{I}-\alpha\mathbf{P})^{-1}]_{xy}$ where $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$ (Page et al., 1999)
SimRank	x and y are similar to the extent that they are joined with similar neighbors, $w(x,y) = \frac{\sum_{u\in\Gamma(x)}\sum_{v\in\Gamma(y)} sim(u,v)}{\lvert\Gamma(x)\rvert\cdot\lvert\Gamma(y)\rvert}$ (Jeh & Widom, 2002)
K-walks	Number of walks of length k from x to y, defined as $w(x,y) = [\mathbf{A}^k]_{xy}$
Meta-Approaches	**Description**
Low-rank Approximation	Compute the rank-k matrix $\mathbf{A}_k$ that best approximates $\mathbf{A}$ (hopefully reducing "noise"), then compute similarity over $\mathbf{A}_k$ using some local or global metric (Eckart & Young, 1936; Golub & Reinsch, 1970)
Unseen Bigrams	Compute initial scores using some local or global metric, then augment the scores $w(x,y)$ using values from $w(z,y)$ for nodes z that are similar to x (Essen & Steinbiss, 1992; Lee, 1999)
Clustering	Compute initial scores using some local or global metric, discard links with the lowest scores, and then re-compute the scores on the modified graph (Johnson, 1967; Hartigan & Wong, 1979)

Table 4: Topology Metrics: Summary of the most common metrics for link prediction. Notation: Let $\Gamma(x)$ be the neighbors of x and $\mathbf{A}$ be the adjacency matrix of G.

3.2.1 Metrics Based on the Local Neighborhood of Nodes

The simplest approaches use only the local neighborhood of nodes in a graph to devise a measure of topology similarity, then use pairwise similarities between nodes to predict the most likely links. As shown in Table 3.2, there are numerous such metrics, often based on the number of neighbors that two nodes share in common, with varying strategies for normalization.

Zhou et al. (2009) compares nine such local similarity measures on six datasets and finds that the simplest link predictor, common neighbors, performs the best overall. They also propose a new metric, RA, that outperforms the initial nine metrics on two of the datasets. This new metric is very similar to the Adamic/Adar metric, but uses a different normalization factor that yields better performance in networks with higher average degree. They also propose a method that uses additional two-hop information to avoid degenerate cases where links are assigned the same similarity score. Their results highlight the importance of selecting the appropriate metrics for specific problems and datasets. In another related investigation, Clauset, Moore, and Newman (2008) evaluate a hierarchical random graph predictor against local topology metrics such as common neighbors, Jaccard's coefficient and the degree product on three types of networks: a metabolic, ecology and a social network. They find that a baseline measure based on shortest paths performs best for the metabolic network, where the relationships are more homogeneous, but that their hierarchical metric performs best when the links create more complex relationships, as in the predator-prey relationships found in the ecology network.

Liu and Lü (2010) proposed a local random-walk algorithm as an efficient alternative to the global random-walk predictors for large networks. This method is evaluated alongside other metrics (i.e., common neighbors, local paths, RA, and a few random-walk variants) and shown to perform better on most of the networks and more efficiently than the global random-walk models.

3.2.2 Metrics Based on the Global Graph Structure

More sophisticated similarity metrics are based on global graph properties, often involving some weighted computation based on the number of paths between a pair of nodes. For instance, the Katz measure (1953) counts the number of paths between a pair of nodes, where shorter paths count more in the computation. Rattigan and Jensen (2005) demonstrated that even this fairly simple metric could be effective for the task of "anomalous link prediction", which is the identification of statistically unlikely links from among the links in the initial graph.

A related measure is the "hitting time" metric, which is the average number of steps required for a random walk starting at node x to reach node y. Gallagher et al. (2008) use such random walks with restart to estimate the similarity between every pair of nodes. They focus on sparsely labeled networks where unlabeled nodes may have only a few labeled nodes to support learning and/or inference in relational classification. The prediction of new links improves the flow of information from labeled to unlabeled nodes, leading to an increase in classification accuracy of up to 15%. Note that adding teleportation probabilities to this random walk approach roughly yields the PageRank algorithm which is said to be at the heart of the Google search engine (Page et al., 1999).

The SimRank metric (Jeh & Widom, 2002) proposes that two nodes x and y are similar if they are linked to neighbors that are similar. Interestingly, they show that this approach is equivalent to a metric based on the time required for two backwards, random walks starting from x and y to arrive at the same node. As with the other approaches based on random walks, this metric could be computed via repeated simulations, but is more efficiently computed via a recursive set-point approach.

3.2.3 Meta-approaches and Supervised Learning Approaches

The metrics above can be modified or combined in multiple ways. Liben-Nowell and Kleinberg (2007) consider several such "meta-approaches" that use some local or global similarity metric as a subroutine. For instance, the metrics discussed above can each be defined in terms of an arbitrary adjacency matrix $\mathbf{A}$. Given this formulation, we can imagine first computing a low-rank approximation $\mathbf{A}_k$ of this matrix using a technique such as singular value decomposition (SVD), and then computing a local or global graph metric using the modified $\mathbf{A}_k$. The idea is that $\mathbf{A}_k$ retains the key structure of the original matrix, but noise has been reduced. Liben-Nowell and Kleinberg also propose two other meta-approaches based on removing spurious links suggested by a first round of similarity computation (the "clustering" approach) or based on augmenting similarity scores for a node x based on the scores for other nodes that are similar to x (the "unseen bigrams" approach). They compare the performance of these three meta-approaches vs. multiple local and global metrics on the task of predicting future links in a social network. The Katz measure and meta-approaches based on clustering and low-rank approximation perform the best on three of the five arXiv datasets, but simple local measures such as common neighbors and Adamic/Adar also perform surprisingly well.

Supervised learning methods can also be used to combine or augment the similarity metrics that we have discussed. For instance, Lichtenwalter et al. (2010) investigate several supervised methods for link prediction in sparsely labeled networks, using many of the metrics from Table 3.2. These metrics are used as features in simple classifiers such as C4.5, J48, and naive Bayes. They find the supervised approach leads to a 30% improvement in AUC over the simple unsupervised link prediction metrics. Similarly, Kashima and Abe (2006) propose a supervised probabilistic model that assumes that a biological network has evolved over time, and uses only topological features to estimate the model parameters. They evaluate the proposed method on protein-protein and metabolic networks and report increased precision compared to simpler metrics such as Adamic/Adar, Preferential Attachment, and Katz.

3.2.4 Discussion

In general, the local topology metrics sacrifice an amount of accuracy for computational gains while the global graph metrics may perform better but are costly to estimate and infeasible on huge networks. Where appropriate, supervised methods that combine multiple local metrics may offer a promising alternative. The next subsection discusses additional work on link prediction that has used supervised methods.

Link prediction using these metrics is especially sensitive to the characteristics of the domain and application. For instance, many networks in biology, where the identification of links is costly, contain missing or incomplete links, while the removal of insignificant links is a more significant issue for social networks. For that reason, researchers have analyzed and proposed many different metrics when working in the domains of web analysis (Kleinberg, 1999; Broder et al., 2000), social network analysis (Zheleva, Getoor, Golbeck, & Kuter, 2010; Xiang et al., 2010; Koren, North, & Volinsky, 2007), citation analysis (Borgman & Furner, 2002), ecology communities (Zhou et al., 2009), biological networks (Jeong et al., 2000), and many others (Barabási & Crandall, 2003; Newman, 2003).

3.3 Hybrid Link Prediction

In this subsection, we examine approaches that perform link prediction using both the attributes and the graph topology. For such approaches, there are two key questions. First, what kinds of features should be used? Second, how is the information from multiple features combined into a single measure or probability to be used for prediction?

We first consider the mix of non-relational and relational features that should be used. As expected, the best features vary based on the domain and specific network. For instance, Taskar et al. (2003) studied link prediction for a network of web pages and found that simple local topology metrics (which they called *transitivity* and *similarity*) were more important than non-relational features based on the words presents in the pages. Similarly, Hasan et al. (2006) found that another topology metric (shortest distance) was the most useful for predicting co-authorship links in a bibliographic network based on DBLP.

If only a single metric/feature, such as "hitting time," will be used for link prediction, then we must ensure that the metric works well for all nodes and yields a consistent ranking. However, if multiple feature values will be combined in some way, then it may be more acceptable to use a wider range of features, especially if a supervised learner will later select or weight the most important features based on the training data. Thus, hybrid systems for link prediction tend to have a more diverse feature set. For instance, Zheleva et al. (2010) propose new features based on combining two different kinds of networks (social and affiliation networks). Features based on the groups and topology are constructed from the combined network and are used along with descriptive non-relational features, yielding an improvement of 15-30% compared to a system without the combined-network features. A second example of more complex features is provided by Ben-Hur and Noble (2005), who design a new pairwise kernel for predicting links between proteins (protein-protein interactions). The pairwise kernel is a tensor-product of two linear kernels on the original feature space, and is especially useful in domains where two nodes might have only a few common features. This approach has also been applied for user preference prediction and recommender systems (Basilico & Hofmann, 2004). Vert and Yamanishi (2005) propose a related approach, where supervised learning is used to create a mapping of the original nodes into a new euclidean space where simple distance metrics can then be used for link prediction.

Given the great diversity of possible features for link prediction, an interesting approach is a system that automatically searches for relevant features to use. For example, Popescul, Popescul, and Ungar (2003a) propose a unique link prediction approach that systematically generates and searches over a space of relational features to learn potential link predictors. They use logistic regression for link prediction and consider the search space covering equi-joins, equality selections, and aggregation operations. In their approach, the model selection algorithm continues to add one feature at a time to the model as long the Bayesian Information Criterion (BIC) score over the training set can be improved. They find that the search algorithm discovers a number of useful topology-based features, such as co-citation and bibliographic coupling, as well as more complex features. However, the complexity of searching a large feature space and avoiding overfitting present challenges.

We next consider the second key question: how should the information from multiple features be combined into a single measure to be used for link prediction? Most prior work has taken a supervised learning approach, where both non-relational and topology-based metrics are used as features that describe each possible link. As with the supervised techniques discussed in Section 3.2, a model is learned from training data which can then be used to predict unseen links.

Most of these supervised approaches apply the classifier separately to each possible link, using a classifier such as a support vector machine, decision tree, or logistic regression (Popescul et al., 2003a; Ben-Hur & Noble, 2005; Hasan et al., 2006). In these approaches, a “flat” feature representation for each link is created, and the prediction made for each possible link is independent of the other predictions.

In contrast, early work on Relational Bayesian Networks (RBNs) (Getoor et al., 2003) and Relational Markov Networks (RMNs) (Taskar et al., 2003) involved a joint inference computation for link prediction, where each prediction could be influenced by nearby link predictions (and sometimes also by newly predicted node labels). Using a webpage network and a social network, Taskar et al. demonstrated that joint inference using belief propagation could improve accuracy compared to the independent inference approach. However, this approach is computationally intensive, and they noted that getting the belief propagation algorithm to converge was a significant problem. A possible solution to this computational challenge is the simpler approach presented by Bilgic, Namata, and Getoor (2007). Their method involved repeatedly predicting labels for each node, predicting links between the nodes using all available features (including predicted labels), then re-predicting the labels with the new links, and so forth. The link prediction was based on an independent inference step using logistic regression, as with the simpler approaches discussed above. However, the repeated application of this step allows the possibility of link feature values changing in between iterations based on the intermediate predictions, thus allowing link predictions to influence each other.

Recently, Backstrom and Leskovec (2011) proposed a novel approach that is supervised, but where the final predictions are based on a random walk rather than directly on the output of some learned classifier. Given a particular target node v in a social network, along with nodes that are known to link to v, they study how to predict which other links from v are likely to arise in the future (or should be recommended). They define a few simple link features based on node profile similarity and messaging behavior, then use these features to estimate initial link weights. They show how to learn these weights (or transition probabilities) in a manner that optimizes the likelihood that a subsequent random walk, starting at v, will arrive at nodes already known to link to v. Because the random walk is thus guided by the links that are already known to exist, they call this process a “supervised random walk.” They argue that this learning process greatly reduces the need to manually specify complex graph-based features, and show that it outperforms other supervised approaches as well as unsupervised approaches such as the Adamic/Adar measure.

A final approach for link prediction is to use some kind of unsupervised dimensionality reduction that yields a new matrix that in some way reveals possible new links. For instance, Hoff, Raftery, and Handcock (2002) propose a latent space approach where the initial link information is projected into a low-dimensional space. Link existence can then be predicted based on the spatial representation of the nodes in the new latent space. These models perform a kind of factorization of the link adjacency matrix and thus are often referred to as matrix factorization techniques. An advantage of such models is that the spatial representation enables simpler visualization and human interpretation. Related approaches have also been proposed for temporal networks (Sarkar & Moore, 2005), for mixed-membership models (Nowicki & Snijders, 2001; Airoldi, Blei, Fienberg, & Xing, 2008), and for situations where the latent vector representing each node is usefully constrained to be binary (Miller, Griffiths, & Jordan, 2009). Typically, these models have the capability of including the attributes as covariates that affect the link prediction but are not directly part of the latent space representation. However, Zhu, Yu, Chi, and Gong (2007) demonstrated how such attributes can also be represented in a related but distinct latent space. More recently, Menon and Elkan (2011) showed how a matrix factorization technique for link prediction can scale to much larger graphs by training with stochastic gradient descent instead of MCMC.

3.4 Discussion

Link prediction remains a challenge, in part because of the very large number of possible links (i.e., N^2 possible links given N observed nodes), and because of widely varying data characteristics. Depending on the domain, the best approach may use only a single non-relational metric or topology metric, or it may use a richer set of features that are evaluated by some learned model. Future work may also wish to consider using an ensemble of link predictors to yield even better accuracy.

Our discussion of link prediction has focused on predicting new links based on existing links and properties of the nodes. In the context of the web, however, "link prediction" has sometimes taken other forms. For instance, Sarukkai (2000) used web server traces to predict the next page that a user will visit, given their recent browsing history. In particular, they use Markov chains, which are related to the random walks discussed in Section 3.2, for this task that they also call "link prediction." More recently, DuBois and Smyth (2010) model relational events (i.e., links) using latent classes where each event/link arises from a latent class and the properties of the event (i.e. sender, receiver, and type) are chosen from distributions over the nodes conditioned on the assigned class. In this work, the local community of a node influences the distribution computed for each node, in a way related to the computations of stochastic block modeling (Airoldi et al., 2008). DuBois & Smyth's task is also a form of link prediction, but where the goal is not to predict the presence or absence of a static link, but the frequency of occurrence for each possible event/link.

One might also be interested in deleting or pruning away noisy, less informative links. For instance, friendship links in Facebook are usually extremely noisy since the cost of adding friendship links is insignificant. Most of the techniques used in this section could also be used to remove existing links wherever the link prediction algorithm yields a very low score (or weight) for an observed link in the original graph.

Indeed, since most link prediction algorithms effectively assign a score to every possible link, they could also be used to assign a weight to *just* the set of initial links in G. This "link weighting" is one of the three subtasks of link interpretation shown in the taxonomy of Figure 2. However, in practice if weights are needed only for the initial links, different features and algorithms will often be possible and/or more effective. The next section discusses such link weighting algorithms, as well as link interpretation in general. Also, in Section 7 we discuss some additional methods for link prediction that seek to jointly transform both nodes and links.

4. Link Interpretation

Link interpretation is the process of constructing weights, labels, or general features for the links. These three tasks of link interpretation are related and somewhat overlapping. First, link weighting is the task of assigning some weight to each link. These weights may represent the relevance or importance of each link, and are typically expressed as continuous values. Thus the weights provide an explicit order over the links. Second, link labeling is similar, except that it usually assigns discrete values to each link. This could represent a positive or negative relationship, or could be used, for instance, to assign one of five topics to email communication flows. Finally, link feature construction is the process of generating a set of discrete or continuous features for the links. For instance, these features might count the frequency of particular words that appeared in messages between the two nodes connected by some link, or simply count the number of such messages.

In a sense, link feature construction subsumes link weighting and labeling, since the weights and labels can be viewed simply as possible link features to be discovered. However, for many tasks it makes sense to compute one particular feature that summarizes the relevance of each link (the weight) and/or one particular feature that summarizes the type of each link (the label). Such weights and labels may be especially useful to later processing, for example with collective classification. Moreover, the techniques used for general feature construction tend toward simpler approaches such as aggregation and discretization, whereas the best techniques for computing weights and labels may involve much more complexity, including global path computations or supervised learning. For this reason, we treat link weighting (Section 4.1) and link labeling (Section 4.2) separately from general link feature construction (Section 4.3).

4.1 Link Weighting

Given the initial graph $G = \langle V, E, \mathbf{X}^V, \mathbf{X}^E \rangle$, the task is to assign a continuous value (the weight) to each existing link in G, representing the importance or influence of that link. As previously discussed, link weighting could potentially be accomplished by applying some link prediction technique and simply retaining the computed scores as link weights. For instance, Lassez, Rossi, and Jeev (2008) perform link prediction and weighting by applying singular value decomposition to the adjacency matrix, then retaining only the k most significant singular-vectors (similar to the low-rank approximation techniques discussed in Section 3.2).

They show that querying (e.g., with PageRank) on the resultant weighted graph can yield more relevant results compared to an unweighted graph.

Unlike with link prediction, however, most link weighting techniques are designed to work only with links that already exist in the graph. These techniques don't work for predicting unseen links because they weight links based on known properties/features of the existing links, or because they compute some additional link features that only yield sensible results for links that already exist.

In the simplest case, link weighting can be just aggregating an intrinsic property of links. For example, Onnela et al. (2007) defines link weights based on the aggregated duration of phone calls between individuals in a mobile communication network. In other cases, simply counting the number of interactions between two nodes may be appropriate.

Thus, when link *features* like duration, direction, or frequency are known, they can be aggregated in some way to generate link weights. If actual link *weights* are already known for some of the links, then supervised methods can be used for weight prediction, using the known weights as training data. For instance, Kahanda and Neville (2009) predict link strength within a Facebook dataset, where stronger relationships are identified based on a user's explicit identification of their "top friends" via a popular Facebook application. Gilbert and Karahalios (2009) also predict link strength for Facebook, but form their training data from survey data collected from 35 participants (yielding strength ratings for about 2000 links). Both of these algorithms generate a large number (50-70) of features about each link in the network, then learn a predictive model via regression or some other technique such as bagged decision trees, which Kahanda and Neville finds performs best among several alternatives. Gilbert and Karahalios generate features based on profile similarity (e.g., do two users have similar education levels?) and based on user interactions (e.g., how frequently and about what topics do two users communicate?). They find the interaction features to be most helpful, especially a feature based on the number of days since the last communication event. Kahanda and Neville use similar kinds of features, which they term *attribute-based* and *transactional* features, and also add *topological* features (such as the Adamic/Adar discussed in Section 3.2) and *network-transactional* (NTR) features. NTR features are those that are based on communications between users (e.g., the number of email messages exchanged) but moderated in some way by the larger network context. This moderation often takes the form of normalization, for instance to dampen the influence of a node that has sent a large number of messages to many different friends. They find that these NTR features are by far the most helpful for prediction, but that many other features also contribute to the overall predictive accuracy.

When training data with sample link weights is not available, approaches based on a parameterized probabilistic model are still possible. However, since candidate link features can no longer be evaluated against the training data, these approaches must (manually) choose the features that they use much more carefully. For instance, Xiang et al. (2010) examine link weight prediction on two social network datasets (Facebook and LinkedIn), but use only 5-11 features for each link. They hypothesize that relationship strength is a hidden cause of user interactions, and propose a link-based latent variable model to capture this dependence. For inference, they use a coordinate ascent optimization procedure to predict the strength of each link. Since the actual strength of each link is not known, prediction tasks in this domain cannot directly evaluate accuracy. However, Xiang et al. demonstrate that using the link strengths produced by their method leads to higher autocorrelation and higher collective classification accuracy when predicting user attributes such as gender or relationship status.

A number of researchers have considered the importance of recency in evaluating link weight, under the assumption that events or interactions that occurred more recently should have more weight. For instance, Roth et al. (2010) propose the "Interactions Rank" metric for weighting a link based on the messages between two nodes. The formula separately weights incoming and outgoing messages for each link, and imposes an exponential decay on the importance of each message based on how old it is. Roth et al. use this metric to weight the links in what they call the "implicit social network," where each node represents a group of users. They demonstrate that this metric can be used to accurately predict users that are missing from an email distribution list. However, the basic metric is simple to compute and could be applied to many other tasks.

The Interactions Rank metric weights a link more heavily if it connects two nodes that have frequently and/or recently communicated. Alternatively, Sharan and Neville (2008) have considered how to weight links in a graph where the links (such as hyperlinks or friendships) may themselves appear or disappear over time. In particular, they construct a summarized graph where all nodes and links that have ever existed in the past are present. Each link in this new graph is weighted based on a kernel function that can provide more weight to links that have been present more often or more recently in the past. They explain how to modify standard relational classifiers to use these weighted links, and demonstrate that a variety of kernels (including exponential and linear decay kernels) produce weighted links that yield higher classification accuracy compared to a non-weighted graph. More recently, Rossi and Neville (2012) have extended this work to handle time-varying attribute values, which may serve as a basis for incorporating temporal dynamics into additional tasks.

4.2 Link Labeling

Given the initial graph $G = \langle V, E, \mathbf{X}^V, \mathbf{X}^E \rangle$, the task is to construct some discrete label for one or more links in G. These labels can be used to describe the type of relationship that each link represents. For instance, in the Facebook example, a link labeling algorithm may create labels representing "work" or "personal" relationships. Such labels would enable subsequent classification models to separately account for the influence of these different kinds of relationships.

Most prior work on link labeling has assumed that some text (such as a message) describes each link, and has been based on unsupervised textual analysis techniques such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003), Latent Semantic Analysis (LSA) (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990), or Probabilistic Latent Semantic Analysis (PLSA) (Hofmann, 1999). Traditionally, these techniques have been used to assign one or more "latent topics" to each document in a collection of documents. The "topics" that are formed are defined implicitly by a probability distribution over how likely each word is to appear, given that the topic is associated with a document. These topics will not always be semantically meaningful, but often manual inspection reveals that most prominent topics do represent sensible concepts such as "advertising" or "government relations." However, even when such semantic associations are not obvious, inferring such topics for a set of links can still aid further analysis, since the topics identify which links represent similar kinds of relationships.

These textual analysis techniques were developed with independent documents in mind, not inter-linked nodes, but they can be adapted to label links in several ways. For instance, Rossi and Neville (2010) examined messages between developers contributing to an open-source software project. They treat each message as a separate document, and use LDA to infer the single most likely latent topic for each message (i.e., a link label). This technique could be used for any graph with textual content associated with the links. Rossi and Neville also go further, to consider the impact of time-varying topics and time-varying topic/word associations, by running multiple iterations of LDA, one per time epoch. Using this model, they study the problem of predicting the effectiveness of different developers (nodes) in the network. They demonstrate that the accuracy of predictions is significantly improved by modeling the temporal evolution of the communication topics.

McCallum, Wang, and Corrada-Emmanuel (2007) describe an alternative way of extending LDA-like approaches for link labeling. LDA is essentially a Bayesian network that models the probabilistic dependencies between documents, associated topics, and words associated with those topics. They propose to extend this model with the Author-Recipient-Topic (ART) model, where the choice of topic for each document (message) depends on both the author and the recipient of the message. Once parameters are learned for the model, inference (e.g., with Gibbs sampling) can be used to infer the most likely latent topics for each message. They make use of these topics to assign roles to people in an email communication network, and demonstrate that it outperforms simpler models.

Supervised techniques can also be used for link labeling. For instance, Taskar et al. (2003) study an academic webpage network and consider how to predict node labels (such as "Student" or "Professor") while simultaneously predicting link labels (such as "adviser-of"). Given a labeled training graph, they learn a complex Relational Markov Network (RMN) that can predict these labels and the existence of new links. To make the link prediction tractable, only some candidate new links are considered, such as those links suggested by a textual reference, inside a page, to some other entity in the graph. The RMN utilizes text-based features, for instance based on the anchor text for known links or the heading for the HTML section in which a possible link reference is found. They demonstrate that the RMN's joint inference over nodes and links improves performance compared to separate inference. However, learning and inference with RMNs can often be a significant challenge, which in practice limits the number and types of feature that can be considered.

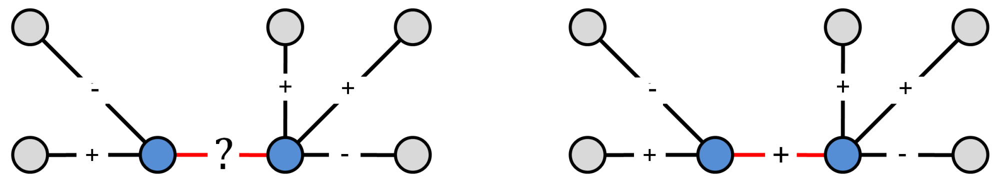
The RMN approach learns from some training data and then uses joint inference over the entire graph. A simpler supervised approach is to create a set of features for each link and use these features for learning and inference with an arbitrary classifier that treats each link separately. Leskovec, Huttenlocher, and Kleinberg (2010) study a particular form of this approach where there are only two link labels, representing a positive or negative relationship (such as friendship vs. animosity). They create link features based on the (signed) degree of the nodes involved in each link and also based on transitivity-like properties computed from the known labels of nearby links. They demonstrate this approach using data from Epinions, Wikipedia, and Slashdot, where users have manually indicated positive or negative relationships to other users. Given a network with almost all edges labeled, the label classifier is able to predict the label (positive or negative) of a single unlabeled edge with high accuracy. Interestingly, they show that a classifier's predictive accuracy for a particular dataset decreases only slightly when the classifier is trained on a different dataset vs. being trained on the same dataset that is used for predictions. They argue that theories of balance and status from social psychology partially explain this ability of their predictive models to generalize across datasets. Unlike most of the other techniques discussed in this section, this work does not make use of text-based features. However, the general problem of predicting the "sign" of a link is related to sentiment analysis (or opinion mining) in natural language processing (Godbole, Srinivasaiah, & Skiena, 2007; Pang & Lee, 2008). These sentiment analysis algorithms could be reformulated to predict the label (such as positive or negative) of a link given its associated text.

Because a link between two nodes can be established based on many different kinds of relationships, there are many other types of algorithms that could potentially be used for labeling links, even if the original algorithm was not designed for this purpose. For instance, Markov Logic Networks (MLNs) have been used to extract semantic networks from text, yielding a graph where the nodes represent objects or concepts (Kok & Domingos, 2008). This process produces relations such as "teaches that" or "is written in" between the nodes, which could be used as link labels in further analysis. Another example is the Group-Topic (GT) model proposed by McCallum, Wang, and Mohanty (2007), which, like the previously mentioned ART model, is a Bayesian network. The model is intended for graphs where two nodes (such as people) become connected when they both participate in the same "event," such as both voting yes for the same political bill. Rather than directly labeling links (like ART), the GT model clusters these nodes (such as people) into latent groups based on textual descriptions of the events/votes. However, the GT model also simultaneously infers a set of likely topics for each event, which could be used to label the implicit links between the nodes. The results of the model could also be used to add new nodes to the graph that represent the latent groups that were discovered.

4.3 Link Feature Construction

Link feature construction is the systematic construction of features on the links, typically for the purpose of improving the accuracy or understandability of SRL algorithms. Link feature construction can be important for many prediction tasks, but has received considerably less attention than node feature construction in the literature. Fortunately, many of the computations that have been developed for node feature construction can also apply to link features. To avoid redundancy, we defer most of our analysis of feature construction to the discussion of node feature construction in Section 6.3. This section briefly discusses how such techniques for node feature construction can be applied to links, then summarizes the major types of link features that can be computed.

Section 6.3 will later describe how feature values for relational data are often based on *aggregating* values from multiple nodes. For instance, such a feature might compute the average or the most common feature value among all of the neighbors of a particular node. Such aggregation-based features help to account for the varying number of neighbors that a node may have. For links, aggregation is less essential, since (usually) each link has precisely



(a) Before link-aggregation (b) After link-aggregation

Figure 4: Link Feature Aggregation Example: The figure demonstrates how an unknown link feature value can be computed by aggregating the link feature values of surrounding links. Here the aggregation operator is Mode.

two endpoint nodes. However, aggregation can still be useful for computing features that collect information from a larger area of the graph. For instance, in Figure 4, a link feature value is being computed for the link in the center of the subgraph (the "target link"). The computation considers the feature values (positive or negative signs) for all of the links that are adjacent to the target link. In this case, the aggregation operator is Mode, and the result is the new link feature value. This example used link features as the input, but node feature values (e.g., of the lightly-shaded nodes in Figure 4) could also be aggregated to form a new link feature. In this way, all of the aggregation operators discussed for nodes in Section 6.3 can also be applied to links.

Figure 5 summarizes the kinds of features that can be constructed for a link. This figure is organized around the sources of information that go into computing a single link feature (i.e., the inputs), rather than the details of the feature computation (such as the type of aggregation or other function used). The bottom of the figure shows the four types of link features, each represented by a subgraph. In each case, the emphasized link at the bottom of the subgraph is the target link for which a new feature value is being computed. Each of the subgraphs shows varying amounts of information because each displays *only* those features, nodes, and/or links that can be used as inputs for that kind of link feature.

The simplest type is the *non-relational link feature*, which can be computed for each link solely from information that is already known about that link. Thus, Figure 5A shows only the feature values which are already known for the target link, which can be used to construct a new feature value. For instance, if a message is associated with each link, then a link feature could count the number of times that a certain word occurs, or the number of distinct words. Alternatively, if a date is associated with the link, then a feature might compute the number of months since the link was formed. Onnela et al. (2007) computed this kind of feature when they aggregated the duration of all phone calls between two people to form a new link feature (which they also used as a link weight).

The remaining feature types are all relational, meaning that they depend in some way on the graph (not just a single link). First, *topology features* (Figure 5B) are those that can be computed using only the topology of the graph. Such a feature might, for instance, compute the total number of links that are adjacent to the target link. Likewise, Kahanda and Neville (2009) computed the clustering coefficient of a pair of linked nodes, which measures the extent to which the two nodes have neighbors in common (Newman, 2003), as well as other topological features such as the Adamic/Adar measure discussed in Section 4.1.

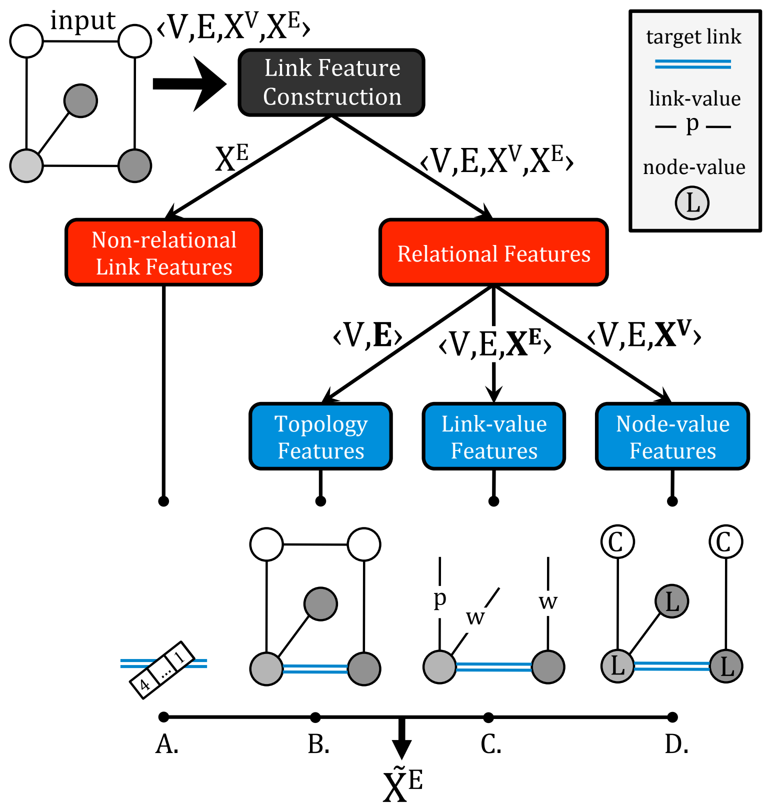


Figure 5: Link Feature Taxonomy: The link feature classes are *non-relational features*, *topology features*, *relational link-value features*, and *relational node-value features*. In the subgraphs at bottom, only the information that is potentially used by that class of link feature (i.e., nodes V, links E, node features X^V, and/or link features X^E) is shown. The emphasized link represents where the feature value is computed (i.e., the "target link").

They used these link features to help predict link strength, but they could also be used for other tasks.

Next, *relational link-value features* are those that are computed using the feature values of nearby links. For instance, Figure 5C shows how link labels of personal (p) or work (w) might be identified from links adjacent to the target link. A new link feature could be formed by representing the distribution of these labels, by taking the most common label, or (when the link features are numeric) by averaging. Leskovec, Huttenlocher, and Kleinberg (2010) used such link-value features when working with graphs where each link had a "sign" feature of positive or negative (as with Figure 4). They computed features based on the signed-degree of the two nodes connected by the target link as well as more complex measures based on other paths between these two nodes (e.g., to measure sign transitivity).

Finally, *relational node-value features* are those that are computed using the feature values of the nodes that are close to or are attached to the target link. For instance, Figure 5D shows how node labels of conservative (C) or liberal (L) might be identified for nodes close to the target link. As with link-value features, these labels could be used to create a new feature value by summarization or aggregation. Often, only the two nodes that are directly attached to the target link are used. For instance, both the work of Gilbert and Karahalios (2009) and Kahanda and Neville (2009) construct link features based on the similarity of two nodes' social network profiles. However, the feature values of more distant nodes could also be used, for instance to compute a new link feature based on how similar the *friends* of two people (nodes) are.

5. Node Prediction

Node transformation includes node prediction (e.g., predicting the existence of new nodes) and node interpretation (e.g., constructing node weights, labels, or features). This section focuses on node prediction, while Section 6 considers node interpretation.

Given a graph with existing nodes V, node prediction can be used in two distinct ways. First, a node prediction algorithm could be used to discover additional nodes that are of the *same* type as those that are already present in V. For instance, given a set of people that communicate via email, a simple algorithm might be used to create new nodes that represent email recipients that are implied by the messages, but not explicitly represented in the original graph. Alternatively, supervised or unsupervised machine learning techniques could be used to discover, for instance, new research papers or people from information available on the web (Craven et al., 2000; Cafarella, Wu, Halevy, Zhang, & Wang, 2008). These techniques are valuable, and can certainly be used to add new nodes to a graph. However, most such work has been examined in the context of general knowledge base construction, rather than relational learning.[4]

We focus on the second type of node prediction, which involves predicting nodes of a *different* type than those that are already present in the graph. These new nodes might represent locations, communities (Kleinberg, 1999), roles (McCallum, Wang, & Corrada-Emmanuel, 2007; Rossi, Gallagher, Neville, & Henderson, 2012), shared characteristics, social processes (Tang & Liu, 2009; Hoff et al., 2002), functions (Letovsky & Kasif, 2003), or some other kind of relationship. For instance, in the running Facebook example, a newly discovered node may represent a common interest or hobby that multiple people share. These nodes are usually referred to as "latent nodes" (and the nodes connected to each such node form a "latent group").[5] The meaning of these nodes will depend upon what features and/or links were included as input to the node prediction algorithm. For instance, including work-based friendships will lead to very different groups than if only personal friendships are considered.

4. The recent work of Kim and Leskovec (2011) is an exception. Their technique uses EM to infer the existence of missing nodes and links based on only the known topology of the graph.
5. Prior work sometimes refers to such nodes as "hidden" nodes, especially when they are thought to represent concrete characteristics, such as geographic location, that could be measured but were, for some reason, not observed in the data.

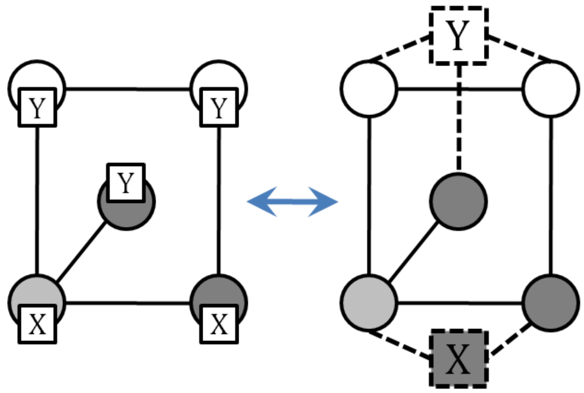


Figure 6: Alternative Representations for Newly Predicted Groups: The left figure shows how a new feature (with value X or Y) could be added to each node, while the right figure demonstrates the creation of two new nodes to represent the groups.

There are many advantages of this type of representation change with regards to accuracy and understandability. For instance, nodes that are not directly connected in the original graph but are similar in some way become, because of the links to the new nodes, closer in graph space. Intuitively, nodes connected to a high level concept should share some latent properties and representing that latent structure can directly impact classification, network analysis, and many other tasks. For instance, reducing the path length between similar nodes enables influence from these nodes to propagate more effectively if collective classification (CC) is performed on these nodes. A model can still learn about and exploit these new nodes and relationships, even if the semantic meaning of the new nodes is not precisely understood.

The most popular methods for predicting new nodes are based on clustering, which in our context means the grouping of nodes such that nodes within a group are more similar to each other than they are to the nodes in other groups. Typically, one new node is created for each group, and then links are added between each existing node and its corresponding group node (see right side of Figure 6). Some techniques may also associate each node with multiple groups, with link weights representing the affinity to each group.

When new groups are discovered, whether via clustering or via some other technique, an alternative to creating new nodes and links is to simply add new feature(s) to each node that represent the group information. The left side of Figure 6 demonstrates this alternative. For instance, a new node feature might represent having running as hobby, or it may simply represent belonging to discovered group #17, which is of unknown meaning. Popescul and Ungar (2004) use the CiteSeer dataset to demonstrate that this technique can derive features that can improve predictive accuracy. An advantage of this approach, as opposed to adding new nodes, is that it potentially enables simpler, non-relational algorithms to make use of the new information. A potential disadvantage, though, is that it also does not allow for algorithms such as CC to propagate influence between newly connected nodes, as discussed above. However, some such methods use this general strategy to generate much larger numbers of latent features that can be used for classification (Tang & Liu, 2009; Menon & Elkan, 2010). Tang & Liu demonstrate that, in some cases, the resultant large number of link-based features may make collective inference unnecessary for obtaining good accuracy. Naturally, whether the information discovered from these clusterings is best represented via new nodes or new features will depend upon the dataset and the inference task. In this section, for simplicity we will discuss each algorithm assuming that new nodes will be created (even if the algorithm was originally described in terms of creating new features).

As with our discussion of link prediction, we organize our discussion around the kinds of information that are used for prediction. Section 5.1 discusses non-relational (attribute-based) node prediction, Section 5.2 discusses topology-based node prediction, and Section 5.3 discusses hybrid approaches that use both the node feature values and the topology of the graph.

5.1 Non-relational (Attribute-Based) Node Prediction

There are many clustering algorithms that can be used to cluster existing nodes using only their non-relational features (attributes), which can then be used to add new nodes to a graph. The two primary types are hierarchical clustering algorithms (e.g., agglomerative or divisive clustering) and partitioning algorithms such as k-means, k-medoids (Berkhin, 2006; Zhu, 2006), EM-based algorithms, and self-organizing maps (Kohonen, 1990). We do not discuss these algorithms further since they have been well studied for non-relational data and can be easily applied to relational data if clustering based only on attribute values is desired.

5.2 Topology-Based Node Prediction

The techniques described in this section link existing nodes to one or more new nodes (i.e., latent groups), based only on the original link structure of the graph. In most cases, finding this grouping depends upon computing some kind of similarity metric between every pair of nodes. Two key questions thus serve to identify these techniques. First, what kind of similarity metric should be used? Second, how should the metric be used to predict groupings? We address each question in turn.

5.2.1 Types of Metrics for Group Prediction

Any type of topology-based link weighting metric (see Table 3.2) could conceivably be used for latent node prediction. A metric will be suitable so long as it produces high values for pairs of nodes that should belong to the same group and lower values for other pairs. For instance, a high value of the Katz metric (see Section 3.2) indicates that two nodes have many short paths between them, and thus may belong to the same group. Metrics representing distance rather than similarity can also be used after negating the metric. For instance, Girvan and Newman (2002) focus on detecting community structure by extending the concept of node-betweenness to links. Intuitively, if a network contains latent groups that are only loosely connected by a few intergroup links, then all shortest paths between different groups must go along these links. These links that connect the different groups are assigned a high link-betweenness value (which corresponds to a *low* similarity value).

The underlying group structure can then trivially be revealed by removing the links with highest betweenness.

This idea of using link-betweenness for relational clustering has been extended in a number of directions. For instance, Newman and Girvan (2004) introduced random-walk betweenness, which is the expected number of times that a random walk between a pair of nodes will pass down a particular link. In addition, Radicchi, Castellano, Cecconi, Loreto, and Parisi (2004) proposed using a link-based clustering coefficient metric. They showed that this metric performs comparably to the original link-betweenness metric of Girvan and Newman, but is much faster because it is a local graph measure instead of a global graph measure.

Zhou (2003) describes a new metric, the "dissimilarity index," which can be computed as follows. For each node i, compute a vector $\mathbf{d_i}$ where each value d_{ij} represents the distance from node i to node j (Zhou measures distance based on the average number of steps needed for a random walk starting at node i to reach node j, but any distance metric could be used). If nodes i and k are very similar, they should have very similar distance vectors. Thus, the dissimilarity index for nodes i and k is defined based on a Euclidean-like distance computation between vectors $\mathbf{d_i}$ and $\mathbf{d_k}$. Zhou demonstrates that this technique outperforms the link-betweenness approach of Girvan & Newman for some random modular networks.

Relatively simple metrics can often lead to useful results. For instance, Ravasz et al. (2002) used a simple clustering coefficient metric to study metabolic networks. Their study reveals that the metabolic networks of forty-three organisms are organized into many small, highly-connected modules. Furthermore, they find that for E. coli, the hidden hierarchical modularity closely overlaps with known metabolic functions.

5.2.2 Using the Metrics for Group Prediction

The simplest techniques for identifying new groups is to perform some kind of hierarchical clustering. For instance, after similarities or weights have been computed for every pair of nodes, all links can be removed from the graph. Next, the weighted links are placed between the nodes one by one, ordered by their weights. The intuition is that varying degrees of clusters are formed as more links are added. In particular, this approach forms a hierarchical tree where the leaves represent the finest granularity of clustering where every node is a separate cluster. As we move up the tree larger clusters are formed, until we reach the top where all the nodes are joined in one large cluster. This type of hierarchical approach was used in the work of Zhou (2003). Girvan and Newman (2002) use a similar strategy, but start instead with the original graph and iteratively remove the less similar links from the graph to reveal the underlying community structure. A challenge with these approaches, as with clustering in general, is to select the appropriate number of final clusters, which corresponds to selecting a level in the clustering tree.

Spectral clustering (Dhillon, 2001; Ng, Jordan, & Weiss, 2001; Kamvar, Klein, & Manning, 2003) can also be used for group identification. Spectral clustering relies upon computing a similarity matrix S that describes all the data points, then transforming the matrix in a way that yields a new matrix U where clustering the rows of U using a simple clustering algorithm (such as k-means) can trivially identify the interesting groups in the data. The matrix transformation has several variants, but involves computing some kind of Laplacian of S, then computing the eigenvectors of the resultant matrix and using those eigenvectors to represent the original data. The motivation for this transformation can be seen as identifying good graph cuts in the original graph (those that yield good separations of highly-connected nodes into groups) or as identifying those nodes that are closely related in terms of random walks; see the work of von Luxburg (2007) for an overview. Spectral clustering was originally applied to non-relational data, but, as with the hierarchical techniques described above, it can be applied to relational data by using link-based metrics for computing the similarity matrix. For instance, Neville and Jensen (2005) use the node adjacency matrix and the spectral clustering technique described by Shi and Malik (2000) to identify latent groups in their graphs. They show that this technique enables simpler inference (since each group can be handled separately), and ultimately yields more accurate classification compared to approaches that ignore the group structure. Tang and Liu (2011) also use spectral clustering on the link graph, but do so in order to create a much larger number of latent features that are then used to learn a supervised classifier. Unlike the latent groups from the work of Neville and Jensen, this technique allows each node to be associated with more than one cluster in the output of the spectral clustering, which Tang & Liu claim leads to improved classification accuracy. Spectral clustering can also be used with more complex similarity metrics, as described in the next subsection.

Techniques borrowed from web search can also be useful for node prediction. For instance, given the adjacency matrix $\mathbf{A}$ for a webpage graph, the Hits algorithm (Kleinberg, 1999) computes the first few eigenvectors of $\mathbf{A}\mathbf{A}^T$ and $\mathbf{A}^T\mathbf{A}$, which represent the most authoritative nodes (the "authorities") as well as prominent nodes that point to them (the "hubs"). Normally, this algorithm is used to find only the single most prominent "community" of authorities and hubs (to assist with a web search), but secondary communities can be discovered by also considering the non-principal eigenvectors of $\mathbf{A}\mathbf{A}^T$ and $\mathbf{A}^T\mathbf{A}$ (Gibson, Kleinberg, & Raghavan, 1998). A node prediction algorithm could then treat each such community as a latent group and add a new node and links to represent this group. These techniques may be especially useful for detecting patterns of influence in a graph and adding more explicit links to represent this influence.

5.3 Hybrid Node Prediction

The techniques in the previous section added new nodes to the graph, often based on clustering, using only the topology of the graph. In principle, a technique that also used the nodes' attributes should produce more meaningful latent groups/nodes. This section considers how to add such attribute information to techniques for node prediction.

A simple approach is to define some kind of similarity metric that combines non-relational and topology-based similarity into a single value, then provide that similarity metric to one of the previously mentioned clustering algorithms. For instance, Neville, Adler, and Jensen (2004) use a weighted combination of attribute and link information

$$S(i,j) = \alpha \cdot \frac{1}{k}\sum_k s_k(i,j) + (1-\alpha)\cdot l$$

as a metric, where $s_k(i,j) = 1$ iff nodes i and j have the same value for the kth attribute, and $l = 1$ iff a link exists between i and j. Here the constant α controls the relative importance of the attributes vs. the links. They use this metric with the NCut spectral clustering technique to add new nodes to the graph, and demonstrate that these additional nodes increase the performance of relational classification. A similar weighted combination of attribute and link-based similarity is used by Bhattacharya and Getoor (2005) for entity resolution.

Attribute-based information can also be incorporated on an ad-hoc basis. For instance, Adibi, Chalupsky, Melz, Valente, et al. (2004) describe a group finding algorithm where an initial seed set of clusters is formed based on a handcrafted set of logical rules, and then these clusters are refined using a probabilistic system based on mutual information. In their system, the logic-based component primarily uses the attributes about each node (person), while the probabilistic system primarily uses the links that describe connections between the people. However, both components make some use of both attributes and links.

A more principled approach is to define some kind of generative model that represents the dependence of the observed attributes and links on some latent group nodes, then use that model to estimate group membership. For instance, Kubica, Moore, Schneider, and Yang (2002) define a generative model where each node belongs to one or more groups, and group members tend to link to each other. In particular, they use a group membership chart to track whether each node belongs to each group, and do a local search over possible states of the chart (using stochastic hill climbing) to try to identify membership changes that would better explain the known data. At each step, maximum likelihood is used to estimate the parameters of the model. They demonstrate the usefulness of their technique on news articles, webpages, and some synthetic data.

Generative models can also be used with more sophisticated inference. For example, Taskar, Segal, and Koller (2001) treat group membership as a latent variable and then uses loopy belief propagation to implicitly perform a clustering of the nodes. Likewise, Mixed Membership Relational Clustering (MMRC) (Long et al., 2007) uses EM variants to estimate group memberships. In particular, it uses a first round of hard clustering (where each object is assigned to exactly one cluster), following by a round of soft clustering where continuous strength values are associated with each membership assignment. Mixed membership stochastic blockmodels (Airoldi et al., 2008) also assign continuous group membership values to each node, but use only topological information (not attributes) for their group assignments and use variational inference techniques with the generative model. Finally, Long, Zhang, Wu, and Yu (2006) demonstrate how node clustering can be performed instead using spectral clustering, and focuses particularly on how to simultaneously cluster multiple types of nodes (e.g., to simultaneously cluster web pages and web users into two distinct sets of groups).

Most group prediction algorithms assume that links are more likely to connect nodes that belong to the same group. An exception is the work of Anthony and desJardins (2007), who also use a generative model where the links and attributes depend on some latent group memberships, but where some types of links are more likely to occur between nodes that do *not* belong to the same group. For instance, they note that if groups in a social network are defined by gender, then a link representing “dating” is more likely to connect two nodes from different groups.

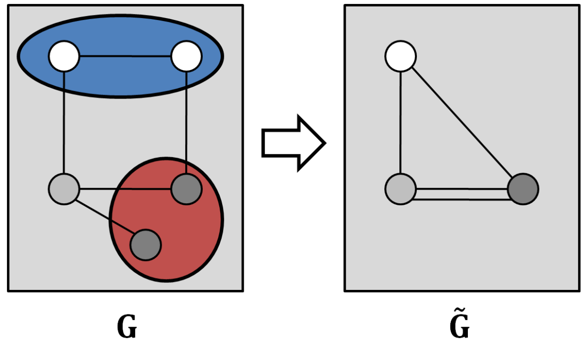


Figure 7: Lifted Graph Representation: The initial graph G is clustered and transformed into a lifted graph representation $\tilde{G}$. The lifted graph representation is created by clustering nodes, links, or both.

5.4 Discussion

Most of the techniques described above produce a single clustering of the nodes, usually based on assigning every node to a single group. In contrast, multi-clustering is an emerging research area that aims to provide multiple orthogonal clusterings of complex data (Strehl & Ghosh, 2003; Topchy, Law, Jain, & Fred, 2004). For instance, individuals in Facebook might be clustered in multiple ways where latent node types might represent friend groups, work relations, socioeconomic status, locations, or family circles. A type of multi-clustering is performed by McCallum, Wang, and Corrada-Emmanuel (2007) where latent nodes are created based on roles and topics. In addition, Kok and Domingos (2007) propose Statistical Predicate Invention (SPI), a node transformation approach based on Markov Logic Networks (Richardson & Domingos, 2006). SPI clusters nodes, features and links forming the basis for the prediction of predicates (or potential nodes). SPI considers multiple relational clusterings based on the observation that multiple distinct clusterings may be necessary to, for instance, group individuals based on their friendships and their work relationships. They demonstrate that MLN inference can estimate these clusters and improves performance compared to two simpler baselines. A similar node prediction approach applies MLNs for role labeling (Riedel & Meza-Ruiz, 2008).

Node *deletion* may also be useful in some cases. For instance, node deletion might be beneficial for removing outdated or spurious nodes from the graph. Alternatively, there may be multiple nodes that represent the same real-world object or concept, in which case deletion for the purposes of entity resolution can be important (Pasula, Marthi, Milch, Russell, & Shpitser, 2003; Bhattacharya & Getoor, 2007; Singla & Domingos, 2006).

Finally, node representation changes can be used to not only to improve accuracy, but also to yield graphs that can be processed more efficiently or that have other desirable properties. Section 5.2 already discussed how Neville and Jensen (2005) used the addition of latent nodes to enable simpler inference. Another possibility is the creation of "super-nodes" that represent more than one of the original nodes. For instance, Figure 7 demonstrates how five original nodes can, after clustering, be collapsed into three super-nodes, yielding a "lifted graph" representation. This kind of representation change can be used for more efficient inference in Markov Logic Networks (see Section 6.3) and for network anonymization (see Section 8.6).

6. Node Interpretation

Node interpretation is the process of constructing weights, labels, or general features for the nodes. As with the symmetric tasks for link interpretation, node weighting seeks to assign a continuous value to each node, representing the node's importance, while node labeling seeks to assign a discrete value to each link, representing the type, group, or class of a node. Likewise, node feature construction is the process of systematically generating general-purpose node features based on, for instance, aggregation, dimensionality reduction, or subgraph patterns.

As discussed in Section 4 for links, node feature construction could be viewed as subsuming node weighting and node labeling, since general feature construction could always be used to construct feature values that are treated as weights or labels for the nodes. In practice, however, the techniques used tend to be rather different. For instance, PageRank is often used for node weighting and supervised classification is often used for node labeling, but these techniques are rarely used for general feature construction. Nonetheless, for node interpretation (more so than with link interpretation) there is some substantial overlap between the techniques actually used for weighting and labeling vs. those used for general feature construction. Below, we first discuss node weighting in Section 6.1 and labeling in Section 6.2. Section 6.3 then discusses node feature construction, mentioning only briefly the relevant techniques that were previously discussed for weighting and labeling.

6.1 Node Weighting

Given the initial graph $G = \langle V, E, \mathbf{X}^V, \mathbf{X}^E \rangle$, the task is to assign a continuous value (the weight) to each existing node in G, representing the importance or influence of that node. Node weighting techniques have been used for information retrieval, search engines, social network analysis, and many other domains as a way to discover the most important nodes with respect to some defined measure. As with node prediction they can be classified based on whether they use only the node attributes, only the graph topology, or both to construct a weighting.

6.1.1 Non-relational (Attribute-Based) Node Weighting

The simplest node weighting techniques use only the *node features* $\mathbf{X}^V$ (i.e., the attributes). For instance, nodes representing documents might be weighted based on the number of query-relevant words they contain, while nodes representing companies might be ranked based on their gross annual sales. Many more sophisticated strategies have also been considered. For instance, Latent Semantic Indexing (Deerwester et al., 1990) can be used to identify the most important semantic concepts in a corpus of text, then nodes can be ranked based on their connection to these concepts. These methods have been extensively applied to quantify or rank the importance of scientific publications (Egghe & Rousseau, 1990). However, because these techniques have been extensively studied elsewhere and also ignore graph structure (such as citations), we do not discuss them further here.

6.1.2 Topology-Based Node Weighting

Several node weighting algorithms that use only the topology of the graph were developed to support early search engines. Examples of this kind of algorithm include PageRank (Page et al., 1999), Hits (Kleinberg, 1999), and SALSA (Lempel & Moran, 2000). Each of these algorithms rank the relative importance of web sites, conceptually based on some kind of eigenvector analysis (Langville & Meyer, 2005), though in practice iterative computation may be used. For instance, PageRank models the web as a Markov Chain and is implemented by systematically computing the principal eigenvector of $\lim_{k\rightarrow\infty} \mathbf{A}^k e$ where $\mathbf{A}$ is the adjacency matrix and e is the unit vector. Hits, as previously described, instead computes the principal eigenvectors of $\mathbf{A}\mathbf{A}^T$ and $\mathbf{A}^T\mathbf{A}$. These algorithms continue to be very important for webpage ranking, but can also be applied to many other kinds of graphs (Kosala & Blockeel, 2000).

In social network analysis, the objective of topology-based node weighting is typically to identify the most influential or significant individuals in a social network. There have been a variety of centrality measures devised that use the local and global network structure to characterize the importance of individuals (Wasserman & Faust, 1994). Examples of these metrics include node degree, clustering coefficient (Watts & Strogatz, 1998), betweenness (Freeman, 1977), closeness (i.e., distance/shortest paths), eigenvector centrality (Bonacich & Lloyd, 2001), and many others (Jackson, 2008; Newman, 2010; Sabidussi, 1966). In addition, White and Smyth (2003) considered how to compute *relative* node rankings, i.e., rankings relative to a set of particularly interesting nodes. They show how to compute such relative rankings both for metrics based on shortest paths as well as for Markov chain-based techniques (e.g., to produce "PageRank with priors"). In addition, some of the similarity metrics described in Table 3.2 can alternatively be formulated for computing weights on nodes.

More recently, node weighting techniques have been extended to measure the relative importance of nodes in temporally-varying data. For instance, both Kossinets, Kleinberg, and Watts (2008) and Tang et al. (2009) define notions of temporal distance based on an analysis of how frequently information is exchanged between nodes. This information can be used to define a range of new graph metrics, such as global temporal efficiency, local temporal efficiency, and the temporal clustering coefficient (Tang et al., 2009). More recently, Tang, Musolesi, Mascolo, Latora, and Nicosia (2010) define notions of temporal betweenness and temporal closeness. They argue that incorporating temporal information with these metrics provides both a better understanding of dynamic processes in the network and more accurately identifies the most important nodes (people). All of these metrics primarily concern networks that have time-varying interactions (e.g., communications between people), but they could also be applied to other types of data with intermittent interactions between nodes or where nodes/link join and leave the network over time. Some of these metrics also apply to links, and could possibly be used to improve link prediction algorithms.

6.1.3 Hybrid Node Weighting

There are also *hybrid* node weighting approaches that use both the attributes and the graph topology (Bharat & Henzinger, 1998; Cohn & Hofmann, 2001). For instance, there are various approaches that modify Hits (Chakrabarti, Dom, Raghavan, et al., 1998; Bharat

& Henzinger, 1998) and PageRank (Haveliwala, 2003) to construct node weights based on both content and links. Topic-Sensitive PageRank (Haveliwala, 2003) seeks to compute a biased set of PageRank vectors using a set of representative topics. Alternatively, Kolda, Bader, and Kenny (2005) propose TOPHITS, a hybrid approach that adds anchor text (i.e., the clickable text on each hyperlink) to the adjacency matrix representation used by Hits. They then use a higher-order analogue of SVD known as Parallel Factors (PARAFAC) decomposition (Harshman, 1970) to identify both the key topics in the graph as well as the most important nodes. Other hybrid approaches have been proposed such as SimRank (Jeh & Widom, 2002), Topical methods (Haveliwala, 2003; Nie, Davison, & Qi, 2006; Kolda & Bader, 2006), Probabilistic HITs (Cohn & Chang, 2000), and many others (Richardson & Domingos, 2002; Lassez et al., 2008). Section 7 discusses further relevant work in the context of joint node and link transformation techniques.

Recently, node weighting approaches have been applied in Adversarial Information Retrieval (AIR) to detect or moderate the influence of spam web sites. Typically, these techniques produce weights using both the topology of the graph and some other information, but not necessarily the kind of attribute information that is used by the techniques discussed above. For instance, TrustRank (Gyongyi, Garcia-Molina, & Pedersen, 2004) is based on PageRank and uses a set of trusted sites evaluated by humans to propagate the trust to other locally reachable sites. On the other hand, SpamRank (Benczúr, Csalogány, Sarlós, & Uher, 2005) measures the amount of undeserved PageRank by analyzing the backlinks of a site. There are other algorithms that try to identify link farms and link spam alliances (Wu & Davison, 2005), given a seed set of known link farm pages. Among these AIR methods, TrustRank is the most widely known but suffers from biases where the human-selected set of trustworthy sites may favor certain communities over others.

6.2 Node Labeling

Given the initial graph $G = \langle V, E, \mathbf{X}^V, \mathbf{X}^E \rangle$, the task is to assign some discrete label for some or all of the nodes in G. We first discuss labeling techniques based on classification, then consider unsupervised textual analysis techniques.

In many cases, node labeling may be considered an end in itself. For instance, in our running Facebook example, the stated goal is to predict the political affiliation of each node where that label is not already known. In other cases, however, node labeling is more properly understood as a representation change that supports the desired task. For instance, for some definitions of anomalous link detection (Rattigan & Jensen, 2005), having estimated node labels would allow us to identify links between nodes whose labels indicate they should rarely, if ever, be connected. Alternatively, for some datasets estimating node labels may enable us to subsequently partition the data based on node type, enabling us to learn more accurate models for each type of node.

Even when node labeling is the final goal, as with our Facebook example, intermediate label estimation may still be useful as a representation change. In particular, Kou and Cohen (2007) describe a "stacked model" for relational classification that relabels the training set with estimated node labels using a non-relational classifier. They then use these estimated labels to learn a new classifier (one that uses both attributes and relational features), and use the new classifier to perform relational classification on the test graph. This approach yields high accuracy, comparable to that of much more complex algorithms for collective classification (CC). Fast and Jensen (2008) analyze this result and discuss how it can be explained by a natural bias in most CC algorithms: training is performed with the given node labels but the inference depends in part on estimated labels (McDowell, Gupta, & Aha, 2009). Stacked models compensate for this bias by instead training with the relabeled (estimated) training set. In addition, inference with the new classifier needs only a single pass over the test graph, yielding much faster inference than CC techniques like Gibbs sampling or belief propagation. More recently, Maes, Peters, Denoyer, and Gallinari (2009) extend these ideas of node relabeling in order to generate a larger training set via multiple simulated iterations of classification. They show that in some cases this approach can outperform stacked models and other CC algorithms like Gibbs sampling.

Thus, there are multiple reasons for creating new labels for the nodes in a graph. This labeling can be accomplished by relational-aware algorithms like those described above as well as by earlier algorithms used for relational or collective classification (Chakrabarti, Dom, & Indyk, 1998; Neville & Jensen, 2000; Taskar et al., 2001; Lu & Getoor, 2003; Macskassy & Provost, 2003). Node labeling can of course also be done by traditional, non-relational algorithms such as SVM, decision trees, kNN, logistic regression, and Naive Bayes, among various others (Lim, Loh, & Shih, 2000; Michie, Spiegelhalter, Taylor, & Campbell, 1994; Burges, 1998; Cristianini & Shawe-Taylor, 2000; Joachims, 1998). These methods simply use features $\tilde{\mathbf{X}}^V$ and do not exploit topology or link-structure.

The above techniques all assign new labels via supervised learning. Labels can also be assigned via unsupervised techniques for textual analysis. There are many networks in the real-world that contain textual content such as social networks, email/communication networks, citation networks, and many others. Traditional textual analysis models such as LSA (Deerwester et al., 1990), PLSA (Hofmann, 1999) and LDA (Blei et al., 2003) can be used to assign each node a topic representing an abstraction of the textual information. More recent techniques such as Link-LDA (Erosheva, Fienberg, & Lafferty, 2004) and Link-PLSA (Cohn & Hofmann, 2001) aim to incorporate the link structure into the traditional techniques in order to more accurately discover a node's type.[6] In particular, the work of Cohn and Hofmann demonstrate that their technique can produce more accurate node labels than techniques that use only the node attributes or only the link topology. There have also been more sophisticated topic models that have been developed for specific tasks such as social tagging (Lu, Hu, Chen, & ran Park, 2010) or temporal data (Huh & Fienberg, 2010; He & Parker, 2010).

6.3 Node Feature Construction

Node feature construction is the systematic construction of features for the nodes, typically for the purpose of improving the accuracy or understandability of SRL algorithms. Feature construction is the most common relational representation change, and is very frequently done before performing a task such as classification. For instance, before performing CC to classify the nodes in our example Facebook political affiliation task, we are likely to compute

6. The names for Link-LDA and Link-PLDA come from the work of Nallapati, Ahmed, Xing, and Cohen (2008), not from the original papers describing the techniques.

some new features representing the information about each node (e.g., age bracket?) and the known information about each node's neighbors (e.g., how many are liberal?).

Different techniques for node feature construction have been described by many previous investigations, though feature construction was not necessarily the focus of many of those investigations. In this section, we summarize and explain the different aspects of feature construction. In particular, Section 6.3.1 presents and discusses a taxonomy of features based on what kinds of *inputs*, such as topology information or link feature values, they use for computing the new feature values. Next, Section 6.3.2 describes the possible *operators*, such as aggregation or discretization, that can be applied to these inputs. Finally, Section 6.3.3 examines how to perform automatic *feature search and selection* to support a desired computational task.

6.3.1 Relational Feature Inputs

A node feature can be categorized according to the types of information that it uses for computing feature values. The possible information to use includes the set of nodes V or links E, the node features $\mathbf{X}^V$, and the link features $\mathbf{X}^E$. Figure 8 shows our taxonomy of node features based on which of these sources of information (the "inputs") they use. This taxonomy is consistent with some distinctions that have been previously made in the literature (e.g., between non-relational and relational features), but to the best of our knowledge this more complete taxonomy has never been previously described. The taxonomy consists of four basic types: non-relational features and three types of relational features (topology features, relational link-value features, and relational node-value features). Below we describe and give examples of each.

- ◇ **Non-relational Features**: A node feature is considered a non-relational feature if the value of the feature for a particular node is computed using only the non-relational features (i.e., attributes) of that node, ignoring any link-based information. For instance, Figure 8A shows a node and the corresponding node's feature vector. A new feature value might be constructed from this vector using some kind of dimensionality reduction, by adding together several feature values, by thresholding a particular value, etc.

- ◇ **Topology Features**: A feature is considered a topology-based feature if values of the feature are computed using only the nodes V and links E, ignoring any existing node and link feature values. For instance, in Figure 8B, a new feature value is being computed for the node in the bottom left of the figure (the "target node"), using only the topological information shown. In particular, the new feature value might count the number of adjacent nodes, or count how many shortest paths in the graph pass through the target node.

- ◇ **Relational Link-value Features**: A feature is considered a relational link-value feature if the feature values of the *links* that are adjacent to the target node are used for computing the new feature. Typically, some kind of aggregation operator is applied to these values, such as count, mode, average, proportion, etc. For instance, in Figure 8C, the values on the links shown represent communication topics (work or personal), and a new link-value feature might compute the mode of these values (p).

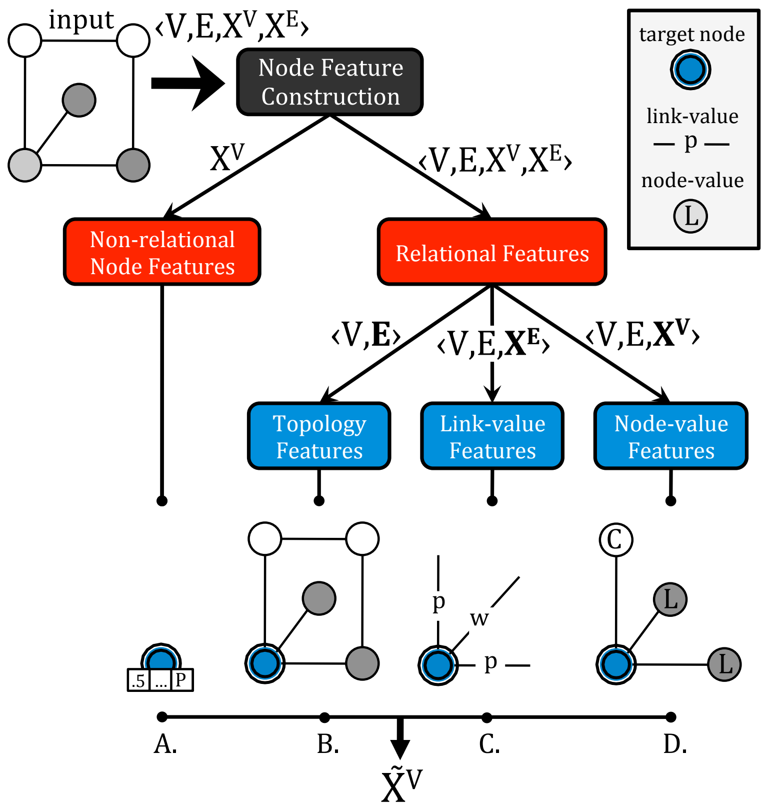


Figure 8: Node Features Taxonomy Based on Inputs Used: The classes of node features are non-relational features, topology features, relational link-value features, and relational node-value features. These classes are defined with respect to the relational information used in the construction of the features (i.e., nodes V, links E, node features $\mathbf{X}^V$, link features $\mathbf{X}^E$). The double-lined "target" node represents where the new feature value is being computed. Parts C and D show only a single feature value for each link or node for simplicity, but in general more than one such feature may exist and be used.

Usually this computation will include only the links directly connected to the target node, but links a few hops away could also be used.

◇ **Relational Node-value Features**: A feature is considered a relational node-value feature if the feature values of nodes linked to the target node are used in the construction. Links are used only for identifying these nodes, although nodes more than one hop away from the target node may also be included. For instance, Figure 8D shows the feature values of adjacent nodes (C or L) which could, for instance, be used to compute a new node-value feature based on the mode (L) of those values. Alternatively, one feature might count the number of adjacent "C" nodes and another might count the number of adjacent "L" nodes.

Feature computation may also be applied recursively. For instance, the ReFeX system (Henderson, Gallagher, Li, Akoglu, Eliassi-Rad, Tong, & Faloutsos, 2011) first computes features for every node based on their degree (a topology-based feature), then considers recursive combinations of these features (such as the mean out-degree of a node's neighbors). Henderson et al. show that such recursive features can often improve classification accuracy for datasets where the network structure is predictive. Alternatively, a topology-based feature such as betweenness might be computed, then a relational node-value feature might compute the average betweenness of the nodes that are neighbors of the target and have a label of "C." This is an example of a hybrid feature that uses both node-value and topology-based information.

Another interesting aspect of relational features is the potential for feature value re-computation. In particular, many techniques for collective classification involve computing a node feature (such as the number of neighbors currently labeled "C") where that feature depends on other feature values that are estimated (e.g., the predicted node labels) and thus may change (Jensen et al., 2004; Sen et al., 2008). In addition, McDowell, Gupta, and Aha (2010) describe features that have a similar need for recomputation, because the "meta-features" they use depend upon the estimated label probabilities for each node in the neighborhood of the target node. In contrast, this kind of feature re-computation has much less applicability for non-relational data, where the nodes are assumed to be independent of each other. However, it can occur with techniques such as semi-supervised learning or co-learning.

6.3.2 Relational Feature Operators

The previous section described features according to the different kinds of *inputs* that they use during feature value computation, whereas this section describes the different *operators* that can be used for this computation. Table 5 summarizes these operators. In some cases, an operator can be used for many different types of relational input. For instance, aggregation operators can be computed using the graph topology, relational node-value inputs, and/or relational link-value inputs, as indicated by the appropriate checkmarks in Table 5. In contrast, path or walk-based operators generally use only the graph topology; for these operators, the lighter colored checkmarks in Table 5 indicate that path/walk-based operators *could* sensibly be used in conjunction with relational link-value or node-values inputs, but this has been rarely if ever done. Below we discuss each of the operators from Table 5 in more detail.

Relational Aggregates: Aggregation refers to a function that returns a single value from a collection of input values such as a set, bag, or list. The most classical statistical aggregation operators are Average, Mode, Exists, Count, Max, Min, and Sum (Neville & Jensen, 2000; Lu & Getoor, 2003). For SRL, another frequent operator is Proportion, which computes, for instance, the fraction of a node's neighbors that meet some criteria such as having the label "C" (McDowell, Gupta, & Aha, 2007). These operators may also be combined with thresholds, e.g., to evaluate whether the Count of a node's neighbors labeled "C" is at least 3. The thresholding turns the numerical aggregate into a Boolean feature, which is needed for tree-based algorithms (Neville, Jensen, Friedland, et al., 2003). Perlich and Provost (2003) describe a set of more complex relational aggregates that depend

		Inputs			
Relational Operators	**Example Techniques**	**Non-relational**	**Topology**	**Relational Link-value**	**Relational Node-value**
Relational aggregates	Mode, Average, Count, Proportion, Degree, ...		✓	✓	✓
Temporal aggregates	Exponential/linear decay, union, ...	✓	✓	✓	✓
Set operators	Union, intersection, multiset, ...	✓		✓	✓
Clique potentials	Direct link cliques, co-citation cliques, triads, ...		✓	✓	✓
Subgraph patterns	Two star, three-star, triangle (i.e., transitivity), ...		✓	✓	✓
Dimensionality reduction	PCA, SVD, Factor Analysis, Principal Factor Analysis, Independent Component Analysis, ...	✓	✓	✓	
Path/walk-based measures	Betweenness, common neighbors, Jaccard's coefficient, Adamic/Adar, shortest paths, random-walks, ...		✓	✓	✓
Textual analysis	LSA, LDA, PLSA, Link-LDA, Link-PLSA, ...	✓		✓	✓
Relational clustering	Spectral partitioning, Hierarchical clustering, Partitioning relocation methods (k-means, k-medoids), ...	✓	✓	✓	✓

Table 5: Relational Feature Operators: Summary of the most popular types of relational feature operators. A check is used to indicate the classes of inputs (see Section 6.3.1) that each operator most naturally uses for constructing feature values, while a lighter check indicates that the operator *could* sensibly be used with that input but that this combination has rarely if ever been used.

on the distribution of attribute values that are associated with each node (e.g., via links or a relational join). For instance, these aggregates may use a function such as the edit distance to compare each node's distribution to a reference distribution computed from the training data. Perlich and Provost demonstrate that these aggregations can in some cases improve performance compared to simpler alternatives. There are also aggregate operators that use only topology-based information. For instance, the operator Degree, which simply counts the number of adjacent links, can be a predictive feature, but should be applied carefully to relational data to avoid bias (Jensen, Neville, & Hay, 2003).

Temporal Aggregates: Relational information might also contain temporal information in the form of timestamps or durations for the links, node, or features. In general, such data can be handled by defining special temporal-aggregation features computed over the raw data (McGovern, Collier, Matthew Gagne, Brown, & Rodger, 2008) or by defining a graph that summarizes all of the temporal information (usually by decreasing the importance of less recent information) (Sharan & Neville, 2008; Rossi & Neville, 2010). Rossi and Neville discuss an example of the latter approach, where they explore the impact of using various temporal-relational information and various kernels for summarization. Alternatively, Section 6.1 discusses how notions of temporal distance can be used to modify path/walk-based metrics such as node betweenness and closeness.

Set Operators: The traditional domain-independent set operators such as set union, intersection, and difference can be applied to construct features (Kohavi & John, 1997). For instance, if there are two attributes that both represent the presence of some word in a page (node), a new feature might represent the case where a page contains both of those words (i.e., feature intersection). For relational data, more complex set-based features are possible. For instance, a feature for collective classification might represent the union of all the class labels of the nodes adjacent to the target node. Neville, Jensen, and Gallagher (2003) propose a more complex approach where the feature value is a multiset that represents the complete distribution of adjacent nodes' labels (e.g., {`3C, 2M, 5L`} to indicate the labels of ten adjacent nodes). Using this feature representation, they show that the "independent-value" approach that assumes that the labels are independently drawn from the same distribution yields the most effective relational classification. Recently, McDowell et al. (2009) showed that, for CC, this "multiset" approach usually outperformed other types of features such as the proportion or count-based aggregates discussed above.

Clique Potentials: Some probabilistic models such as Relational Markov Networks (RMNs) (Taskar et al., 2002) perform inference over related nodes without computing aggregates. Instead, they use clique-specific potential functions to represent the probabilistic dependencies, and a product term in the probability computation naturally expands to accommodate a varying number of neighbors for each node. In one sense, this is a "featureless" approach, since there is no need to choose a relational aggregation function. However, different kinds of dependencies can still be represented by different cliques. For instance, Taskar et al. consider different sets of cliques for webpage classification: one based only on hyperlinks, the other including information based on where links appear within a page. Likewise, later work added additional types of cliques to enable link prediction (Taskar et al., 2003). Thus, even with these models there remain important feature choices to be made.

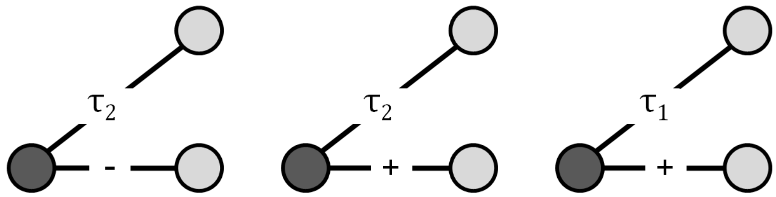


Figure 9: Subgraph Patterns with Link Labels. Each subgraph represents a possible pattern that a particular feature could look for in relation to the target node (the bottom-left node in each case).

Other probabilistic models also use link-based information without computing explicit features, such as the random walk-based classifier of Lin and Cohen (2010) or the weighted-neighbor approach of Macskassy and Provost (2007). Even in these cases, however, choices remain about what types of links to use. For instance, in webpage graphs, "co-citation" links may be more predictive of class labels than direct links (Macskassy & Provost, 2007; McDowell et al., 2009).

Subgraph Patterns: A subgraph pattern feature is one that is based on the existence of a particular pattern in the graph adjacent to the target node. Such a feature might count how many times a particular pattern exists for the target node, or produce a value of `true` if at least one such pattern exists. The simplest such pattern is called *reciprocity*; it is true when the target node i links to node j and j links back to i. In most cases, however, the patterns are more complex and involve more nodes. Robins, Pattison, Kalish, and Lusher (2007) define many such patterns including *two-star* (a node with at least two links), *three-star* (a node with at least three links), and *triangle* (also known as transitivity, where $i \rightarrow j \rightarrow k$ and $i \rightarrow k$). Most such patterns can be defined for both directed and undirected links.

Many other patterns are possible. For instance, Robins, Snijders, Wang, and Handcock (2006) use subgraph patterns for probabilistically modeling graphs. They argue that using more complex patterns such as the *alternating k-triangle* (based on finding k triangles that all share a common side) can help to avoid degeneracy that might otherwise arise during graph generation. Furthermore, subgraph patterns can also be extended to exploit labels on the links and/or nodes. For instance, assume some links are labeled with τ_1 or τ_2 (representing different topics) and some links are labeled with a plus or minus sign (representing positive or negative relationships). Figure 9 demonstrates three possible subgraph patterns, based on different link labelings, relative to the target node shown at the bottom left of each subgraph. A subgraph feature could compute, for each node, the number of matches for one of these patterns, and this feature could be used for later analysis.

Dimensionality Reduction The goal of dimensionality reduction is to find a lower k-dimensional representation of the initial n features (Sarwar, Karypis, Konstan, & Riedl, 2000; Fodor, 2002). More formally, given an initial n-dimensional feature vector $\mathbf{x} = \{x_1, x_2, ..., x_n\}$, find a lower k-dimensional representation $\tilde{\mathbf{x}}$ such that $\tilde{\mathbf{x}} = \{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_k\}$ with $k \leq n$ where the most significant information of the original data is captured, according to some criterion. There are many dimensionality reduction methods such as Principal

Component Analysis (PCA), Principal Factor Analysis (PFA), and Independent Component Analysis (ICA).

Dimensionality reduction techniques can be applied on the adjacency matrix $\mathbf{A}$ of the graph G to create a low-dimensionality graph representation; Section 3.3 explained how this can be used for link prediction. These techniques can also be useful for feature computation. For instance, Bilgic, Mihalkova, and Getoor (2010) investigate active learning to improve the accuracy of collective classification. Their technique involves both non-relational and relational features, but they demonstrate that first applying dimensionality reduction (with PCA) to the non-relational features simplifies learning, leading to substantial gains in accuracy.

Other Operators: We mention only briefly those operators that have already been discussed extensively elsewhere. **Path-based measures** (such as betweenness and distance) and **walk-based measures** (such as PageRank) were discussed in Sections 6.1. These types of measures have been used as features in a classifier to predict links (Lichtenwalter et al., 2010) as well as for validating relational sampling techniques (Leskovec, Chakrabarti, Kleinberg, Faloutsos, & Ghahramani, 2010; Moreno & Neville, 2009; Ahmed, Neville, & Kompella, 2012a, 2012b). These measures typically use only the topology (not the features), but one could easily imagine computing metrics based, for instance, only on paths where each edge had a particular label or type. **Textual analysis** techniques were discussed in Sections 4.2 and 6.2, and **relational clustering** techniques were discussed in Section 5. These operators were used specifically for node/link prediction, weighting, or labeling, but can also be used for more general feature construction.

Finally, there are operators based on **similarity measures**. Similarity between two nodes is often computed, for instance for link prediction (Section 3) or weighting (Section 4.1). Such computations can easily lead to a feature value for a *link*, since the link obviously refers to two endpoint nodes that can be compared. However, for computing a *node* feature value, there is usually no obvious other node for comparison, so similarity measures are not typically used for node feature values. Such measures can, however, be used for node prediction, and Section 5 discusses how in some cases newly discovered nodes/groups can be used to create new node features. As a particular instance of relational similarity functions, graph kernels for structured data (Gärtner, 2003) can also be used. Such kernels can be used either between the nodes of a single graph (Kondor & Lafferty, 2002) or to compute the similarity between two graphs (Vishwanathan, Schraudolph, Kondor, & Borgwardt, 2010). For instance, the former type of kernel is another technique that could also be used for link or group prediction.

Discussion: Many of the feature operators discussed can naturally be used to compute feature values for links in additions to nodes. For instance, textual analysis can be applied to links if there is text associated with each link, and most node-centered path-based measures have analogous formulations for links. One difference is that nodes naturally may link to many other nodes, whereas we assume links with just two endpoints. Thus, relational aggregates such as Count do not initially seem as useful for computing link features. However, Figure 4 previously demonstrated how link-aggregation can be accomplished by broadening the computation to include the multiple links or nodes that are logically connected to each endpoint node of the target link. Naturally, some feature inputs and operators are better suited for computing node features vs. for computing link features. The next section examines how to select the most appropriate features for a given task.

6.3.3 Searching, Evaluating, and Selecting Relational Features

Given the large number of possible features that *could* be used for some task (such as the example Facebook classification task), which features should *actually* be used to learn a model? In some cases, such selection is done manually based on prior experience or trial and error. In many situations, though, more automatic feature selection is desirable. For non-relational data, this has been a widely studied topic in machine learning (Guyon & Elisseeff, 2003; Koller & Sahami, 1996; Yang & Pedersen, 1997; Dash & Liu, 1997; Jain & Zongker, 1997; Pudil, Novovicová, & Kittler, 1994), but selecting *relational* features has received considerably less attention. Given the large number of possible features, efficient strategies for searching over and evaluating the possible features is needed. In this section, we first summarize these two key problems of feature search and feature evaluation, then give examples of how these issues have been resolved in actual SRL systems.

Search: The first step in searching over the relational features is to define the possible relational feature space by specifying the possible raw feature inputs (e.g., node and link feature values) and operators to consider. The possible operators can include *domain-independent operators* (e.g., mode, count) and/or *problem-specific operators* (e.g., count the number of friends divided by the number of groups). Domain-independent operators are obviously more general and easier to apply, while the problem-specific operators can reduce the number of possibilities that must be considered but require more effort and expert knowledge. However, both approaches are vulnerable to selection biases (Jensen et al., 2003; Jensen & Neville, 2002). The second step is to pick an appropriate search strategy, usually either *exhaustive*, *random*, or *guided*. An exhaustive strategy will consider all features that are possible given the specified inputs and operators, while a random strategy will consider only a fraction of this space. A guided strategy will use some heuristic or sub-system to identify the features that should be considered. In all three cases, each feature that is considered is subjected to some evaluation strategy that assesses it usefulness; these strategies are described next.

Evaluation and Selection: Each feature that is considered must be evaluated in some way to determine if it will be retained for use in the final model. For instance, a candidate feature may be evaluated by adding it to the current classification model; if it improves accuracy on a holdout set, then it is immediately (and greedily) added to the set of retained features (Davis, Burnside, Castro Dutra, Page, & Costa, 2005; Davis, Ong, Struyf, Burnside, Page, & Costa, 2007). In other cases, every candidate feature is assigned some score and then only the best scoring feature is retained (Neville, Jensen, Friedland, et al., 2003), or features are added to the model based on decreasing score, so long as the new features continue to improve the model (Mihalkova & Mooney, 2007). Simpler techniques that do not require evaluating the overall model can also be used. For instances, metrics such as correlation or mutual information can be used to estimate how useful the feature is for the desired task. Other metrics or strategies that could be used include Akaike's information criterion (AIC) (Akaike, 1974), Mallows C_p (Mallows, 1973), Bayesian information criterion (BIC) (Hannan & Quinn, 1979; Schwarz, 1978) and many others (Shao, 1996; George &

Proposed System	Search method	Feature Evaluation
RPT (Neville, Jensen, Friedland, et al., 2003)	Exhaustive	Chi-square statistic/p-value
RDN-Boosting (Natarajan, Khot, Kersting, Gutmann, & Shavlik, 2012; Khot, Natarajan, Kersting, & Shavlik, 2011)	Exhaustive	Weighted variance
ReFeX (Henderson et al., 2011)	Exhaustive	Log-binning disagreement
Spatiotemporal RPT (McGovern et al., 2008)	Random	Chi-square statistic/p-value
SAYU (Davis et al., 2005)	Aleph	AUC-PR
nFOIL (Landwehr et al., 2005)	FOIL	Conditional Log-Likelihood
SAYU-VISTA (Davis et al., 2007)	Aleph	AUC-PR
ProbFOIL (De Raedt & Thon, 2010)	FOIL	m-estimate
kFOIL (Landwehr et al., 2010)	FOIL	Kernel target alignment
PRM struct. learning (Getoor, Friedman, Koller, & Taskar, 2001)	Greedy hill-climbing	Bayesian model selection
TSDL (Kok & Domingos, 2005)	Beam search	WPLL
BUSL (Mihalkova & Mooney, 2007)	Template-based	WPLL
PBN Learn-And-Join (Khosravi, Tong Man, Xu, & Bina, 2010)	Level-wise search	Pseudo-likelihood
Discriminative MLN structure learning (Huynh & Mooney, 2008; Biba, Ferilli, & Esposito, 2008)	Aleph++	m-estimate

Table 6: Systems for Searching for and Selecting Node Features: A summary of some of the systems that can be used to automatically search for and select the most appropriate features for a given task. Note that, depending on the context, these papers may be describe their function in terms of learning the best *rules* for a system or of learning the *structure* (e.g., of a MLN). Only some of the MLN-based systems are described; for some of these, WPLL is the "weighted pseudo log-likelihood."

McCulloch, 1993). Frequently, a possible feature may have a particular parameter whose value must be set (such as a threshold); selecting the best value for a given feature can use the same evaluation metrics or may use a simpler estimation technique, e.g., based on maximum likelihood.

Examples: Table 6 summarizes the strategies used by a number of SRL systems that automatically search for features. The columns of the table describe how each system searches for features and how the features are evaluated. For instance, Relational Probability Trees (RPTs) (Neville, Jensen, Friedland, et al., 2003) are an extension of probability estimation trees for relational data that use an *exhaustive* search strategy for feature selection. In particular, RPT learning involves automatically searching over the space of possible features using aggregation functions such as Mode, Average, Count, Proportion, Min, Max, Exists, and Degree. These aggregations can involve node and link feature values (e.g., for Average) or just topology information (e.g., for Degree). These features are used for classification tasks, such as predicting the class label for a document. Each feature is evaluated based on using the chi-square statistic to measure the correlation between the feature and the class label; this yields a feature score and an associated p-value. Features with p-values below the level of statistical significance are discarded, then the remaining feature with the highest score is chosen for inclusion in the model. This selection process has also been extended to use randomization tests to adjust for biases that are common in relational data (Jensen et al., 2003; Jensen & Neville, 2002). RPTs have also been extended for temporal domains (Sharan & Neville, 2008; Rossi & Neville, 2012).

RPTs represent the conditional probability distributions using a single tree. In contrast, Natarajan et al. (2012) propose using gradient boosting (Friedman, 2001) such that each conditional probability distribution is represented as a weighted sum of regression trees grown in a stage-wise optimization. The features for each tree are selected via a depth-limited, exhaustive search, though they note that domain knowledge could also be used to guide this search. Natarajan et al. argue that the resultant set of multiple, relatively shallow trees allows efficient learning of complex structures, and demonstrate that this technique can outperform alternatives based on single trees or the Markov Logic Networks discussed below.

Another system that uses exhaustive search is ReFeX (Henderson et al., 2011), which uses aggregates of Sum and Mean operators to recursively generate features based on the degree of a node and its local neighborhood. To prune the resultant large set, ReFeX uses logarithmic binning of the feature values, clusters features based on their similarity in the binned space, and then retains only one feature from each cluster. The logarithmic binning is chosen because it favors features that are more discriminative for high-degree nodes. This recursive approach has also been modified for constructing features over dynamic networks (Rossi, Gallagher, Neville, & Henderson, 2012).

Alternatively, spatiotemporal RPTs (McGovern et al., 2008) use a *random* search strategy. In particular, these RPTs add temporal and spatial-based features to the set of possible features. The resultant feature space is too large for exhaustive search, so instead random sampling is used. After a pre-defined number of features have been considered, the best scored feature is added to the model.

The remaining systems that we will discuss all use a *guided* search strategy, where some heuristic or sub-system provides candidate features that are considered. For instance, several such systems (Davis et al., 2005; Landwehr et al., 2005) use an ILP system to generate candidate features, then evaluate those features and select some for ultimate use. In particular, SAYU (Davis et al., 2005) uses the ILP system Aleph (Srinivasan, 1999) to generate a candidate feature (which they consider to be a new "view" on the original data). Aleph creates candidates features based on positive examples, from the training data, of the concept which is being predicted. Each proposed feature is evaluated by learning a new model that includes the feature and then computing the area under the precision-recall curve (AUC-PR). If a feature improves the AUC-PR score, it is permanently added to the model and the feature search continues. SAYU-VISTA (Davis et al., 2007) retains this same general approach but extends the types of features that can be considered, in particular adding the ability to dynamically link together objects of different types and to recursively build new features from other constructed features. Davis et al. demonstrate that the link connections are especially helpful in improving performance compared to the original SAYU system. Landwehr et al. (2005) describe the nFOIL system which is very similar to SAYU but was developed independently, while De Raedt and Thon (2010) describe how ProbFOIL upgrades a deterministic rule learner like FOIL to be probabilistic. Landwehr et al. (2010) describe the related kFOIL system which integrates FOIL with kernel methods. They also consider the impact of several different feature scoring functions.

A number of systems have considered how to perform structure learning for Probabilistic Relational Models (PRMs) (Getoor et al., 2001) or for Markov Logic Networks (MLNs) (Domingos & Richardson, 2004), which is a more general case of the feature selection problems described above. For instance, a MLN is a weighted set of first-order formulas; structure learning corresponds to learning these formulas while weight learning corresponds to learning the associated weights. The first MLN structure learning approaches systematically construct candidate clauses by starting from an empty clause, greedily adding literals to it, and testing the resulting clauses fit to the training data using a statistical measure (Kok & Domingos, 2005; Biba et al., 2008). However, these "top-down" approaches are inefficient because the initial proposal of clauses ignores the training data, resulting in a large number of possible features being considered and possible problems with local minima. In response, a number of "bottom-up" approaches have been proposed. In particular, Mihalkova and Mooney (2007) use a propositional Markov network structure learner to construct template networks to guide the construction of features based on the training data. More recent work has examined how to enable bottom-up approaches to learn longer clauses based on constraining the search to only consider features consistent with certain patterns or *motifs* (Kok & Domingos, 2010), or by clustering the input nodes to create a "lifted graph" representation, enabling feature search over a smaller graph (Kok & Domingos, 2009).

Khosravi et al. (2010) perform MLN structure learning by first learning the structure of a simpler Parametrized Bayes Net (PBN) (Poole, 2003), then converting the result into a MLN. For data that contains a significant number of descriptive attributes, they show that this approach dramatically improves the runtime of structure learning and also improves predictive accuracy. Schulte (2011) has given a theoretical justification for this approach. Another alternative, proposed by Khot et al. (2011), is to extend the previously mentioned work of Natarajan et al. (2012) on gradient boosting to MLNs. Essentially, the problem of learning MLNs is transformed into a series of relational regression problems where the functional gradients are represented as clauses or trees. For several datasets they demonstrate faster MLN structure learning that is as accurate or better than baselines including the algorithms of Mihalkova and Mooney (2007) and Kok and Domingos (2010).

The above techniques for MLNs all seek to learn a network structure that best explains the training data as a whole. In contrast, for situations where the prediction of a specific predicate is desired (e.g., to predict the political affiliation in our Facebook example), Huynh and Mooney (2008) and Biba et al. (2008) both propose discriminative approaches to MLN structure learning. For instance, Huynh and Mooney use a modified version of Aleph (Srinivasan, 1999) to compute a large number of candidate clauses, then use a form of L_1-regularization to force the weights that are subsequently learned for these clauses to be zero when the clause is not very helpful for predicting the predicate. This regularization, in conjunction with an appropriate optimization function, effectively leads to selecting a smaller set of features that are useful for the desired task.

Discussion: We focus in this article on graph-based data representations (see Section 1.2). However, many of the examples discussed above use a logical representation instead. We include them in this section because the techniques used for constructing and searching for features or rules are very similar in both settings. For instance, both RPTs (a graph-based approach) and RDN-Boosting (a logical approach) use an exhaustive search over probabilistic decision trees, with different feature scoring strategies.

Popescul et al. (2003a) examine how to automatically learn new relational features for links (to support link prediction), but their techniques could also be applied to constructing node features. In particular, they treat each feature as a relational database query, and use the concept of refinement graphs (Shapiro, 1982) to consider refining an initial query with equi-joins, equality selections, and statistical aggregates. After each refinement, further refinements can be considered; this search is guided by sampling over some possible further refinements and proceeding only if the results of a particular refinement or type seems promising. The features chosen are combined with a logistic regression classifier. For evaluation of the specific features, they use the Bayesian Information Criterion (BIC) (Schwarz, 1978), which includes a term than penalizes feature complexity to reduce the danger of overfitting.

We discussed multiple systems that include notions of aggregation including RPTs, SAYU-VISTA, and the work of Popescul et al. (2003a) discussed above. There are also other aggregate-based learning approaches such as Crossmine (Yin, Han, Yang, & Yu, 2006), CLAMF (Frank, Moser, & Ester, 2007), Multi-relational Decision Trees (MRDTL) (Leiva, Gadia, & Dobbs, 2002), Confidence-based Concept Discovery (C^2D) (Kavurucu, Senkul, & Toroslu, 2008), and many others (Perlich & Provost, 2006; Krogel & Wrobel, 2001; Knobbe, Siebes, & Marseille, 2002). There are also other possibilities for feature evaluation. For instance, GleanerSRL (Goadrich & Shavlik, 2007) uses Aleph (Srinivasan, 1999) to search for clauses and then uses a metric of $precision \times recall$ for evaluating the clauses.

7. Jointly Transforming Nodes and Links

In the previous sections, we primarily discussed relational representation transformation techniques that are applied independently of one another. For instance, one technique might be used to predict links, while another builds on the transformed representation by applying a node labeling technique. This section instead examines "joint" transformation tasks that combine node and link transformation in some way, for instance to label the nodes and weight the links simultaneously. Such techniques may enable each subtask to influence the other in helpful ways, and avoids any bias that might be introduced by requiring the serialization of two tasks (such as link weighting and node labeling) that might usefully be performed jointly.

One recent approach proposed by Namata, Kok, and Getoor (2011) collectively performs link prediction, node labeling, and entity resolution (which can be seen as a form of node deletion/merging). They present an iterative algorithm that solves all three tasks simultaneously by propagating information among solutions to the above three tasks. In particular, they introduce the notion of *inter-relational* features, which are relational features for one task that depend upon the predicted values for another. Their results show that using such features can improve accuracy, and that inferring predicted values for all three tasks simultaneously can significantly improve accuracy compared to performing the three tasks in sequence, even if all possible orderings are considered.

Techniques that model the full distribution across links and attributes such as RMNs (Taskar et al., 2002), PRMs (Friedman et al., 1999), and MLNs (Domingos & Richardson, 2004) can also be used in this scenario, for instance to jointly predict node and link labels. In this section, however, we focus particularly on recent techniques that all presume the existence of some textual content that is associated with the nodes or links of the graph (although the basic algorithms would also work with other kinds of features). We consider three types of techniques, based on what kind of input text they use: stand-alone text documents (e.g., legal memos with no links), text documents connected by links (e.g., webpages with hyperlinks), or entities connected by links that have associated text (e.g., people connected by email messages). Table 7 lists some of the most prominent models, grouped according to these three types. The columns of this table indicate what kinds of input the models use (middle section) and the types of transformation they can perform (right-hand section). The text documents corresponds to node features in this table, while text associated with links yields link features. Below we discuss each of the three types of techniques in more detail.

7.1 Using Text Documents with No Links

First, many techniques can be used to assign topics or labels to the nodes when those nodes (such as documents) have associated text. For instance, the first row of Table 7 indicates that LDA and PLSA use only the nodes and node features and can perform node prediction, weighting, and labeling. Section 6 already mentioned how these techniques can be used to label each node with one or more discovered topics, which is their more typical use. However, these techniques can also perform node weighting (using the weights associated with the topics) and/or node prediction (by converting the discovered topics to new latent nodes as discussed in the introduction to Section 5). In Table 7, we use lighter checkmarks to represent these kind of situations where a transformation task *could* be performed by a particular model but is not its primary use/output.

LDA and PLSA treat each document as a bag of words and seek to assign one or more topics (labels) to each document based on the words. In contrast, Nubbi (Chang, Boyd-Graber, & Blei, 2009) designs an approach based on LDA where a graph is defined based on objects (nodes) that are referenced in a set of documents, then links are predicted based on the relationships that are implied in the text of the documents. In addition, the nodes and links are associated with their most likely topic(s) based on these relationships. Thus, this model simultaneously performs link prediction, link labeling, and node labeling. A similar

	Input				Links			Nodes		
					Link Prediction	**Link Weighting**	**Link Labeling**	**Node Prediction**	**Node Weighting**	**Node Labeling**
Joint Transformation Model	E	$\mathbf{X^E}$	V	$\mathbf{X^V}$	$\tilde{E}$	$\tilde{\mathbf{X}}^{\mathbf{E}}$	$\tilde{\mathbf{X}}^{\mathbf{E}}$	$\tilde{V}$	$\tilde{\mathbf{X}}^{\mathbf{V}}$	$\tilde{\mathbf{X}}^{\mathbf{V}}$
LDA/PLSA			✓	✓				✓	✓	✓
Nubbi			✓	✓	✓		✓	✓	✓	✓
Link-LDA, Link-PLSA	✓		✓	✓	✓	✓		✓	✓	✓
Pairwise-Link-LDA	✓		✓	✓	✓	✓		✓	✓	✓
Link-PLSA-LDA	✓		✓	✓	✓	✓		✓	✓	✓
Relational Topic Model (RTM)	✓		✓	✓	✓	✓		✓	✓	✓
Topic-Link LDA	✓		✓	✓	✓	✓		✓	✓	✓
Group-Topic (GT)	✓	✓	✓			✓	✓	✓	✓	✓
Author-Recipient-Topic (ART)	✓	✓	✓			✓	✓			✓
Block-LDA	✓	✓	✓		✓	✓		✓	✓	✓

Table 7: Summary of the Joint Transformation Models: The middle section of the table indicates what types of graph features are used as inputs to the model, while the right side of the table indicates what types of link or node transformation can be performed by the model. Lighter checkmarks indicate that the output of the model can be transformed to perform a particular transformation task (e.g., to use the node labels to create new latent group nodes), but where that task was not the primary goal of the specified model.

result is produced by the semantic network extraction of Kok and Domingos (2008) that was discussed in Section 4.2.

7.2 Using Text Document with Links

The second type of joint transformation also uses text documents, but adds known links between the documents to the model. For instance, Section 6 discussed how Link-LDA and Link-PLSA add link modeling to LDA and PLSA in order to perform node labeling; as discussed above for LDA and PLSA this can be modified to also achieve node prediction and weighting. As shown in Table 7, Link-LDA and Link-PLSA can also be used for link prediction and weighting by learning a model from a training graph and then using it to predict unseen links on a new test graph (Nallapati et al., 2008).

Link-LDA and Link-PLSA model links in a way that is very similar to how they model the presence of words in a document (node). For instance, in Link LDA's generative model, to generate one word, each document chooses a topic, then chooses a word from a topic-specific multinomial. The identical process (using a topic-specific multinomial) is used to generate, for a particular document, one target document to link to. Thus, Link-LDA and Link-PLSA directly extend the original LDA and PLSA models to add links.

Nallapati et al. (2008) argue that Link-LDA's and Link-PLSA's extensions for links, while pragmatic, do not adequately capture the topical relationship between two documents that are linked together. Instead, they propose two alternatives. The first, Pairwise Link-LDA, replaces the link model of Link-LDA with a model based on mixed membership stochastic blockmodels (Airoldi et al., 2008), where each possible link is modeled as a Bernoulli variable that is conditioned on a topic chosen based on the topic distributions of each of the two endpoints of the link. The second approach, Link-PLSA-LDA, retains the link generation model of Link-LDA, but changes the word generation model for some of the documents (the ones with incoming links) so that the words in such a document depend on the topics of other documents that link to it. The downside of this latter approach is that it only works when the nodes can be divided into a set with only outgoing links and a set with only incoming links. However, Nallapati et al. argue that this limitation can be largely overcome by duplicating any nodes that have both incoming and outgoing links. Moreover, this approach is much faster and more scalable than Pairwise Link-LDA. Nallapati et al. demonstrate that both models outperform Link-LDA on a likelihood ranking task, and that Link-PLSA-LDA also outperforms Link-LDA on a link prediction task. They also show that Link-PLSA-LDA and Link-LDA were comparable in terms of execution time, but that Pairwise Link-LDA was much slower.

Changes to the generative model used by each of these approaches encode different assumptions about the data and can lead to significant performance differences. For instance, Chang and Blei (2009) introduce the Relational Topic Model (RTM) and compare it to the Pairwise Link-LDA model discussed above. Both models allow similar flexibility in terms of how links are defined, but Chang and Blei argue that their model forces the same topic assignments that are used to generate the words in the documents to also generate the links, which is not true of Pairwise Link-LDA. They then demonstrate that RTM provides more accurate predictions and link suggestions than Pairwise Link-LDA and several other baselines.

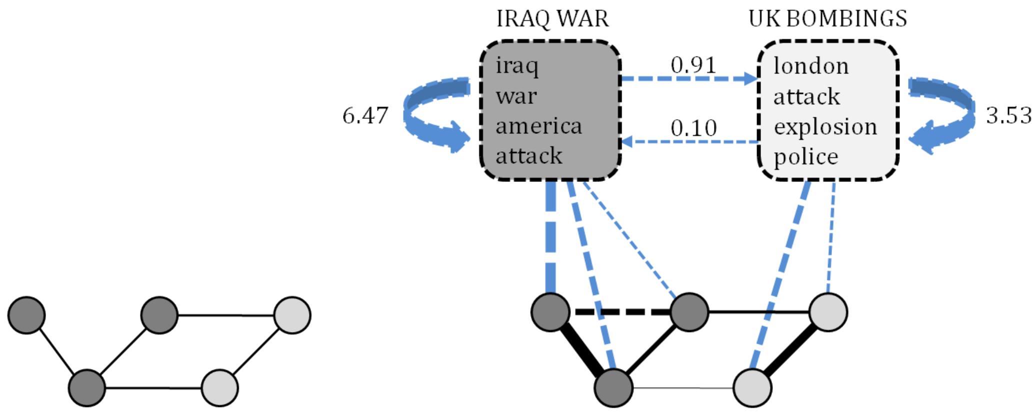
Another possible change to the model is to add other types of objects. For instance, Topic-Link LDA (Liu, Niculescu-Mizil, & Gryc, 2009) models not only documents, links, and the most likely topics associated with each document, but also explicitly considers the author of each document and clusters these authors into multiple "communities." Creating this new clustering is not equivalent to finding per-document topics because each author is associated with more than one document. They argue that this approach is analogous to unifying the separate tasks of (1) assigning topics to documents and (2) analyzing the social network of authors. They show that their approach can in some cases outperform LDA and Link-LDA.

7.3 Using Text Associated with Links

The final type of joint transformation techniques form link features based on text associated with links, such as the text of email messages (McCallum, Wang, & Corrada-Emmanuel, 2007) or scientific abstracts that relate to a particular protein-protein interaction (Balasubramanyan & Cohen, 2011). Several such techniques were discussed previously in the context of link interpretation. For instance, Section 4.2 discussed how models such as the Author-Recipient-Topic (ART) model (McCallum, Wang, & Corrada-Emmanuel, 2007) and the Group-Topic (GT) model (McCallum, Wang, & Mohanty, 2007) extend LDA to perform link labeling; the strength of these predicted labels (topics) can also be used to weight the links. In addition, the GT model directly assigns nodes to groups (i.e., node labeling), while the labels that ART associates with each link could also be used to label the associated nodes. The RART model (McCallum, Wang, & Corrada-Emmanuel, 2007) extends ART by allowing a node to have multiple roles. More recently, Block-LDA (Balasubramanyan & Cohen, 2011) merges the ideas from these latent variables models with stochastic blockmodels. More specifically, the Block-LDA shares information through three components: the link model shares information with a block structure which is then shared by the topic model. Unlike GT and ART, however, Block-LDA focuses on labeling the nodes rather than the links. Balasubramanyan and Cohen evaluate Block-LDA on a protein dataset and the Enron email corpus and demonstrate that it outperforms Link-LDA and several other baselines on the task of protein functional category prediction.

7.4 Discussion

Most of the techniques discussed above are variants of latent group models that focus on node and/or link label prediction, but they can also be used for node prediction where the new nodes represent newly discovered topics or latent groups. These models have also been extended to incorporate notions of time (Dietz, Bickel, & Scheffer, 2007; Wang, Blei, & Heckerman, 2008; Wang & McCallum, 2006), topic hierarchies (Li & McCallum, 2006), and correlations between topics (Blei & Lafferty, 2007). In addition, links are usually assumed to be generated based on the overall topic(s) of a node or link. In contrast, the Latent Topic Hypertext Model (LTHM) (Gruber, Rosen-Zvi, & Weiss, 2008) models each link as originating from some specific word in a document. Somewhat surprisingly, they show that this approach leads to a model with fewer parameters than models like Link-LDA, and demonstrate that their approach outperforms both Link-LDA and Link-PLSA when evaluated on a link prediction task.



(a) Initial Graph (b) Joint Transformation

Figure 10: Example of Joint Transformation: In this example, new latent nodes are added to represent discovered topics, and weighted links are added from each original node to a new latent node. In addition, weighted links are added between the latent nodes, representing connection strength between these topics. Finally, new links between the original nodes may be also be predicted. Note this example is adapted from results found in the work of Nallapati et al. (2008).

If new nodes are added to the graph to represent discovered topics, then links are invariably added to connect existing nodes to the new nodes. However, some models may also learn information about how the discovered topics are related to each other. For instance, Figure 10 shows how two new topics are discovered in a graph and how they are connected to the existing nodes. In addition, the topics are connected to each other with new links where the weight of each link represents how frequently a document from that topic cites a document representing a different topic. Adding these additional links to the graph lets the original nodes be connected more closely not only to their primary topics but also to related topics.

8. Discussion and Challenges

In this section we discuss additional issues that are related to relational representation transformation and highlight important challenges for future work.

8.1 Guiding and Evaluating Representation Transformation

The goal of representation transformation is often to "improve" the data representation in some way that leads to better results for a subsequent task or possibly to a more understandable representation. How can we evaluate whether a particular transformation technique has accomplished this goal? We first address this question, then consider when the final goal can be used to more directly guide the initial transformation.

For some tasks, representation evaluation is straightforward provided that ground truth values are known for a hold-out data set. For instance, to test if a technique for link prediction is effective, accuracy can be measured for links predicted for the hold-out set (Taskar et al., 2003; Liu et al., 2009). The particular evaluation metric can be modified as appropriate for the domain. For instance, Chang and Blei (2009) evaluate the precision of the twenty highest-ranked links suggested for each document, while Nallapati et al. (2008) consider a custom metric called "RKL" that measures the rank of the last true link suggested by the model. Likewise, if the desired task involves classification, then a classification algorithm can be run on the hold-out data, with and without the representation change, to see if the change increases classification accuracy.

In other cases, it may be difficult to directly measure how well a representation change has performed, but classification can be used as a surrogate measure: if accuracy increases, the change is assumed to be beneficial. For instance, classification has been used to evaluate link prediction (Gallagher et al., 2008), link weighting (Xiang et al., 2010), link labeling (Rossi & Neville, 2010; Macskassy, 2007), and node prediction (Neville & Jensen, 2005). In addition, node labeling is naturally a classification problem, while node weighting is usually evaluated in other ways, e.g., based on query relevance.

Other techniques can be used when direct evaluation is not feasible, but there exists some other metric that is believed to be related. For instance, higher autocorrelation in a graph can be associated with the presence of more sensible links, and algorithms such as collective classification typically perform better when the level of autocorrelation is higher. Thus, Xiang et al. (2010) demonstrate the success of their technique for estimating relationship strengths (link weights) based in part on showing an increase in autocorrelation when measured for several attributes in a social network. Likewise, increased information gain for some of the attributes could be used to demonstrate an improved representation (Lippi, Jaeger, Frasconi, & Passerini, 2009), or link perplexity could be used to assess topic labelings (Balasubramanyan & Cohen, 2011). Naturally, the most appropriate evaluation techniques vary based upon the task, and a comparison of transformation techniques may yield different results depending upon what metric is chosen.

Ideally, representation transformation would be guided more directly by the final goal as it is executed, rather than only being evaluated when the transformation is complete. This is often the case for the feature selection and structure learning algorithms discussed in Section 6.3: task accuracy (or a surrogate measure) is evaluated with a particular feature added, and it is retained if accuracy has improved. In other cases, the transformation is even more directly specified by the desired end goal. For instance, the "supervised random walk" approach discussed in Section 3.3 uses a gradient descent method to obtain new link weights such that links predicted by a subsequent random walk (their final goal) will be more accurate. Likewise, Menon and Elkan (2010) show how to add supervision to methods for generating latent features (see introduction to Section 5) so that the features learned would be more relevant to their final classification task. They show, however, that adding such supervision is not always helpful. As a final example, Shi, Li, and Yu (2011) use a quadratic program to optimize a linear combination of link weights such that the final link weights will lead directly to more accurate classification via a label propagation algorithm.

In general, ensuring that a particular transformation will improve performance on the final SRL task remains challenging. Many transformations cannot be directly guided by the final goal, either because suitable supervised data is not available, or because it is not clear how to modify the transformation algorithms to use such information (e.g., with the latent topic models of Section 7 or the group detection algorithms of Section 5).

8.2 Causal Discovery

Causal discovery refers to identifying cause-and-effect relationships (i.e., smoking causes cancer) from either online experimentation (Aral & Walker, 2010) or from observational data. The challenge is to distinguish true causal relationships from mere statistical correlations. One approach is to use quasi-experimental designs (QEDs), which take advantage of circumstances in non-experimental data to identify situations that provide the equivalent of experimental control and randomization. Jensen, Fast, Taylor, and Maier (2008) propose a system to discover knowledge by applying QEDs that were discovered automatically. More recently, Oktay, Taylor, and Jensen (2010) apply three different QEDs to demonstrate how one can gain causal understanding of a social media system. There is also another causal discovery technique for linear models proposed by Wang and Chan (2010). The challenge remains of how to extend these techniques to apply to a broader range of relational data.

8.3 Subgraph Transformation and Graph Generation

The majority of this article focused on transformation tasks centered around the nodes or links of the graphs. However, there are also useful tasks for *subgraph transformation* which seek to identify frequent/informative substructures in a set of graphs or to create features or classify such subgraphs (Inokuchi, Washio, & Motoda, 2000; Deshpande, Kuramochi, Wale, & Karypis, 2005). For instance, Kong and Yu (2010) consider how to use semi-supervised techniques to perform feature selection for subgraph classification given only a few labeled subgraphs. As with nodes and links, for subgraphs the tasks of prediction, labeling, weighting, and feature generation can all be described. Many of the techniques that we described for node-centered features can also be used in this context, but a full discussion of subgraph transformation is beyond the scope of this article.

Recently, graph generation algorithms have attracted significant interest. These algorithms use some model to represent a family of graphs, and present a way to generate multiple samples from this family. Two prominent models are Kronecker Product Graph Models (KPGMs) (Leskovec, Chakrabarti, et al., 2010) and those based on *preferential attachment* (Price, 1976; Barabási & Albert, 1999). These graph generation methods take advantage of global (with KPGMs) and local (with preferential attachment models) graph properties to generate a distribution of graphs that can potentially include attributes. Sampling from these models can be useful for creating more robust algorithms, for instance by training a classifier on a family of related graphs instead of on a single graph. Newman (2003) surveys additional network models and properties that are relevant to graph generation.

8.4 Model Representation

In SRL there is also the notion of model representation: what kind of statistical model is learned to represent the relationship between the nodes, links, and their features? Some of the most prominent models for SRL are Probabilistic Relational Models (PRMs) (Friedman et al., 1999), Relational Markov Networks (RMNs) (Taskar et al., 2002), Relational Dependency Networks (RDNs) (Neville & Jensen, 2007), Structural Logistic Regression (Popescul et al., 2003b), Conditional Random Fields (CRFs) (Lafferty, McCallum, & Pereira, 2001), and Markov Logic Networks (MLNs) (Domingos & Richardson, 2004; Richardson & Domingos, 2006); full discussion of these models is beyond the scope of this article. In many cases techniques for relational representation transformation, such as link prediction, can be performed regardless of what kind of statistical model will be subsequently used. However, the choice of statistical model does strongly interact with what kinds of node and link features are useful (or even possible to use); Section 6.3 describes some of these connections. While a number of relevant comparisons have already been published (Jensen et al., 2004; Neville & Jensen, 2007; Macskassy & Provost, 2007; Sen et al., 2008; McDowell et al., 2009; Crane & McDowell, 2011), more work is needed to evaluate the interaction between the choice of statistical model and feature selection, and to evaluate which statistical models work best in domains with certain characteristics.

8.5 Temporal and Spatial Representation Transformation

Where appropriate, we have already discussed multiple techniques that can incorporate temporal information from graph data (see especially Sections 4.2, 6.1, and 6.3). These techniques focused on solving particular problems such as node classification, but dealing with such data invariably requires studying how to represent the time-varying elements. However, more work is needed to examine the general tradeoffs involved with different temporal representations. For instance, Hill, Agarwal, Bell, and Volinsky (2006) provide a generic framework for modeling any temporal dynamic network where the central goal is to build an approximate representation that satisfies pre-specified objectives. They focus on summarization (representing historical behavior between two nodes in a concise manner), simplification (removing noise from both edges and nodes, spurious transactions, or stale relationships), efficiency (supporting fast analysis and updating), and predictive performance (optimizing the representation to maximize predictive performance). This work provides a number of useful building blocks, but more comparisons are needed to, for instance, evaluate the merits of using summarized networks with general-purpose algorithms vs. using more specialized algorithms with data that maintains the temporal distinctions.

Temporal data is one particular kind of data that can be represented as a relational sequence. Kersting, De Raedt, Gutmann, Karwath, and Landwehr (2008) survey the area of *relational sequence learning* and explains multiple tasks related to such data, such as sequence mining and alignment. These tasks often involve the need to identify relevant features or structure, such as identifying frequent patterns or useful similarity functions. Thus, the set of useful techniques for feature construction and search in this domain overlap with those discussed in Section 6.3.

8.6 Privacy Preserving Representation

There is sometimes a desire to make private graph-based data publicly available (e.g., to support research or public policy) in a way that preserves the privacy of the individuals described by the data. The goal of *privacy preserving representation* is to transform the data in a way that minimizes information loss while maximizing anonymization, e.g., to prevent individuals in the anonymized network from being identified. Naive approaches to anonymization operate by simply replacing an individual's name (or other attributes) with arbitrary and meaningless unique identifiers. However, in social networks there are many adversarial methods through which the true identity of a user can often be discovered from such an anonymized network. In particular, the adversarial methods can use the network structure and/or remaining attributes to discover the identities of users within the network (Liu & Terzi, 2008; Zhou, Pei, & Luk, 2008; Narayanan & Shmatikov, 2009).

An early approach by Zheleva and Getoor (2007) examines how a graph may be modified to prevent sensitive relationships (a particular kind of labeled link) from being disclosed. They describe their approach in terms of node anonymization and edge anonymization. Node anonymization clusters the nodes into m equivalence classes based on node attributes only, while most of the edge anonymization approaches are based on cleverly removing sensitive edges. Backstrom, Dwork, and Kleinberg (2007) address a related family of attacks where an adversary is able to learn whether an edge exists between targeted pairs of nodes.

More recently, Hay, Miklau, Jensen, Towsley, and Weis (2008) study privacy issues in graphs that contain no attributes. Their goal is to prevent "structural re-identification" (i.e., identity reconstruction using graph topology information) by anonymizing a graph via creating an aggregate network model that allows for samples to be drawn from the model. The approach generalizes a graph by partitioning the nodes and then summarizing the graph at the partition level. This approach differs from the other approaches described above because it drastically changes the representation as opposed to making more incremental changes. However, this method enforces privacy while still preserving enough of the network properties to allow for a wide variety of network analyses to be performed.

In each of these investigations the key factors are the information available in the graph, the resources of the attacker, and the type of attacks that must be defended against. In addition, if an attacker can possibly obtain additional information related to the graph from other sources, then the challenges are even more difficult. More work is needed to provide strong privacy guarantees while still enabling partial public release of graph-based information.

9. Conclusion

Given the increasing prevalence and importance of relational data, this article has surveyed some of the most significant current issues in relational representation transformation. After presenting a new taxonomy of important transformation tasks in Section 2, we next discussed the four primary tasks of link prediction, link interpretation, node prediction, and node interpretation. Section 7 considered how some of these tasks can be accomplished simultaneously via techniques for joint transformation. Finally, Section 8 considered how to perform representation evaluation and key challenges for future work.

There are additional possible representation transformations that we have not had space to discuss, or that do not fit cleanly in the taxonomy of Figure 2. For instance, in a bipartite graph of customers and products, it may be useful to eliminate all product nodes, replacing their information content with new links among the customers that purchased the same product. This is somewhat related to the group discovery techniques of Section 5. We have also not considered in any depth the potential for transforming nodes into edges or vice versa (though the representation choices of Figure 6 are also relevant here), and this technique can sometimes be a useful pre-processing step.

The taxonomy presented in Section 2 highlighted the symmetry between the possible transformation tasks for links and those for nodes. This symmetry helped to organize this survey, and also suggests areas where techniques developed for one of these entities can be used for an analogous task with the other. For instance, Liben-Nowell and Kleinberg (2007) reformulated traditional node weighting algorithms to weight links. Likewise, topic discovery techniques based on LDA can be used both for node labeling and for link labeling. Finally, many of the techniques used to create node features can also be used to create link features, and vice versa, although node features have been studied much more thoroughly.

As discussed in Section 8, there remains much work to do. For instance, link prediction remains a very difficult problem, especially for the general case where any two arbitrary nodes might be connected together. Even more significantly, while we have described a wide range of techniques that can address each of the transformation tasks, at the end of the day the practitioner is left with a wide range of choices without many guarantees about what might work best. For instance, node weighting may improve classification accuracy for one dataset but decrease it on another. This challenge is made all the more difficult because the techniques that we have described come from a wide range of areas, including graph theory, social network analysis, numerical linear algebra (e.g., matrix factorization), metric learning, information theory, information retrieval, inductive logic programming, statistical relational learning, and probabilistic graphical models. While the breadth of techniques relevant to relational transformation is a wonderful resource, it also means that evaluating the representation change techniques that are relevant to a particular task is a time-consuming, technically challenging, and incomplete process. Therefore, much more work is needed to establish a theoretical understanding of how different representation changes affect the data, how different data characteristics interact with this process, and how the combination of these techniques and the data characteristics affect the final results of an analysis with relational data.

Acknowledgments

We thank all the reviewers for many helpful suggestions and feedback. The majority of this work was completed at the Naval Research Laboratory, where Ryan Rossi was supported by an ASEE/ONR NREIP summer internship in 2010 and by a NSF Graduate Research Fellowship (at Purdue University). Luke McDowell was supported in part by NSF award number 1116439 and by a grant from ONR. This research was also partly supported by the NSF under the contract number IIS-1149789. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ONR, NSF, or the U.S. Government.

References

Adamic, L. A., & Adar, E. (2001). Friends and neighbors on the web. *Social Networks*, *25*(3), 211–230.

Adibi, J., Chalupsky, H., Melz, E., Valente, A., et al. (2004). The KOJAK group finder: Connecting the dots via integrated knowledge-based and statistical reasoning. In *Proceedings of the 16th Conference on Innovative Applications of Artifical Intelligence*, pp. 800–807.

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*(5), 734–749.

Ahmed, N., Neville, J., & Kompella, R. (2012a). Network sampling designs for relational classification. In *Proceedings of the 6th International AAAI Conference on Weblogs and Social Media*.

Ahmed, N., Neville, J., & Kompella, R. (2012b). Space-efficient sampling from social activity streams. In *BigMine*, pp. 1–8.

Ahmed, N., Berchmans, F., Neville, J., & Kompella, R. (2010). Time-based sampling of social network activity graphs. In *Proceedings of the 8th Workshop on Mining and Learning with Graphs*, pp. 1–9.

Airoldi, E. M., Blei, D. M., Fienberg, S. E., & Xing, E. P. (2008). Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, *9*, 1981–2014.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19*(6), 716–723.

Albert, R., Jeong, H., & Barabási, A. (1999). Internet: Diameter of the world-wide web. *Nature*, *401*(6749), 130–131.

Amarel, S. (1968). On representations of problems of reasoning about actions. *Machine Intelligence*, *3*, 131–171.

Anthony, A., & desJardins, M. (2007). Data clustering with a relational push-pull model. In *Proceedings of the Seventh IEEE International Conference on Data Mining Workshops*, ICDMW '07, pp. 189–194.

Aral, S., & Walker, D. (2010). Creating Social Contagion through Viral Product Design: A Randomized Trial of Peer Influence in Networks. In *Proceedings of the 31st International Conference on Information Systems*.

Backstrom, L., & Leskovec, J. (2011). Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the 4th International Conference on Web Search and Data Mining*, pp. 635–644.

Backstrom, L., Dwork, C., & Kleinberg, J. M. (2007). Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th International World Wide Web Conference*, pp. 181–190.

Balasubramanyan, R., & Cohen, W. (2011). Block-LDA: Jointly modeling entity-annotated text and entity-entity links. In *Proceedings of the 7th SIAM International Conference on Data Mining*.

Barabási, A., & Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509–512.

Barabási, A., & Crandall, R. (2003). Linked: The new science of networks. *American journal of Physics*, *71*(4), 409–410.

Basilico, J. B., & Hofmann, T. (2004). Unifying collaborative and content-based filtering. In *Proceedings of the 21st International Conference on Machine Learning*, pp. 65–72.

Ben-Hur, A., & Noble, W. (2005). Kernel methods for predicting protein-protein interactions. *Bioinformatics*, *21*(Suppl. 1), 38–46.

Benczúr, A., Csalogány, K., Sarlós, T., & Uher, M. (2005). Spamrank–fully automatic link spam detection. In *Adversarial Information Retrieval on the Web*, pp. 25–38.

Berkhin, P. (2006). Survey of clustering data mining techniques. *Grouping Multidimensional Data: Recent Advances in Clustering*, *10*, 25–71.

Bharat, K., & Henzinger, M. (1998). Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of the 21st International SIGIR Conference on Research and Development in Information Retrieval*, pp. 104–111.

Bhattacharya, I., & Getoor, L. (2005). Relational clustering for multi-type entity resolution. In *Proceedings of the 4th International workshop on Multi-relational Mining*, pp. 3–12.

Bhattacharya, I., & Getoor, L. (2007). Collective entity resolution in relational data. *Transactions on Knowledge Discovery from Data*, *1*(1), 1–36.

Biba, M., Ferilli, S., & Esposito, F. (2008). Discriminative structure learning of Markov logic networks. *Inductive Logic Programming*, *5194*, 59–76.

Bilgic, M., Mihalkova, L., & Getoor, L. (2010). Active learning for networked data. In *Proceedings of the 27th International Conference on Machine Learning*.

Bilgic, M., Namata, G. M., & Getoor, L. (2007). Combining collective classification and link prediction. In *Proceedings of the 7th IEEE International Conference on Data Mining Workshops*, pp. 381–386.

Blei, D., Ng, A., & Jordan, M. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Blei, D., & Lafferty, J. (2007). A correlated topic model of science. *The Annals of Applied Statistics*, *1*(1), 17–35.

Bonacich, P., & Lloyd, P. (2001). Eigenvector-like measures of centrality for asymmetric relations. *Social Networks*, *23*(3), 191–201.

Borgman, C., & Furner, J. (2002). Scholarly communication and bibliometrics. *Annual Review of Information Science and Technology*, *36*, 3–72.

Brightwell, G., & Winkler, P. (1990). Maximum hitting time for random walks on graphs. *Random Structures & Algorithms*, *1*(3), 263–276.

Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A., & Wiener, J. (2000). Graph structure in the web. *Computer networks*, *33*(1-6), 309–320.

Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, *2*(2), 121–167.

Cafarella, M. J., Wu, E., Halevy, A., Zhang, Y., & Wang, D. Z. (2008). Webtables: Exploring the power of tables on the web. In *Proceedings of VLDB*, pp. 538–549.

Camacho, J., Guimerà, R., & Nunes Amaral, L. (2002). Robust patterns in food web structure. *Physical Review Letters*, *88*(22), 228102: 1–4.

Chakrabarti, S., Dom, B., & Indyk, P. (1998). Enhanced hypertext categorization using hyperlinks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 307–318.

Chakrabarti, S., Dom, B., Raghavan, P., Rajagopalan, S., Gibson, D., & Kleinberg, J. (1998). Automatic resource compilation by analyzing hyperlink structure and associated text. *Computer Networks and ISDN Systems*, *30*(1-7), 65–74.

Chang, J., & Blei, D. (2009). Relational topic models for document networks. In *The 9th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 81–88.

Chang, J., Boyd-Graber, J., & Blei, D. (2009). Connections between the lines: augmenting social networks with text. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 169–178.

Clauset, A., Moore, C., & Newman, M. (2008). Hierarchical structure and the prediction of missing links in networks. *Nature*, *453*(7191), 98–101.

Cohn, D., & Chang, H. (2000). Learning to probabilistically identify authoritative documents. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 167–174.

Cohn, D., & Hofmann, T. (2001). The missing link-a probabilistic model of document content and hypertext connectivity. *Advances in Neural Information Processing Systems*, *13*, 430–436.

Crane, R., & McDowell, L. K. (2011). Evaluating markov logic networks for collective classification. In *Proceedings of the 9th MLG Workshop at the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., & Slattery, S. (2000). Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence*, *118*(1-2), 69–113.

Cristianini, N., & Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press.

Dash, M., & Liu, H. (1997). Feature selection for classification. *Intelligent data analysis*, *1*(3), 131–156.

Davis, J., Burnside, E., Castro Dutra, I., Page, D., & Costa, V. (2005). An integrated approach to learning Bayesian networks of rules. In *Proceedings of the European Conference on Machine Learning*, pp. 84–95.

Davis, J., Ong, I., Struyf, J., Burnside, E., Page, D., & Costa, V. S. (2007). Change of representation for statistical relational learning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pp. 2719–2725.

De Raedt, L. (2008). *Logical and relational learning*. Springer.

De Raedt, L., & Kersting, K. (2008). *Probabilistic inductive logic programming*. Springer-Verlag.

De Raedt, L., & Thon, I. (2010). Probabilistic rule learning. *Inductive Logic Programming*, *6489*, 47–58.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*, 391–407.

Deshpande, M., Kuramochi, M., Wale, N., & Karypis, G. (2005). Frequent substructure-based approaches for classifying chemical compounds. *IEEE Transactions on Knowledge and Data Engineering*, *13*, 1036–1050.

Dhillon, I. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 269–274.

Dietz, L., Bickel, S., & Scheffer, T. (2007). Unsupervised prediction of citation influences. In *Proceedings of the 24th International Conference on Machine Learning*, pp. 233–240.

Domingos, P., & Richardson, M. (2004). Markov logic: A unifying framework for statistical relational learning. In *Proceedings of the ICML Workshop on Statistical Relational Learning*, pp. 49–54.

DuBois, C., & Smyth, P. (2010). Modeling Relational Events via Latent Classes. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 803–812.

Dunne, J., Williams, R., & Martinez, N. (2002). Food-web structure and network theory: the role of connectance and size. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(20), 12917.

Easley, D., & Kleinberg, J. (2010). *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press.

Eckart, C., & Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, *1*(3), 211–218.

Egghe, L., & Rousseau, R. (1990). *Introduction to informetrics*. Elsevier Science Publishers.

Erosheva, E., Fienberg, S., & Lafferty, J. (2004). Mixed-membership models of scientific publications. *Proceedings of the National Academy of Sciences of the United States of America*, *101*(Suppl 1), 5220.

Essen, U., & Steinbiss, V. (1992). Cooccurrence smoothing for stochastic language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pp. 161–164.

Faloutsos, M., Faloutsos, P., & Faloutsos, C. (1999). On power-law relationships of the internet topology. In *Proceedings of the ACM SIGCOMM International Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pp. 251–262.

Fast, A., & Jensen, D. (2008). Why stacked models perform effective collective classification. In *Proceedings of the IEEE International Conference on Data Mining*, pp. 785–790.

Fodor, I. (2002). A Survey of Dimension Reduction Techniques. *US DOE Office of Scientific and Technical Information*, *18*.

Frank, R., Moser, F., & Ester, M. (2007). A method for multi-relational classification using single and multi-feature aggregation functions. *Proceedings of the Principles and Practice of Knowledge Discovery in Databases*, *1*, 430–437.

Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry*, *40*, 35–41.

Friedman, J. (2001). Greedy function approximation: A gradient boosting machine.. *The Annals of Statistics*, *29*(5), 1189–1232.

Friedman, N., Getoor, L., Koller, D., & Pfeffer, A. (1999). Learning probabilistic relational models. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pp. 1300–1309. Springer-Verlag.

Gallagher, B., Tong, H., Eliassi-Rad, T., & Faloutsos, C. (2008). Using ghost edges for classification in sparsely labeled networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 256–264.

Gärtner, T. (2003). A survey of kernels for structured data. *ACM SIGKDD Explorations Newsletter*, *5*(1), 49–58.

George, E., & McCulloch, R. (1993). Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, *88*, 881–889.

Getoor, L., Friedman, N., Koller, D., & Taskar, B. (2003). Learning probabilistic models of link structure. *Journal of Machine Learning Research*, *3*, 679–707.

Getoor, L., & Taskar, B. (Eds.). (2007). *Introduction to Statistical Relational Learning*. MIT Press.

Getoor, L., & Diehl, C. P. (2005). Link mining. *SIGKDD Explorations*, *7*, 3–12.

Getoor, L., Friedman, N., Koller, D., & Taskar, B. (2001). Learning probabilistic models of relational structure. In *Proceedings of International Conference on Machine Learning*, pp. 170–177.

Gibson, D., Kleinberg, J., & Raghavan, P. (1998). Inferring web communities from link topology. In *Proceedings of the 9th ACM Conference on Hypertext and Hypermedia*, pp. 225–234.

Gilbert, E., & Karahalios, K. (2009). Predicting tie strength with social media. In *Proceedings of the 27th CHI International Conference on Human Factors in Computing Systems*, pp. 211–220.

Girvan, M., & Newman, E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, *99*(12), 7821–7826.

Goadrich, M., & Shavlik, J. (2007). Combining clauses with various precisions and recalls to produce accurate probabilistic estimates. In *Proceedings of the 17th International Conference on Inductive Logic Programming*, pp. 122–131.

Göbel, F., & Jagers, A. (1974). Random walks on graphs. *Stochastic processes and their applications*, *2*(4), 311–336.

Godbole, N., Srinivasaiah, M., & Skiena, S. (2007). Large-scale sentiment analysis for news and blogs. In *Proceedings of the International Conference on Weblogs and Social Media*.

Golub, G., & Reinsch, C. (1970). Singular value decomposition and least squares solutions. *Numerische Mathematik*, *14*(5), 403–420.

Green, J. (1972). Latitudinal variation in associations of planktonic Rotifera. *Journal of zoology*, *167*(1), 31–39.

Gruber, A., Rosen-Zvi, M., & Weiss, Y. (2008). Latent topic models for hypertext. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, pp. 230–239.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, *3*, 1157–1182.

Gyongyi, Z., Garcia-Molina, H., & Pedersen, J. (2004). Combating web spam with trustrank. In *Proceedings of VLDB*, pp. 576–587.

Hannan, E., & Quinn, B. (1979). The determination of the order of an autoregression. *Journal of the Royal Statistical Society. Series B (Methodological)*, *41*, 190–195.

Harshman, R. (1970). Foundations of the PARAFAC procedure: Models and conditions for an explanatory multi-modal factor analysis. *UCLA Working Papers in Phonetics*, *16*(1), 84.

Hartigan, J., & Wong, M. (1979). A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C, Applied statistics*, *28*, 100–108.

Hasan, M. A., Chaoji, V., Salem, S., & Zaki, M. (2006). Link prediction using supervised learning. In *Proceedings of the SDM Workshop on Link Analysis, Counterterrorism and Security*.

Haveliwala, T. (2003). Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE transactions on knowledge and data engineering*, *15*, 784–796.

Hay, M., Miklau, G., Jensen, D., Towsley, D., & Weis, P. (2008). Resisting structural re-identification in anonymized social networks. In *Proceedings of VLDB*, pp. 102–114.

He, D., & Parker, D. (2010). Topic Dynamics: an alternative model of 'Bursts' in Streams of Topics. In *Proceeding of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 443–452.

Henderson, K., Gallagher, B., Li, L., Akoglu, L., Eliassi-Rad, T., Tong, H., & Faloutsos, C. (2011). It's Who You Know: Graph Mining Using Recursive Structural Features. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1–10.

Hill, S., Agarwal, D., Bell, R., & Volinsky, C. (2006). Building an effective representation for dynamic networks. *Journal of Computational and Graphical Statistics*, *15*(3), 584–608.

Hoff, P., Raftery, A., & Handcock, M. (2002). Latent space approaches to social network analysis. *Journal of the American Statistical Association*, *97*(460), 1090–1098.

Hofmann, T. (1999). Probabilistic latent semantic analysis. In *Proceedings of Uncertainty in Artificial Intelligence*, pp. 289–296.

Huh, S., & Fienberg, S. (2010). Discriminative Topic Modeling based on Manifold Learning. In *Proceeding of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 653–661.

Huynh, T., & Mooney, R. (2008). Discriminative structure and parameter learning for markov logic networks. In *Proceedings of the 25th International Conference on Machine Learning*.

Inokuchi, A., Washio, T., & Motoda, H. (2000). An apriori-based algorithm for mining frequent substructures from graph data. In *Principles of Data Mining and Knowledge Discovery*, pp. 13–23.

Jaccard, P. (1901). *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz.

Jackson, M. (2008). *Social and economic networks*. Princeton Univ Press.

Jain, A., & Zongker, D. (1997). Feature selection: Evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(2), 153–158.

Jeh, G., & Widom, J. (2002). SimRank: A measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 538–543.

Jensen, D., & Neville, J. (2002). Linkage and autocorrelation cause feature selection bias in relational learning. In *Proceedings of the 19th International Conference on Machine Learning*, pp. 259–266.

Jensen, D., Neville, J., & Gallagher, B. (2004). Why collective inference improves relational classification. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 593–598.

Jensen, D., Neville, J., & Hay, M. (2003). Avoiding bias when aggregating relational data with degree disparity. In *Proceedings of the 20th International Conference on Machine Learning*, pp. 274–281.

Jensen, D., Fast, A., Taylor, B., & Maier, M. (2008). Automatic identification of quasi-experimental designs for discovering causal knowledge. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 372–380.

Jeong, H., Mason, S., Barabási, A., & Oltvai, Z. (2001). Lethality and centrality in protein networks. *Nature*, *411*(6833), 41–42.

Jeong, H., Tombor, B., Albert, R., Oltvai, Z., & Barabási, A. (2000). The large-scale organization of metabolic networks. *Nature*, *407*(6804), 651–654.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*, pp. 137–142.

Johnson, S. (1967). Hierarchical clustering schemes. *Psychometrika*, *32*(3), 241–254.

Kahanda, I., & Neville, J. (2009). Using transactional information to predict link strength in online social networks. In *Proceedings of the 4th International Conference on Weblogs and Social Media*, pp. 106–113.

Kamvar, S., Klein, D., & Manning, C. (2003). Spectral learning. In *Proceedings of the 18th International Joint Conference On Artificial Intelligence*, pp. 561–566.

Kashima, H., & Abe, N. (2006). A parameterized probabilistic model of network evolution for supervised link prediction. In *Proceedings of the IEEE International Conference on Data Mining*, pp. 340–349.

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

Kavurucu, Y., Senkul, P., & Toroslu, I. (2008). Aggregation in confidence-based concept discovery for multi-relational data mining. In *Proceedings of IADIS European Conference on Data Mining*, pp. 43–52.

Kersting, K., & De Raedt, L. (2002). Basic principles of learning Bayesian logic programs. Tech. rep. 174, Institute for Computer Science, University of Freiburg.

Kersting, K., De Raedt, L., Gutmann, B., Karwath, A., & Landwehr, N. (2008). Relational sequence learning. *Probabilistic inductive logic programming*, *4911*, 28–55.

Khosravi, H., Tong Man, O., Xu, X., & Bina, B. (2010). Structure learning for markov logic networks with many descriptive attributes. In *Proceedings of the 24th Conference on Artificial Intelligence*, pp. 487–493.

Khot, T., Natarajan, S., Kersting, K., & Shavlik, J. (2011). Learning markov logic networks via functional gradient boosting. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pp. 320–329. IEEE.

Kim, M., & Leskovec, J. (2011). The network completion problem: Inferring missing nodes and edges in networks. In *Proceedings of the SIAM International Conference on Data Mining*.

Kleczkowski, A., & Grenfell, B. (1999). Mean-field-type equations for spread of epidemics: the small world'model. *Physica A: Statistical Mechanics and its Applications*, *274*(1-2), 355–360.

Kleinberg, J. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM*, *46*(5), 604–632.

Knobbe, A., Siebes, A., & Marseille, B. (2002). Involving aggregate functions in multi-relational search. In *Principles of Data Mining and Knowledge Discovery*, pp. 145–168.

Kohavi, R., & John, G. (1997). Wrappers for feature subset selection. *Artificial intelligence*, *97*(1-2), 273–324.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, *78*(9), 1464–1480.

Kok, S., & Domingos, P. (2005). Learning the structure of Markov logic networks. In *Proceedings of the 22nd International Conference on Machine Learning*, pp. 441–448.

Kok, S., & Domingos, P. (2007). Statistical predicate invention. In *Proceedings of the 24th International Conference on Machine Learning*, pp. 433–440.

Kok, S., & Domingos, P. (2008). Extracting semantic networks from text via relational clustering. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pp. 624–639.

Kok, S., & Domingos, P. (2009). Learning markov logic network structure via hypergraph lifting. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 505–512.

Kok, S., & Domingos, P. (2010). Learning markov logic networks using structural motifs. In *Proceedings of the 27th International Conference on Machine Learning*.

Kolda, T. G., Bader, B. W., & Kenny, J. P. (2005). Higher-order web link analysis using multilinear algebra. In *Proceedings of the IEEE International Conference on Data Mining*, pp. 242–249.

Kolda, T., & Bader, B. (2006). The TopHITS model for higher-order web link analysis. In *Proceedings of the SIAM Data Mining Conference Workshop on Link Analysis, Counterterrorism and Security*, pp. 26–29.

Koller, D., & Sahami, M. (1996). Toward optimal feature selection. In *Proceedings of the 13th International Conference on Machine Learning*, pp. 284–292.

Kondor, R., & Lafferty, J. (2002). Diffusion kernels on graphs and other discrete structures. In *Proceedings of the 19th International Conference on Machine Learning*, pp. 315–322.

Kong, X., & Yu, P. (2010). Semi-supervised feature selection for graph classification. In *Proceeding of the 16th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pp. 793–802.

Koren, Y., North, S., & Volinsky, C. (2007). Measuring and extracting proximity graphs in networks. *Transactions on Knowledge Discovery from Data (TKDD)*, *1*(3), 12:1–12:30.

Kosala, R., & Blockeel, H. (2000). Web Mining Research: A Survey. *ACM SIGKDD Explorations Newsletter*, *2*(1), 1–15.

Kossinets, G., Kleinberg, J., & Watts, D. (2008). The structure of information pathways in a social communication network. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 435–443.

Kou, Z., & Cohen, W. (2007). Stacked graphical models for efficient inference in markov random fields. In *Proceedings of the 7th SIAM International Conference on Data Mining*, pp. 533–538.

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

Krogel, M., & Wrobel, S. (2001). Transformation-based learning using multirelational aggregation. *Inductive logic programming*, *2157*, 142–155.

Kubica, J., Moore, A., Schneider, J., & Yang, Y. (2002). Stochastic link and group detection. In *Proceedings of the 18th AAAI Conference on Artificial Intelligence*, pp. 798–806.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289.

Landwehr, N., Kersting, K., & De Raedt, L. (2005). nFOIL: Integrating naıve bayes and FOIL. In *Proceedings of the 20th AAAI Conference on Artificial Intelligence*, pp. 275–282.

Landwehr, N., Passerini, A., De Raedt, L., & Frasconi, P. (2010). Fast learning of relational kernels. *Machine learning*, *78*(3), 305–342.

Langville, A., & Meyer, C. (2005). A Survey of Eigenvector Methods for Web Information Retrieval. *SIAM Review*, *47*(1), 135–161.

Lassez, J.-L., Rossi, R., & Jeev, K. (2008). Ranking links on the web: Search and surf engines. In *IEA/AIE*, pp. 199–208.

Lee, L. (1999). Measures of distributional similarity. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pp. 25–32.

Leicht, E., Holme, P., & Newman, M. (2006). Vertex similarity in networks. *Physical Review E*, *73*(2), 026120.

Leiva, H. A., Gadia, S., & Dobbs, D. (2002). Mrdtl: A multi-relational decision tree learning algorithm. In *Proceedings of the 13th International Conference on Inductive Logic Programming*, pp. 38–56.

Lempel, R., & Moran, S. (2000). The stochastic approach for link-structure analysis (SALSA) and the TKC effect. *Computer Networks*, *33*(1-6), 387–401.

Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C., & Ghahramani, Z. (2010). Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research*, *11*, 985–1042.

Leskovec, J., Huttenlocher, D., & Kleinberg, J. (2010). Predicting positive and negative links in online social networks. In *Proceedings of the 19th International World Wide Web Conference*, pp. 641–650.

Letovsky, S., & Kasif, S. (2003). Predicting protein function from protein/protein interaction data: a probabilistic approach. *Bioinformatics*, *19*(Suppl 1), i197.

Li, W., & McCallum, A. (2006). Pachinko allocation: DAG-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, pp. 577–584.

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

Lichtenwalter, R., Lussier, J., & Chawla, N. (2010). New Perspectives and Methods in Link Prediction. In *Proceeding of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 243–252.

Lim, T., Loh, W., & Shih, Y. (2000). A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning*, *40*(3), 203–228.

Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, pp. 296–304.

Lin, F., & Cohen, W. W. (2010). Semi-supervised classification of network data using very few labels. In *Proceedings of the International Conference on Advances in Social Network Analysis and Mining*.

Lippi, M., Jaeger, M., Frasconi, P., & Passerini, A. (2009). Relational information gain. *Machine Learning*, *83*(2), 1–21.

Liu, K., & Terzi, E. (2008). Towards identity anonymization on graphs. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 93–106.

Liu, W., & Lü, L. (2010). Link prediction based on local random walk. *Europhysics Letters*, *89*, 58007.

Liu, Y., Niculescu-Mizil, A., & Gryc, W. (2009). Topic-link LDA: joint models of topic and author community. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 665–672.

Long, B., Zhang, Z., & Yu, P. (2007). A probabilistic framework for relational clustering. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 470–479.

Long, B., Zhang, Z., Wu, X., & Yu, P. S. (2006). Spectral clustering for multi-type relational data. In *Proceedings of the 23rd International Conference on Machine Learning*, pp. 585–592.

Lu, C., Hu, X., Chen, X., & ran Park, J. (2010). The Topic-Perspective Model for Social Tagging Systems. In *Proceeding of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 683–692.

Lu, Q., & Getoor, L. (2003). Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning*, pp. 496–503.

Macskassy, S., & Provost, F. (2003). A simple relational classifier. In *Proceedings of the SIGKDD 2nd Workshop on Multi-Relational Data Mining*, pp. 64–76.

Macskassy, S., & Provost, F. (2007). Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, *8*, 935–983.

Macskassy, S. A. (2007). Improving learning in networked data by combining explicit and mined links. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pp. 590–595.

Maes, F., Peters, S., Denoyer, L., & Gallinari, P. (2009). Simulated iterative classification a new learning procedure for graph labeling. *Machine Learning and Knowledge Discovery in Databases*, *5782*, 47–62.

Mallows, C. (1973). Some comments on C_p. *Technometrics*, *42*(1), 87–94.

Maslov, S., & Sneppen, K. (2002). Specificity and stability in topology of protein networks. *Science*, *296*(5569), 910–913.

May, R., & Lloyd, A. (2001). Infection dynamics on scale-free networks. *Physical Review E*, *64*(6), 66112.

McCallum, A., Wang, X., & Corrada-Emmanuel, A. (2007). Topic and role discovery in social networks with experiments on enron and academic email. In *Journal of Artificial Intelligence Research*, pp. 249–272.

McCallum, A., Wang, X., & Mohanty, N. (2007). Joint group and topic discovery from relations and text. In *Statistical Network Analysis: Models, Issues and New Directions, Lecture Notes in Computer Science 4503*, pp. 28–44.

McDowell, L., Gupta, K., & Aha, D. (2009). Cautious collective classification. *Journal of Machine Learning Research*, *10*, 2777–2836.

McDowell, L., Gupta, K., & Aha, D. (2007). Cautious inference in collective classification. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*.

McDowell, L., Gupta, K., & Aha, D. (2010). Meta-Prediction for Collective Classification. In *Proceedings of the 23rd International FLAIRS Conference*.

McGovern, A., Friedland, L., Hay, M., Gallagher, B., Fast, A., Neville, J., & Jensen, D. (2003). Exploiting relational structure to understand publication patterns in high-energy physics. *SIGKDD Explorations*, *5*(2), 165–172.

McGovern, A., Collier, N., Matthew Gagne, I., Brown, D., & Rodger, A. (2008). Spatiotemporal Relational Probability Trees: An Introduction. In *Eighth IEEE International Conference on Data Mining, ICDM.*, pp. 935–940.

Menon, A., & Elkan, C. (2011). Link prediction via matrix factorization. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pp. 437–452.

Menon, A., & Elkan, C. (2010). Predicting labels for dyadic data. *Data Mining and Knowledge Discovery*, *21*(2), 327–343.

Michie, D., Spiegelhalter, D., Taylor, C., & Campbell, J. (1994). *Machine Learning, Neural and Statistical Classification*. Ellis Horwood Limited.

Mihalkova, L., & Mooney, R. (2007). Bottom-up learning of Markov logic network structure. In *Proceedings of the 24th International Conference on Machine Learning*, pp. 625–632.

Miller, K., Griffiths, T., & Jordan, M. (2009). Nonparametric latent feature models for link prediction. *Advances in Neural Information Processing Systems (NIPS)*, *10*, 1276–1284.

Minsky, M. (1974). A framework for representing knowledge. Tech. rep., Massachusetts Institute of Technology, Cambridge, MA, USA.

Mislove, A., Marcon, M., Gummadi, K., Druschel, P., & Bhattacharjee, B. (2007). Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet measurement*, pp. 29–42.

Moore, C., & Newman, M. (2000). Epidemics and percolation in small-world networks. *Physical Review E*, *61*(5), 5678–5682.

Moreno, S., & Neville, J. (2009). An investigation of the distributional characteristics of generative graph models. In *Proceedings of the 1st Workshop on Information in Networks*.

Nallapati, R., Ahmed, A., Xing, E., & Cohen, W. (2008). Joint latent topic models for text and citations. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 542–550.

Namata, G., Kok, S., & Getoor, L. (2011). Collective graph identification. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 87–95. ACM.

Narayanan, A., & Shmatikov, V. (2009). De-anonymizing social networks. In *Proceedings of the 30th IEEE Symposium on Security and Privacy*, pp. 173–187.

Natarajan, S., Khot, T., Kersting, K., Gutmann, B., & Shavlik, J. (2012). Gradient-based boosting for statistical relational learning: The relational dependency network case. *Machine Learning*, *86*, 25–56.

Neville, J., Adler, M., & Jensen, D. (2004). Spectral clustering with links and attributes. Tech. rep. 04-42, Dept of Computer Science, University of Massachusetts Amherst.

Neville, J., & Jensen, D. (2000). Iterative classification in relational data. In *Proceedings of the Workshop on SRL, 17th AAAI Conference on Artificial Intelligence*, pp. 42–49.

Neville, J., & Jensen, D. (2005). Leveraging relational autocorrelation with latent group models. In *Proceedings of the 5th IEEE International Conference on Data Mining*, pp. 322–329.

Neville, J., & Jensen, D. (2007). Relational dependency networks. *Journal of Machine Learning Research*, *8*, 653–692.

Neville, J., Jensen, D., Friedland, L., & Hay, M. (2003). Learning relational probability trees. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 625–630.

Neville, J., Jensen, D., & Gallagher, B. (2003). Simple estimators for relational Bayesian classifers. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pp. 609–612.

Neville, J., Simsek, O., Jensen, D., Komoroske, J., Palmer, K., & Goldberg, H. (2005). Using relational knowledge discovery to prevent securities fraud. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pp. 449–458.

Newman, M. (2010). *Networks: An Introduction*. Oxford Univ Press.

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

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

Newman, M. (2001b). The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, *98*(2), 404–409.

Newman, M., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, *69*(2), 26113.

Ng, A., Jordan, M., & Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pp. 849–856.

Nie, L., Davison, B., & Qi, X. (2006). Topical link analysis for web search. In *Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval*, p. 98.

Nowicki, K., & Snijders, T. (2001). Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, *96*, 1077–1087.

Oktay, H., Taylor, B., & Jensen, D. (2010). Causal Discovery in Social Media Using Quasi-Experimental Designs. In *Proceedings of the ACM SIGKDD 1st Workshop on Social Media Analytics (SOMA-KDD)*.

Onnela, J.-P., Saramaki, J., Hyvonen, J., Szabo, G., Lazer, D., Kaski, K., Kertesz, J., & Barabasi, A.-L. (2007). Structure and tie strengths in mobile communication networks. *Proceedings of the National Academy of Sciences*, *104*, 7332–7336.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Tech. rep., Technical Report, Stanford Digital Library Technologies Project.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, *2*(1-2), 1–135.

Pastor-Satorras, R., & Vespignani, A. (2001). Epidemic spreading in scale-free networks. *Physical Review Letters*, *86*(14), 3200–3203.

Pasula, H., Marthi, B., Milch, B., Russell, S., & Shpitser, I. (2003). Identity uncertainty and citation matching. In *In NIPS*. MIT Press.

Perlich, C., & Provost, F. (2003). Aggregation-based feature invention and relational concept classes. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 167–176.

Perlich, C., & Provost, F. (2006). Acora: Distribution-based aggregation for relational learning from identifier attributes. *Machine Learning*, *62*, 65–105.

Poole, D. (2003). First-order probabilistic inference. In *International Joint Conference on Artificial Intelligence*, pp. 985–991.

Popescul, A., Popescul, R., & Ungar, L. H. (2003a). Statistical relational learning for link prediction. In *Proceedings of the Workshop on Learning Statistical Models from Relational Data at IJCAI*.

Popescul, A., Popescul, R., & Ungar, L. H. (2003b). Structural logistic regression for link analysis. In *Proceedings of the Second International Workshop on Multi-Relational Data Mining*, pp. 92–106.

Popescul, A., & Ungar, L. H. (2004). Cluster-based concept invention for statistical relational learning. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 665–670.

Price, D. (1976). A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science*, *27*(5), 292–306.

Pudil, P., Novovicová, J., & Kittler, J. (1994). Floating search methods in feature selection. *Pattern recognition letters*, *15*(11), 1119–1125.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., & Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences*, *101*(9), 2658–2663.

Rattigan, M. J., & Jensen, D. (2005). The case for anomalous link discovery. *SIGKDD Explorations Newsletter*, *7*(2), 41–47.

Ravasz, E., Somera, A., Mongru, D., Oltvai, Z., & Barabási, A. (2002). Hierarchical organization of modularity in metabolic networks. *Science*, *297*(5586), 1551–1555.

Resnick, P., & Varian, H. (1997). Recommender systems. *Communications of the ACM*, *40*(3), 56–58.

Richardson, M., & Domingos, P. (2002). The intelligent surfer: Probabilistic combination of link and content information in pagerank. In *Advances in Neural Information Processing Systems*, pp. 1441–1448.

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, *62*(1), 107–136.

Riedel, S., & Meza-Ruiz, I. (2008). Collective semantic role labelling with markov logic. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pp. 193–197.

Robins, G., Pattison, P., Kalish, Y., & Lusher, D. (2007). An introduction to exponential random graph (p*) models for social networks. *Social Networks*, *29*, 173–191.

Robins, G., Snijders, T., Wang, P., & Handcock, M. (2006). Recent developments in exponential random graph (p*) models for social networks. *Social Networks*, *29*, 192–215.

Rossi, R., Gallagher, B., Neville, J., & Henderson, K. (2012). Dynamic behavioral mixed-membership model for large evolving networks. In *Arxiv preprint arXiv:1205.2056*, pp. 1–17.

Rossi, R., & Neville, J. (2010). Modeling the evolution of discussion topics and communication to improve relational classification. In *Proceedings of the ACM SIGKDD 1st Workshop on Social Media Analytics (SOMA-KDD)*, pp. 1–10.

Rossi, R., Gallagher, B., Neville, J., & Henderson, K. (2012). Role-Dynamics: Fast Mining of Large Dynamic Networks. In *LSNA-WWW*, pp. 1–9.

Rossi, R., & Neville, J. (2012). Time-evolving relational classification and ensemble methods. In *Proceedings of the 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 1–12.

Roth, M., Ben-David, A., Deutscher, D., Flysher, G., Horn, I., Leichtberg, A., Leiser, N., Merom, R., & Mattias, Y. (2010). Suggesting Friends Using the Implicit Social Graph. In *Proceeding of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 233–242.

Russell, S. J., & Norvig, P. (2009). *Artificial Intelligence: A Modern Approach* (3rd International Edition edition). Prentice Hall.

Sabidussi, G. (1966). The centrality index of a graph. *Psychometrika*, *31*(4), 581–603.

Salton, G., & McGill, M. (1983). *Introduction to modern information retrieval*, Vol. 1. McGraw-Hill New York.

Sarkar, P., & Moore, A. (2005). Dynamic Social Network Analysis using Latent Space Models. *SIGKDD Explorations Newsletter*, *7*(2), 31–40.

Sarukkai, R. R. (2000). Link prediction and path analysis using markov chains. In *Proceedings of the 9th International World Wide Web Conference on Computer Networks: The International Journal of Computer and Telecommunications Networking*, pp. 377–386.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). Application of dimensionality reduction in recommender system–a case study. In *ACM WebKDD 2000 Web Mining for E-Commerce Workshop*.

Schulte, O. (2011). A tractable pseudo-likelihood function for bayes nets applied to relational data. In *SDM*, pp. 462–473.

Schwarz, G. (1978). Estimating the dimension of a model. *The annals of statistics*, *6*(2), 461–464.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., & Eliassi-Rad, T. (2008). Collective classification in network data. *AI Magazine*, *29*(3), 93.

Shao, J. (1996). Bootstrap model selection. *Journal of the American Statistical Association*, *91*(434), 655–665.

Shapiro, E. (1982). Algorithmic Program Debugging. ACM Distinguished Dissertation..

Sharan, U., & Neville, J. (2008). Temporal-relational classifiers for prediction in evolving domains. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pp. 540–549.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 888–905.

Shi, X., Li, Y., & Yu, P. (2011). Collective prediction with latent graphs. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management*, pp. 1127–1136.

Singla, P., & Domingos, P. (2006). Entity resolution with markov logic. In *Proceedings of the 6th IEEE International Conference on Data Mining*, pp. 572–582.

Srinivasan, A. (1999). The aleph manual. *Computing Laboratory, Oxford University*, *1*.

Strehl, A., & Ghosh, J. (2003). Cluster ensembles–a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, *3*, 583–617.

Tang, J., Musolesi, M., Mascolo, C., & Latora, V. (2009). Temporal distance metrics for social network analysis. In *Proceedings of the 2nd ACM workshop on Online social networks*, pp. 31–36.

Tang, J., Musolesi, M., Mascolo, C., Latora, V., & Nicosia, V. (2010). Analysing information flows and key mediators through temporal centrality metrics. In *Proceedings of the 3rd Workshop on Social Network Systems*, pp. 1–6.

Tang, L., & Liu, H. (2009). Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 817–826.

Tang, L., & Liu, H. (2011). Leveraging social media networks for classification. *Journal of Data Mining and Knowledge Discovery*, *23*, 447–478.

Taskar, B., Abbeel, P., & Koller, D. (2002). Discriminative probabilistic models for relational data. In *Eighteenth Conference on Uncertainty in Artificial Intelligence*, pp. 485–492.

Taskar, B., Segal, E., & Koller, D. (2001). Probabilistic classification and clustering in relational data. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pp. 870–878.

Taskar, B., Wong, M., Abbeel, P., & Koller, D. (2003). Link prediction in relational data. In *Advances in Neural Information Processing Systems*.

Topchy, A., Law, M., Jain, A., & Fred, A. (2004). Analysis of consensus partition in cluster ensemble. In *Proceedings of the 4th IEEE International Conference on Data Mining*, pp. 225–232.

Vert, J., & Yamanishi, Y. (2005). Supervised graph inference. *Advances in Neural Information Processing Systems*, *17*, 1433–1440.

Vishwanathan, S., Schraudolph, N., Kondor, R., & Borgwardt, K. (2010). Graph kernels. *Journal of Machine Learning Research*, *11*, 1201–1242.

von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, *17*(4), 395–416.

Wagner, A., & Fell, D. (2001). The small world inside large metabolic networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, *268*(1478), 1803–1810.

Wang, C., Blei, D., & Heckerman, D. (2008). Continuous time dynamic topic models. In *Proceedings of Uncertainty in Artificial Intelligence*.

Wang, X., & McCallum, A. (2006). Topics over time: a non-Markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 424–433.

Wang, Z., & Chan, L. (2010). An efficient causal discovery algorithm for linear models. In *Proceeding of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1109–1118.

Wasserman, S., & Faust, K. (1994). *Social network analysis: Methods and applications*. Cambridge University Press.

Watts, D., & Strogatz, S. (1998). Collective dynamics of small-worldnetworks. *Nature*, *393*(6684), 440–442.

White, S., & Smyth, P. (2003). Algorithms for estimating relative importance in networks. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, pp. 266–275.

Wu, B., & Davison, B. (2005). Identifying link farm spam pages. In *Special interest tracks and posters of the 14th International Conference on World Wide Web*, pp. 820–829.

Xiang, R., Neville, J., & Rogati, M. (2010). Modeling relationship strength in online social networks. In *Proceedings of the 19th International World Wide Web Conference*, pp. 981–990.

Yang, Y., & Pedersen, J. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pp. 412–420.

Yin, X., Han, J., Yang, J., & Yu, P. (2006). Crossmine: Efficient classification across multiple database relations. *Transactions on Knowledge and Data Engineering*, *18*(6), 770–783.

Zheleva, E., Getoor, L., Golbeck, J., & Kuter, U. (2010). Using friendship ties and family circles for link prediction. In *Advances in Social Network Mining and Analysis*, pp. 97–113.

Zheleva, E., & Getoor, L. (2007). Preserving the privacy of sensitive relationships in graph data. In *PinKDD*, pp. 153–171.

Zhou, B., Pei, J., & Luk, W. (2008). A brief survey on anonymization techniques for privacy preserving publishing of social network data. *SIGKDD Explorations*, *10*(2), 12–22.

Zhou, H. (2003). Distance, dissimilarity index, and network community structure. *Physical review e*, *67*(6), 61901.

Zhou, T., Lü, L., & Zhang, Y. (2009). Predicting missing links via local information. *The European Physical Journal B-Condensed Matter and Complex Systems*, *71*(4), 623–630.

Zhu, S., Yu, K., Chi, Y., & Gong, Y. (2007). Combining content and link for classification using matrix factorization. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 487–494. ACM.

Zhu, X. (2006). Semi-supervised learning literature survey. *Computer Science Tech Reports*, *1530*, 1–60.