

# Label and Link Prediction in Relational Data

Ben Taskar Pieter Abbeel Ming-Fai Wong Daphne Koller

btaskar, abbeel, mingfai.wong, koller@cs.stanford.edu

Stanford University

Stanford, CA 94305

## Abstract

Many real-world domains are *relational* in nature, consisting of a set of entities linked to each other in complex ways. Two important tasks in such data are predicting entity labels and links between entities. We present a flexible framework that builds on (conditional) Markov networks and successfully addresses both tasks by capturing complex dependencies in the data. These models can compactly represent probabilistic patterns over subgraph structures and use them to predict labels and links effectively. We show how to train these models, and how to use approximate probabilistic inference over the learned model for collective classification of multiple related entities and links. We evaluate our framework on several relational datasets, including university webpages and social networks. Our approach achieves significantly better performance than flat classification, which attempts to predict each label and link in isolation.

## 1 Introduction

The vast majority of work in statistical classification methods has focused on “flat” data – data consisting of identically-structured entities, typically assumed to be independent and identically distributed (IID). However, many real-world data sets are innately relational: hyperlinked webpages, cross-citations in patents and scientific papers, social networks, medical records, and more. Such data consist of entities of many types, where each entity type is characterized by a different set of attributes. Entities are related to each other via different types of links, and the link structure is an important source of information.

Consider a collection of hypertext documents that we want to classify using some set of labels. For example, for a university website, we would like to predict which pages belong to a student, a professor, a course, etc. Most naively, we can use a bag of words model, classifying each webpage solely using the words that appear on the page. However, hypertext has a very rich structure that this approach loses entirely. One document has hyperlinks to others, typically indicating that their topics are related. Each document also has internal structure, such as a partition into sections; hyperlinks that emanate from the same section of the document are even more likely to point to similar documents. When classifying a collection of documents, these are important cues, that

can potentially help us achieve better classification accuracy. Therefore, rather than classifying each document separately, we want to provide a form of *collective classification*, where we simultaneously decide on the class labels of all of the entities together, and thereby can explicitly take advantage of the correlations between the labels of related entities.

Another challenge arises from the task of predicting which entities are related to which others and what are the types of these relationships. For example, we might also want to predict not just which page belongs to a professor and which to a student, but also which professor is which student’s advisor. In some cases, the existence of a relationship will be predicted by the presence of a hyperlink between the pages, and we will have only to decide whether the link reflects an advisor-advisee relationship. In other cases, we might have to infer the very existence of a link from indirect evidence, such as a large number of co-authored papers. In a very different application, we might want to predict links representing participation of individuals in certain terrorist activities.

One possible approach to this task is to consider the presence and/or type of the link using only attributes of the potentially linked entities and of the link itself. For example, in our university example, we might try to predict and classify the link using the words on the two webpages, and the anchor words on the link (if present). This approach has the advantage that it reduces to a simple classification task and we can apply standard machine learning techniques. However, it completely ignores a rich source of information that is unique to this task — the graph structure of the link graph. For example, a strong predictor of an advisor-advisee link between a professor and a student is the fact that they jointly participate in several projects. In general, the link graph typically reflects common patterns of interactions between the entities in the domain. Taking these patterns into consideration should allow us to provide a much better prediction for links.

A somewhat more sophisticated approach might be to build a link predictor that explicitly takes into consideration other relevant links in the graph. We can implement this either by converting graph features into attributes [10], or by explicitly learning a relational classifier, using techniques such as *inductive logic programming* [8]. Unfortunately, this approach is limited to cases where we are trying to predict a single link at a time, and the other links in the graph are given. In practice, only some or none of the links in the graph

are known, and we are trying to simultaneously predict a large number of links in the graph.

We propose the use of a joint probabilistic model for an entire collection of related entities and links. Following the approach of Lafferty [7], we base our approach on discriminatively trained undirected graphical models, or *Markov networks* [11]. We introduce the framework of *relational Markov network (RMNs)*, which compactly defines a Markov network over a relational data set. The graphical structure of an RMN is based on the relational structure of the domain, and can easily model complex patterns over related entities. For example, we can represent a pattern where two linked documents are likely to have the same topic. We can also capture patterns that involve groups of links: for example, consecutive links in a document tend to refer to documents with the same label. We can also represent “transitive” patterns, where the presence of links A to B and B to C and increases (or decreases) the likelihood of an A to C link. As we demonstrate, RMNs allow tremendous flexibility in representing complex patterns.

Undirected models lend themselves well to discriminative training, where we optimize the conditional likelihood of the labels given the features. Discriminative training, given sufficient data, generally provides significant improvements in classification accuracy over generative training [13]. We provide an effective parameter estimation algorithm for RMNs which uses conjugate gradient combined with approximate probabilistic inference (belief propagation [11]) for estimating the gradient. We also show how to use approximate probabilistic inference over the learned model for collective classification of multiple related entities. We provide experimental results on classification and relation type prediction tasks in web data and a link prediction task in a social network dataset, showing significant gains in accuracy arising from the modeling of relational dependencies.

## 2 Relational Domains

Consider hypertext as a simple example of a relational domain. A relational domain is defined by a schema, which describes entities, their attributes and relations between them. In our domain, there are two entity types: Page and Hyperlink. If a webpage is represented as a bag of words, Page would have a set of boolean attributes  $\text{Page.HasWord}_k$  indicating whether the word  $k$  occurs on the page. It would also have the label attribute  $\text{Page.Label}$ , indicating the topic of the page, which takes on a set of categorical values. The Hyperlink entity type has three attributes:  $\text{Hyperlink.Type}$ , indicating the type of relationship between the two pages,  $\text{Hyperlink.From}$  and  $\text{Hyperlink.To}$ , both of which refer to Page entities.

In general, a *schema* specifies of a set of entity types  $\mathcal{E} = \{E_1, \dots, E_n\}$ . Each type  $E$  is associated with three sets of attributes: content attributes  $E.X$  (e.g.  $\text{Page.HasWord}_k$ ), label attributes  $E.Y$  (e.g.  $\text{Page.Label}$ ), and reference attributes  $E.R$  (e.g.  $\text{Hyperlink.To}$ ). For simplicity, we restrict label and content attributes to take on categorical values. Reference attributes include a special unique key attribute  $E.K$  that identifies each entity. Other reference attributes  $E.R$  refer to entities of a single type  $E' = \text{Range}(E.R)$  and take values in  $\text{Domain}(E'.K)$ . Following [4] in addressing link ex-

istence prediction, we introduce into our schema object types that correspond to *potential* links between entities. Thus, if we want to reason about all possible links between entities in the model, we can introduce a potential link between every pair of entities in the domain. Each link object has a binary *existence* attribute  $\text{Exists}$ , which is *true* if the link between the associated entities exists and *false* otherwise.

An *instantiation*  $\mathcal{I}$  of a schema  $\mathcal{E}$  specifies the set of entities  $\mathcal{I}(E)$  of each entity type  $E \in \mathcal{E}$  and the values of all attributes for all of the entities. For example, an instantiation of the hypertext schema is a collection of webpages, specifying their labels, words they contain and links between them. We will use  $\mathcal{I}.X$ ,  $\mathcal{I}.Y$  and  $\mathcal{I}.R$  to denote the content, label and reference attributes in the instantiation  $\mathcal{I}$ ;  $\mathcal{I}.x$ ,  $\mathcal{I}.y$  and  $\mathcal{I}.r$  to denote the values of those attributes. The component  $\mathcal{I}.r$ , which we call an *instantiation graph*, specifies the set of entities (nodes) and their reference attributes (edges). A hypertext instantiation graph specifies a set of webpages and links between them, but not their words or labels.

## 3 Relational Markov Networks

In this section, we present the framework of undirected graphical models, also known as *Markov Networks* [11] or *Markov Random Fields*, and their extension to relational domains.

**Markov Networks.** Let  $\mathbf{V}$  denote a set of discrete random variables and  $\mathbf{v}$  an assignment of values to  $\mathbf{V}$ . A Markov network for  $\mathbf{V}$  defines a joint distribution over  $\mathbf{V}$ . It consists of an undirected dependency graph, and a set of parameters associated with the graph. For a graph  $G$ , a *clique* is a set of nodes  $\mathbf{V}_c$  in  $G$ , not necessarily maximal, such that each  $V_i, V_j \in \mathbf{V}_c$  are connected by an edge in  $G$ . Note that a single node is also considered a clique.

**Definition 1:** Let  $G = (\mathbf{V}, E)$  be an undirected graph with a set of cliques  $C(G)$ . Each  $c \in C(G)$  is associated with a set of nodes  $\mathbf{V}_c$  and a *clique potential*  $\phi_c(\mathbf{V}_c)$ , which is a non-negative function defined on the joint domain of  $\mathbf{V}_c$ . Let  $\Phi = \{\phi_c(\mathbf{V}_c)\}_{c \in C(G)}$ . The Markov net  $(G, \Phi)$  defines the distribution  $P(\mathbf{v}) = \frac{1}{Z} \prod_{c \in C(G)} \phi_c(\mathbf{v}_c)$ , where  $Z$  is the standard normalizing *partition function*. ■

Each potential  $\phi_c$  is simply a table of values for each assignment  $\mathbf{v}_c$  that defines a “compatibility” between values of variables in the clique. The potential is often represented compactly by a log-linear combination of a small set of indicator functions, or *features*, of the form  $f(\mathbf{V}_c) \equiv \delta(\mathbf{V}_c = \mathbf{v}_c)$ . In this case, the potential can be written as:  $\phi_c(\mathbf{v}_c) = \exp\{\sum_i w_i f_i(\mathbf{v}_c)\} = \exp\{\mathbf{w}_c \cdot \mathbf{f}_c(\mathbf{v}_c)\}$ .

For classification, we are interested in constructing discriminative models using *conditional Markov nets* (or conditional random fields [7]), which are simply Markov networks renormalized to model a conditional distribution of some set of target variables  $\mathbf{Y}$  given observed variables  $\mathbf{X}$ :  $P(\mathbf{y} | \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C(G)} \phi_c(\mathbf{y}_c, \mathbf{x}_c)$ , where  $Z(\mathbf{x})$  is the partition function, now dependent on  $\mathbf{x}$ .

**Relational Markov Networks.** A *relational Markov network (RMN)* [12] specifies the cliques and potentials between attributes of related entities at a template level, so a single

model provides a coherent distribution for any collection of instances from the schema. RMNs specify the cliques using *relational clique templates* to identify tuples of variables in the instantiation in a relational query language.

**Definition 2:** A *relational clique template*  $C = (\mathbf{F}, \mathbf{W}, \mathbf{S})$  consists of three components:

- $\mathbf{F} = \{F_i\}$  — a set of object variables, for which  $\epsilon(F_i)$  denotes the entity type of  $F_i$ .
- $\mathbf{W}(\mathbf{F}, \mathbf{R})$  — a boolean formula using conditions of the form  $F_i.R_j = F_k.R_l$ .
- $\mathbf{F}, \mathbf{S} \subseteq \mathbf{F}, \mathbf{X} \cup \mathbf{F}, \mathbf{Y}$  — a subset of attributes in  $\mathbf{F}$ .

For example, if we want to define cliques between the class labels of linked pages, to capture the tendency of pages with the same label tend to link to each other, as in Fig. 1, we might define:  $\mathbf{F}$  to be the set *page1, page2* and *link* of types *Page*, *Page* and *Hyperlink*, respectively;  $\mathbf{W}(\mathbf{F}, \mathbf{R})$  to be  $link.From = page1.Key \wedge link.To = page2.Key$ ; and  $\mathbf{F}, \mathbf{S}$  to be *page1.Category* and *page2.Category*.

A clique template specifies a set of *ground cliques* in an instantiation  $\mathcal{I}$ :

$$C(\mathcal{I}) \equiv \{c = \mathbf{e}, \mathbf{S} : \mathbf{e} \in \mathcal{I}(\mathbf{F}) \wedge \mathbf{W}(\mathbf{e}, \mathbf{r})\},$$

where  $\mathbf{e}$  is a tuple of entities  $\{e_i\}$  in which each  $e_i$  is of type  $\epsilon(F_i)$ ;  $\mathcal{I}(\mathbf{F}) = \mathcal{I}(\epsilon(F_1)) \times \dots \times \mathcal{I}(\epsilon(F_n))$  denotes the cross-product of entities in the instantiation; the clause  $\mathbf{W}(\mathbf{e}, \mathbf{r})$  ensures that the entities are related to each other in specified ways;  $\mathbf{e}, \mathbf{S}$  selects the appropriate attributes of the entities.

**Definition 3:** A *Relational Markov network (RMN)*  $\mathcal{M} = (\mathbf{C}, \Phi)$  specifies a set of clique templates  $\mathbf{C}$  and corresponding potentials  $\Phi = \{\phi_C\}_{C \in \mathbf{C}}$  to define a conditional distribution:

$$P(\mathcal{I}, \mathbf{Y} \mid \mathcal{I}, \mathbf{X}, \mathcal{I}, \mathbf{R}) = \frac{1}{Z} \prod_{C \in \mathbf{C}} \prod_{c \in C(\mathcal{I})} \phi_C(\mathcal{I}, \mathbf{Y}_c, \mathcal{I}, \mathbf{X}_c)$$

where  $Z = Z(\mathcal{I}, \mathbf{X}, \mathcal{I}, \mathbf{R})$  is the partition function. ■

Using the log-linear representation of potentials,  $\phi_C(\mathbf{V}_C) = \exp\{\mathbf{w}_C \cdot \mathbf{f}_C(\mathbf{V}_C)\}$ , we can write

$$\log P(\mathcal{I}, \mathbf{Y} \mid \mathcal{I}, \mathbf{X}, \mathcal{I}, \mathbf{R}) = \mathbf{w} \cdot \mathbf{f}(\mathcal{I}, \mathbf{Y}, \mathcal{I}, \mathbf{X}, \mathcal{I}, \mathbf{R}) - \log Z$$

where  $\mathbf{f}_C(\mathcal{I}, \mathbf{Y}, \mathcal{I}, \mathbf{X}, \mathcal{I}, \mathbf{R}) = \sum_{c \in C(\mathcal{I})} \mathbf{f}_C(\mathcal{I}, \mathbf{Y}_c, \mathcal{I}, \mathbf{X}_c)$  is the sum over all appearances of the template  $C(\mathcal{I})$  in the instantiation, and  $\mathbf{f}$  is the vector of all  $\mathbf{f}_C$ .

Given a particular instantiation  $\mathcal{I}$  of the schema, the RMN  $\mathcal{M}$  produces an *unrolled* Markov network over the attributes of entities in  $\mathcal{I}$ , in the obvious way. The cliques in the unrolled network are determined by the clique templates  $\mathbf{C}$ . We have one clique for each  $c \in C(\mathcal{I})$ , and all of these cliques are associated with the same clique potential  $\phi_C$ .

**Probabilistic Models of Graph Structure.** The combination of a relational language with a probabilistic graphical model provides a very flexible framework for modeling complex patterns common in relational graphs. First, as observed by Getoor *et al.* [4], there are often correlations between the attributes of entities and the relations in which they participate. For example, in a social network, people with the same hobby

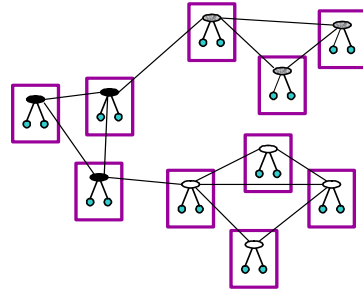


Figure 1: An unrolled Markov net over linked documents. The links follow a common pattern: documents with the same label tend to link to each other more often.

are more likely to be friends. More interestingly, we have correlations between the *labels* of entities and the relation type. For example, in a computer science department website, only students can be teaching assistants in a course. We can easily capture such correlations by introducing cliques that involve these attributes. Importantly, these cliques are informative even when attributes are not observed in the test data. For example, if we have evidence indicating an advisor-advisee relationship, our probability that X is a faculty member increases, and thereby our belief that X participates in a teaching assistant in a course Z decreases.

We can also represent much richer patterns over the link graph. Consider, for example, a professor X and two other entities Y and Z. If X’s webpage mentions Y and Z in the same context, it is likely that there is correlation between the type of relationship for X-Y and the type for Y-Z. For example, if Y is Professor X’s advisee, then probably so is Z. Our framework accommodates these patterns by introducing pairwise cliques between the appropriate relation variables.

Interactions are not limited to pairs of relations. “Transitive” patterns are also common, where the presence of an A-B link and of a B-C link increases (or decreases) the likelihood of an A-C link. For example, students often assist in courses taught by their advisor. Note that this type of interaction cannot be accounted for just using pairwise cliques. By introducing cliques over triples of relations, we can capture such patterns as well. We can incorporate even more complicated patterns, but of course we are limited by the ability of belief propagation to scale up as we introduce larger cliques and tighter loops in the Markov network.

We note that our ability to model these more complex graph patterns relies on our use of an undirected Markov network as our probabilistic model. In contrast, the approach of Getoor *et al.* uses directed graphical models (Bayesian networks and PRMs [6]) to represent a probabilistic model of both relations and attributes. While their approach captures the dependence of link existence on attributes of entities, the constraint that the probabilistic dependency graph be a directed acyclic graph prevents them from representing the more interesting correlations described above.

## 4 Learning the Models

We focus on the case where the clique templates are given; our task is to estimate the clique potentials, or feature

weights. Thus, assume that we are given a set of clique templates  $\mathbf{C}$  which partially specify our (relational) Markov network, and our task is to compute the weights  $\mathbf{w}$  for the potentials  $\Phi$ . In the learning task, we are given some training set  $D$  where both the content attributes and the labels are observed. Any particular setting for  $\mathbf{w}$  fully specifies a probability distribution  $P_{\mathbf{w}}$  over  $D$ , so we can use the *likelihood* as our objective function, and attempt to find the weight setting that maximizes the likelihood (ML) of the labels given other attributes. However, to help avoid overfitting, we assume a “shrinkage” prior over the weights (a zero-mean Gaussian), and use maximum a posteriori (MAP) estimation. More precisely, we assume that different parameters are a priori independent and define  $p(w_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-w_i^2/2\sigma^2\}$ .

Both the ML and MAP objective functions are concave and there are many methods available for maximizing them. Our experience is that conjugate gradient and even simple gradient perform very well for logistic regression and relational Markov nets.

**Learning Markov Networks.** We first consider discriminative MAP training in the flat setting. In this case  $D$  is simply a set of IID instances; let  $d$  index over all labeled training data  $D$ . The discriminative likelihood of the data is  $\prod_d P_{\mathbf{w}}(y_d | \mathbf{x}_d)$ . We introduce the parameter prior, and maximize the log of the resulting MAP objective function:

$$L(\mathbf{w}, D) = \sum_{d \in D} (\mathbf{w} \cdot \mathbf{f}(y_d, \mathbf{x}_d) - \log Z(\mathbf{x}_d)) - \frac{\mathbf{w} \cdot \mathbf{w}}{2\sigma^2} + C .$$

The gradient of the objective function is computed as:

$$\nabla L(\mathbf{w}, D) = \sum_{d \in D} (\mathbf{f}(y_d, \mathbf{x}_d) - E_{P_{\mathbf{w}}}[\mathbf{f}(Y_d, \mathbf{x}_d)]) - \frac{\mathbf{w}}{\sigma^2} .$$

The last term is the shrinking effect of the prior and the other two terms are the difference between the expected feature counts and the empirical feature counts, where the expectation is taken relative to  $P_{\mathbf{w}}$ :

$$E_{P_{\mathbf{w}}}[\mathbf{f}(Y_d, \mathbf{x}_d)] = \sum_{y'} \mathbf{f}(y'_d, \mathbf{x}_d) P_{\mathbf{w}}(y'_d | \mathbf{x}_d) .$$

Thus, ignoring the effect of the prior, the gradient is zero when empirical and expected feature counts are equal.<sup>1</sup> The prior term gives the smoothing we expect from the prior: small weights are preferred in order to reduce overfitting. Note that the sum over  $y'$  is just over the possible categorizations for one data sample every time.

**Learning RMNs.** The analysis for the relational setting is very similar. Now, our data set  $D$  is actually a single instantiation  $\mathcal{I}$ , where the same parameters are used multiple times — once for each different entity that uses a feature. A particular choice of parameters  $\mathbf{w}$  specifies a particular RMN, which induces a probability distribution  $P_{\mathbf{w}}$  over the unrolled Markov network. The product of the likelihood of  $\mathcal{I}$  and the

<sup>1</sup>The solution of maximum likelihood estimation with log-linear models is actually also the solution to the dual problem of maximum entropy estimation with constraints that empirical and expected feature counts must be equal [3].

parameter prior define our objective function, whose gradient  $\nabla L(\mathbf{w}, \mathcal{I})$  again consists of the empirical feature counts minus the expected features counts and a prior term:

$$\mathbf{f}(\mathcal{I}, \mathbf{y}, \mathcal{I}, \mathbf{x}, \mathcal{I}, \mathbf{r}) - E_{\mathbf{w}}[\mathbf{f}(\mathcal{I}, \mathbf{Y}, \mathcal{I}, \mathbf{x}, \mathcal{I}, \mathbf{r})] - \frac{\mathbf{w}}{\sigma^2}$$

where the expectation  $E_{P_{\mathbf{w}}}[\mathbf{f}(\mathcal{I}, \mathbf{Y}, \mathcal{I}, \mathbf{x}, \mathcal{I}, \mathbf{r})]$  is

$$\sum_{\mathcal{I}, \mathbf{y}'} \mathbf{f}(\mathcal{I}, \mathbf{y}', \mathcal{I}, \mathbf{x}, \mathcal{I}, \mathbf{r}) P_{\mathbf{w}}(\mathcal{I}, \mathbf{y}' | \mathcal{I}, \mathbf{x}, \mathcal{I}, \mathbf{r}) .$$

This last formula reveals a key difference between the relational and the flat case: the sum over  $\mathcal{I}, \mathbf{y}'$  involves the exponential number of assignments to all the label attributes in the instantiation. In the flat case, the probability decomposes as a product of probabilities for individual data instances, so we can compute the expected feature count for each instance separately. In the relational case, these labels are correlated — indeed, this correlation was our main goal in defining this model. Hence, we need to compute the expectation over the joint assignments to all the entities together. Computing these expectations over an exponentially large set is the expensive step in calculating the gradient. It requires that we run inference on the unrolled Markov network.

**Inference in Markov Networks.** The inference task in our conditional Markov networks is to compute the posterior distribution over the label variables in the instantiation given the content variables. Exact inference algorithms in graphical models can compute this distribution efficiently for specific graph topologies such as sequences, trees and other low treewidth graphs. However, the networks resulting from domains such as hypertext are very large (in our experiments, they contain tens of thousands of nodes) and densely connected. Exact inference is intractable in these cases.

We therefore resort to approximate inference. There is a wide variety of approximation schemes for Markov networks. We chose to use *belief propagation* for its simplicity and relative efficiency and accuracy. Belief Propagation (BP) is a local message passing algorithm introduced by Pearl [11]. It is guaranteed to converge to the correct marginal probabilities for each node only for singly connected Markov networks. Empirical results [9] show that it often converges in general networks, and when it does, the marginals are a good approximation to the correct posteriors. As our results in Section 5 show, this approach works well in our domain.

## 5 Experiments

We tested our framework on the standard *WebKB* dataset [2] as well as two new real-world datasets which we selected because of interesting relational structure not common to many publicly available datasets. The first new dataset is a WebKB-inspired collection of webpages from several university web sites labeled by rich set of entity and relation categories. The second dataset is a database of university students, including their personal information and lists of their friends.

### 5.1 WebKB

The data set contains webpages from four different Computer Science departments: Cornell, Texas, Washington and Wisconsin. Each page has a label attribute, representing the type

of webpage which is one of *course*, *faculty*, *student*, *project* or *other*. The data set is problematic in that the category *other* is a grab-bag of pages of many different types. The number of pages classified as *other* is quite large, so that a baseline algorithm that simply always selected *other* as the label would get an average accuracy of 75%. We could restrict attention to just the pages with the four other labels, but in a relational classification setting, the deleted webpages might be useful in terms of their interactions with other webpages. Hence, we compromised by eliminating all *other* pages with fewer than three outlinks, making the number of *other* pages commensurate with the other categories.<sup>2</sup> For each page, we have access to the entire html of the page and the links to other pages. Our goal is to collectively classify webpages into one of these five categories. In all of our experiments, we learn a model from three schools and test the performance of the learned model on the remaining school, thus evaluating the generalization performance of the different models.

**Flat Models.** The simplest approach we tried predicts the categories based on just the text content on the webpage. The text of the webpage is represented using a set of binary attributes that indicate the presence of different words on the page. We found that stemming and feature selection did not provide much benefit and simply pruned words that appeared in fewer than three documents in each of the three schools in the training data. We also experimented with incorporating meta-data: words appearing in the title of the page, in anchors of links to the page and in the last header before a link to the page [14]. Note that meta-data, although mostly originating from pages linking into the considered page, are easily incorporated as features, i.e. the resulting classification task is still flat feature-based classification. Our first experimental setup compares three well-known text classifiers — Naive Bayes, one-against-others linear support vector machines (Svm), and logistic regression (Logistic) — using words and meta-words. The two discriminative approaches outperform Naive Bayes by an average of nearly 10%. Logistic and Svm give very similar results. The average error over the 4 schools was reduced by around 4% by introducing the meta-data attributes.

**Relational Models.** Incorporating meta-data gives a significant improvement, but we can take additional advantage of the correlation in labels of related pages by classifying them collectively. We want to capture these correlations in our model and use them for transmitting information between linked pages to provide more accurate classification. We experimented with several relational models. Recall that logistic regression is simply a flat conditional Markov network. All of our relational Markov networks use a logistic regression model locally for each page.

Our first model captures direct correlations between labels of linked pages. These correlations are very common in our data: courses and research projects almost never link to each other; faculty rarely link to each other; students have links to

<sup>2</sup>The category distribution is: course (237), faculty (148), other (332), project (82) and student (542). The numbers of pages/links are: Cornell (280/574), Texas (292/574), Washington (315/728) and Wisconsin (454/1614).

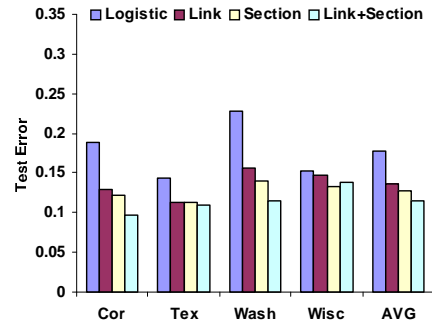


Figure 2: Flat versus collective classification on WebKB: flat logistic regression with meta-data, and three different relational models: Link, Section, and a combined Section+Link.

all categories but mostly courses. The Link model, shown in Fig. 1, captures this correlation through links: in addition to the local bag of words and meta-data attributes, we introduce a relational clique template over the labels of linked pages.

A second relational model uses the insight that a webpage often has internal structure that allows it to be broken up into *sections*. For example, a faculty webpage might have one section that discusses research, with a list of links to all of the projects of the faculty member, a second section might contain links to the courses taught by the faculty member, and a third to his advisees. We can view a section of a webpage as a fine-grained version of Kleinberg’s hub [5] (a page that contains a lot of links to pages of particular category). Intuitively, if we have links to two pages in the same section, they are likely to be on similar topics. To take advantage of this trend, we need to enrich our schema with a new relation **Section**, with attributes *Key*, *Page* (document in which it appears), and *Category*. We also add the attribute *Section* to **Hyperlink** to refer to the section it appears in. In the RMN, we have two new relational clique templates. The first contains the label of a section and the label of the page it is on:

```
SELECT page.Category, sec.Category
FROM Page page, Section sec
WHERE sec.Page = page.Key
```

The second clique template involves the label of the section containing the link and the label of the target page.

```
SELECT sec.Category, page.Category
FROM Section sec, Hyperlink link, Page page
WHERE link.Sec = sec.Key and link.To = page.Key
```

The original dataset did not contain section labels, so we introduced them using the following simple procedure. We defined a section as a sequence of three or more links that have the same path to the root in the html parse tree. In the training set, a section is labeled with the most frequent category of its links. There is a sixth category *none*, assigned when the two most frequent categories of the links are less than a factor of 2 apart. In the entire data set, the breakdown of labels for the sections we found is: *course* (40), *faculty* (24), *other* (187), *research.project* (11), *student* (71) and *none* (17). Note that these labels are hidden in the test data, so the learning algorithm now also has to learn to predict section labels. Although not our final aim, correct prediction of section

labels is very helpful. Words appearing in the last header before the section are used to better predict the section label by introducing a clique over these words and section labels.

We compared the performance of Link, Section and Section+Link (a combined model which uses both types of cliques) on the task of predicting webpage labels, relative to the baseline of flat logistic regression with meta-data. Our experiments used MAP estimation with a Gaussian prior on the feature weights with standard deviation of 0.3. Fig. 2 compares the average error achieved by the different models on the four schools, training on three and testing on the fourth. We see that incorporating any type of relational information consistently gives significant improvement over the baseline model. The Link model incorporates more relational interactions, but each is a weaker indicator. The Section model ignores links outside of coherent sections, but each of the links it includes is a very strong indicator. In general, we see that the Section models performs slightly better. The joint model is able to combine benefits from both and generally outperforms all of the other models. The only exception is for the task of classifying the Wisconsin data. In this case, the joint Section+Link model contains many links, as well as some large tightly connected loops, so belief propagation did not converge for a subset of nodes. Hence, the results of the inference, which was stopped at a fixed arbitrary number of iterations, were highly variable and resulted in lower accuracy.

## 5.2 Extended WebKB

We collected and manually labeled a new relational dataset inspired by WebKB [2]. Our dataset consists of Computer Science department webpages from 3 schools: Stanford, Berkeley, and MIT.

A total of 2954 of pages are labeled into one of eight categories: faculty, student, research scientist, staff, research group, research project, course and organization (organization refers to any large entity that is not a research group). *Owned pages*, which are owned by an entity but are not the main page for that entity, were manually assigned to that entity. The average distribution of classes across schools is: organization (9%), student (40%), research group (8%), faculty (11%), course (16%), research project (7%), research scientist (5%), and staff (3%).

We established a set of candidate links between entities based on evidence of a relation between them. One type of evidence for a relation is a hyperlink from an entity page or one of its owned pages to the page of another entity. A second type of evidence is a *virtual link*: We assigned a number of aliases to each page using the page title, the anchor text of incoming links, and email addresses of the entity involved. Mentioning an alias of a page on another page constitutes a virtual link. The resulting set of 7161 candidate links were labeled as corresponding to one of five relation types — Advisor (faculty, student), Member (research group/project, student/faculty/research scientist), Teach (faculty/research scientist/staff, course), TA (student, course), Part-Of (research group, research proj) — or “none”, denoting that the link does not correspond to any of these relations.

The observed attributes for each page are the words on the page itself and the “meta-words” on the page — the words

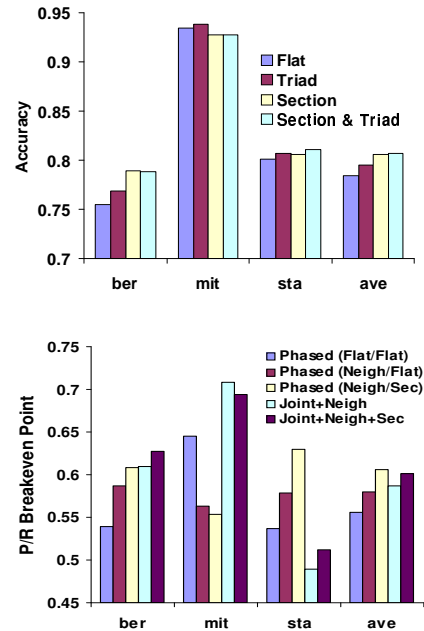


Figure 3: Top: Relation prediction with entity labels given. Bottom: Relation prediction without entity labels.

in the title, section headings, anchors to the page from other pages. For links, the observed attributes are the anchor text, text just before the link (hyperlink or virtual link), and the heading of the section in which the link appears.

Our task is to predict the relation type, if any, for all the candidate links. We tried two settings for our experiments: with page categories observed (in the test data) and page categories unobserved. For all our experiments, we trained on two schools and tested on the remaining school.

**Observed Entity Labels.** We first present results for the setting with observed page categories. Given the page labels, we can rule out many impossible relations; the resulting label breakdown among the candidate links is: none (38%), member (34%), part-of (4%), advisor (11%), teach (9%), TA (5%).

We tried several models. Link-Flat is our baseline model: it assumes relations are independent of each other, so we use multinomial logistic regression as our classifier. The features used by this model are the labels of the two linked pages and the words on the links going from one page and its owned pages to the other page. The number of features is  $\approx 1000$ .

The relational models try to improve upon the baseline model by modeling the interactions between relations and predicting relations jointly. The Section model introduces cliques over relations whose links appear consecutively in a section on a page. This model tries to capture the effect that similarly related entities (e.g., advisees, members of projects) are often listed together on a webpage. The Triad model tries to capture higher order patterns involving triples of related entities, as discussed in Section 3. Specifically, we introduce cliques over sets of three candidate links that form a triangle in the link graph. The Section + Triad model includes the cliques of the two models above.

As shown in Fig. 3 (top), both the Section and Triad mod-

els outperform the flat model, and the combined model has an average accuracy gain of 2.26%, or 10.5% relative reduction in error. As only a subset of all candidate links is affected by the section or triad cliques, we also computed the average accuracy gain on just the links involved in these richer cliques. We obtained an improvement of 3.78% on the 50% of links affected by **Section**.

As an example of the interesting inferences made by the models, we found a student-professor pair that was misclassified by **Flat** model as none (there is only a single hyperlink from the student’s page to the advisor’s) but correctly identified by both the **Section** and **Triad** models. The **Section** model utilizes a paragraph on the student’s webpage describing his research, with a section of links to his research groups and the link to his advisor. Examining the parameters of the **Section** model clique, we found that the model learned that it is likely for people to mention their research groups and advisors in the same section. By capturing this trend, the **Section** model is able to increase the confidence of the student-advisor relation. The **Triad** model corrects the same misclassification in a different way. Using the same example, the **Triad** model makes use of the information that both the student and the teacher belongs to the same research group, and the student TAed a class taught by his advisor. It is important to note that none of the other relations are observed in the test data, but rather the model bootstraps its inferences.

**Unobserved Entity Labels.** When the labels of pages are not known during relations prediction, we cannot rule out possible relations for candidate links based on the labels of participating entities. Thus, we have many more candidate links that do not correspond to any of our relation types (e.g., links between an organization and a student). This makes the existence of relations a very low probability event, with the following breakdown among the potential relations: none (71%), member (16%), part-of (2%), advisor (5%), teach (4%), TA (2%). In addition, when we construct a Markov network in which page labels are not observed, the network is much larger and denser, making the (approximate) inference task much harder. Thus, in addition to models that try to predict page entity and relation labels simultaneously, we also tried a two-phase approach, where we first predict page categories, and then use the predicted labels as features for the model that predicts relations.

For predicting page categories, we compared two models. **Entity-Flat** model is multinomial logistic regression that uses words and “meta-words” from the page and its owned pages in separate “bags” of words. The number of features is roughly 10,000. The **Neighbors** model is a relational model that also exploits regularities in web site organization: pages with similar urls often belong to the same category or tightly linked categories (research group/project, professor/course). For each page, two pages with urls closest in edit distance are selected as “neighbors”, and we introduced pairwise cliques between “neighboring” pages. The **Neighbors** model outperforms the **Flat** model across all schools by an average of 4.9% accuracy gain.

Given the page categories, we can now apply the different models for link classification. Thus, the **Phased (Flat/Flat)** model uses the **Entity-Flat** model to classify the page la-

els, and then the **Link-Flat** model to classify the candidate links using the resulting entity labels. The **Phased (Neighbors/Flat)** model uses the **Neighbors** model to classify the entity labels, and then the **Link-Flat** model to classify the links. The **Phased (Neighbors/Section)** model uses the **Neighbors** to classify the entity labels and then the **Section** model to classify the links.

We also tried two models that predict page and relation labels simultaneously. The **Joint + Neighbors** model is simply the union of the **Neighbors** model for page categories and the **Flat** model for relation labels given the page categories. The **Joint + Neighbors + Section** model additionally introduces the cliques that appeared in the **Section** model between links that appear consecutively in a section on a page. We train the joint models to predict page and link labels simultaneously.

Since the proportion of relation “none” is so overwhelming, we use the following decision rule to classify relations: If the probability of “none” is less than a given threshold, predict the most likely label (other than none), otherwise predict the most likely label (including none). We report precision recall breakeven point using this rule with the threshold set to a value where precision of actual relations (of all types except none) equals recall on the test data. Fig. 3 (bottom) compares the resulting breakeven points achieved by the different models on the three schools. Relational models, both phased and joint, did better than flat models on the average. However, performance varies from school to school and for both joint and phased models, performance on one of the schools is worse than that of the flat model.

### 5.3 Social Network Data

The second dataset we used has been collected by a portal website at Stanford university that hosts an online community for students [1]. Among other services, it allows students to enter information about themselves, create lists of their friends and browse the social network. Personal information includes residence, gender, major and year, as well as favorite sports, music, books, social activities, etc. We focused on the task of predicting the “friendship” links between students from their personal information and a subset of their links. We selected students living in sixteen different residences or dorms and restricted the data to the friendship links only within each residence. Each residence has about 15–25 students and an average student lists about 25% of housemates as friends.

We used an eight-fold train-test split, where we trained on fourteen residences and tested on two. Note that the students in the training and test set are disjoint and no links between them exist. Predicting links between two students from just personal information alone is a very difficult task, so we tried a more realistic setting, where some proportion of the links is observed in the test data, and can be used as evidence for predicting the remaining links. We used the following proportions of observed links in the test data: 10%, 25%, and 50%. The observed links were selected at random, and the results we report are averaged over five folds of these random selection trials.

Using just the observed portion of links, we constructed the following features: for each student, the proportion of stu-

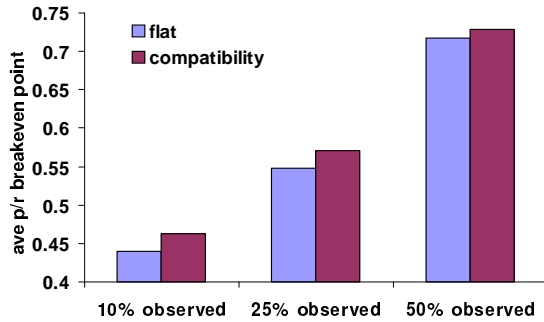


Figure 4: Average precision/recall breakeven point for 10%, 25%, 50% observed links.

dents in the residence that list him/her and the proportion of students he/she lists; for each pair of students, the proportion of other students they have as common friends. The values of the proportions were discretized into four bins. These features capture some of the relational structure and dependencies between links: Students who list (or are listed by) many friends in the observed portion of the links tend to have links in the unobserved portion as well. More importantly, having friends in common increases the likelihood of a link between a pair of students.

The Flat model uses logistic regression with the above features as well as personal information about each user. In addition to characteristics of the two people, we also introduced a feature for each match of a characteristic, e.g. both people are computer science majors or both are freshmen.

The Compatibility model (which resembles our Section model above) introduces cliques between each pair of links emanating from each person. Similarly to the Flat model, these cliques include a feature for each match of the characteristics of the two potential friends. This model captures the tendency of a person to have friends who share many characteristics (even though the person might not possess them). We also tried models with triad cliques, but the belief propagation often failed to converge, producing erratic results.

Fig. 4 compares the average precision/recall breakpoint achieved by the different models at the three different settings of observed links. The Compatibility model outperforms the flat with p-values 0.0036, 0.00064 and 0.054 respectively, according to a paired t-test.

## 6 Discussion and Conclusions

We address the novel task of collective label and link classification, where we are simultaneously trying to predict and classify an entire set of labels and links in a link graph. Our approach provides a coherent probabilistic foundation for this task, by defining a joint probabilistic model over objects and links. Our framework allows us to represent a very rich set of relational patterns in the probabilistic model, and use them in prediction. The resulting models significantly improve the classification accuracy over flat models.

Our results in this paper are only a first step toward understanding the power of relational classification, and many extensions are possible. On the representational side, we can gain significant power from introducing hidden variables (that

are not observed even in the training data). In a different extension, one of the problems limiting the applicability of approach is the reliance on belief propagation, which often does not converge in more complex problems. We believe that this issue can be addressed if we consider a tighter integration of learning and inference.

Our results use a set of relational patterns that we have discovered to be useful in the domains that we have considered. However, many other rich and interesting patterns are possible. Thus, in the relational setting, the issue of feature construction is critical. It is therefore important to explore the problem of automatic feature induction, as in [3].

Finally, we believe that this framework can provide a principled approach for addressing a wide range of applications, including predicting communities of people and hierarchical structure of people and organizations, based on both the local attributes and the patterns of static and dynamic interaction.

## References

- [1] L. Adamic, O. Buyukkocuten, and E. Adar. A social network caught in the web. <http://www.hpl.hp.com/shl/papers/social/>, 2002.
- [2] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the world wide web. In *Proc AAAI98*, pages 509–516, 1998.
- [3] S. Della Pietra, V. Della Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997.
- [4] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Probabilistic models of relational structure. In *Proc. ICML01*, Williamstown, MA., 2001.
- [5] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. of the ACM*, 46(5):604–632, 1999.
- [6] D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proc. AAAI98*, pages 580–587, Madison, Wisc., 1998.
- [7] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML01*, 2001.
- [8] Nada Lavrač and Saso Džeroski. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, New York, New York, 1994.
- [9] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: an empirical study. In *Proc. UAI99*, pages 467–475, 1999.
- [10] J. Neville and D. Jensen. Iterative classification in relational data. In *Proc. AAAI00 Workshop on Learning Statistical Models from Relational Data*, pages 13–20, 2000.
- [11] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Francisco, 1988.
- [12] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proc. UAI02*, Edmonton, Canada, 2002.
- [13] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, New York, 1995.
- [14] Y. Yang, S. Slattery, and R. Ghani. A study of approaches to hypertext categorization. *J. of Intelligent Information Systems*, 18(2), 2002.