Learning Contextual Dependency Network Models for Link-Based Classification

Yonghong Tian, *Member*, *IEEE*, Qiang Yang, *Senior Member*, *IEEE*, Tiejun Huang, *Member*, *IEEE*, Charles X. Ling, *Member*, *IEEE*, and Wen Gao, *Senior Member*, *IEEE*

Abstract—Links among objects contain rich semantics that can be very helpful in classifying the objects. However, many irrelevant links can be found in real-world link data such as Web pages. Often, these noisy and irrelevant links do not provide useful and predictive information for categorization. It is thus important to automatically identify which links are most relevant for categorization. In this paper, we present a contextual dependency network (CDN) model for classifying linked objects in the presence of noisy and irrelevant links. The CDN model makes use of a dependency function that characterizes the contextual dependencies among linked objects. In this way, CDNs can differentiate the impacts of the related objects on the classification and consequently reduce the effect of irrelevant links on the classification. We show how to learn the CDN model effectively and how to use the Gibbs inference framework over the learned model for collective classification of multiple linked objects. The experiments show that the CDN model demonstrates relatively high robustness on data sets containing irrelevant links.

Index Terms—Data dependencies, hypertext/hypermedia, machine learning, link-based classification, link context, contextual dependency networks, Gibbs inference.

1 Introduction

Many real-world data sets are richly interconnected. Typical examples include the Web, hypertext, and bibliographic data. Links among objects may demonstrate certain patterns. For example, documents with the same or related topics tend to link to each other more often. When classifying a collection of documents, these important clues can be potentially helpful for achieving better classification accuracy. Many researchers (e.g., [1]) have pointed out that a naive application of standard statistical classification procedures, which typically assume the instances are independent and identically distributed (i.i.d), would lead to inappropriate conclusions. What is important is to be able to exploit the dependencies between linked objects to improve the predictive accuracy of the learned model. As in [1], we also refer to the classification models that exploit the dependencies endowed by the links among objects as *link-based classification*.

Intuitively, link-based classification attempts to propagate the beliefs about one object to influence others to which it is related. Exactly following this idea, an iterative classification scheme can be used to improve accuracy by exploiting the inferred class labels of related instances (e.g., [4], [22]). Several relational models were proposed to characterize the correlation between link data to aid classification, e.g., probabilistic relational models (PRMs) [3], [7], relational Markov networks (RMNs) [8], and relational dependency networks (RDNs) [9], [10]. These models allow the specification of a probability model for types of objects, and also allow attributes of an entity to depend probabilistically on attributes of other related entities.

The real world is complex. Links can be from an object to another object of the same topic, or they can point at objects of different topics. The latter are sometimes considered as legitimate and at other times referred to as "noise" when they do not provide useful and predictive information for categorization. Ideally, we would like the learned relational models to be both predictive and general, so that not only clean data sets can be reasoned about, but also the ones that contain different types of links. Towards this end, this paper proposes a *contextual dependency network* (CDN) model. Similar to RDNs, the CDN model also uses dependency networks (DNs) [13] for modeling relational data. On top of the DN framework, we introduce additional parameters called *dependency functions* to directly capture the strengths of dependencies among linked objects. In this way, CDNs can differentiate between the impacts of the related objects on the classification, and thus effectively reduce the effect of irrelevant links on the classification.

The CDN learning algorithm is, in principle, based on pseudolikelihood techniques. Our aim is to avoid the complexities of estimating a full joint distribution. In particular, we view the learning process of CDNs as a dynamic-interacting process of multiple Markov chains, and use a self-mapping transformation algorithm to learn a set of relational conditional probability distributions (CPDs). We also show how to use Gibbs inference over the learned model for a collective classification of multiple linked objects.

- *Y. Tian is with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, China. E-mail: yhtian@ict.ac.cn.*
- *Q. Yang is with the Department of Computer Science, Hong Kong University of Science and Technology, Clearwater Bay, Kowloon, Hong Kong, China. E-mail: qyang@cs.ust.hk.*
- *T. Huang and W. Gao are with the Institute of Digital Media, Peking University, Beijing 100871, China. E-mail: {tjhuang, wgao}@pku.edu.cn.*
- *C.X. Ling is with the Department of Computer Science, University of Western Ontario (UWO), London, Ontario N6A 5B7, Canada. E-mail: cling@csd.uwo.ca.*

Manuscript received 1 Apr. 2005; revised 6 Dec. 2005; accepted 23 May 2006; published online 19 Sept. 2006.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0142-0405.

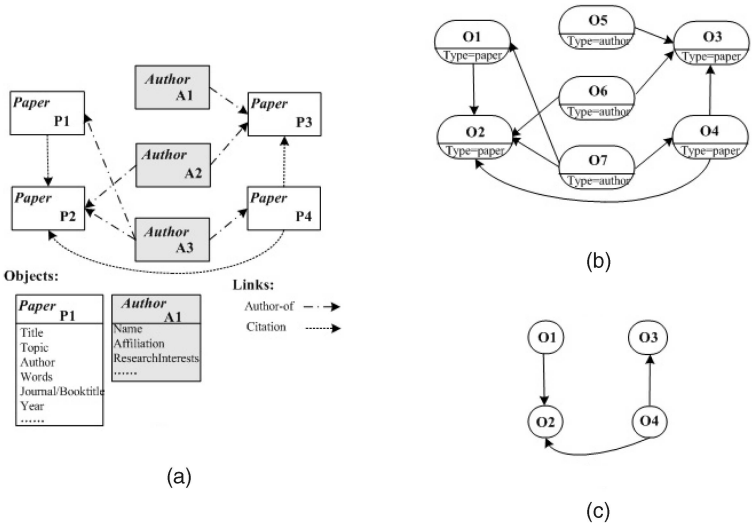


Fig. 1. A simple example of link data and its link graphs. (a) The data. (b) The link graph for all objects, where each object is associated with a type (e.g., paper and author). (c) The link graph for objects with $\text{type} = paper$.

To demonstrate the robustness of CDNs, we construct several subsets from Cora [16] and WebKB [17], each of which contains different percentages of irrelevant links. We experimentally compare the classification performance of CDNs with RDNs and two baseline link-based models. The experimental results show that CDNs provide a statistically significant improvement in accuracy and show relatively high robustness on the data sets containing some noisy links. These promising results indicate that the CDN models can scale well in the presence of noise.

The paper is organized as follows: Starting with a discussion of related work in Section 2, we present the CDN model in Section 3 and describe how to model contextual dependencies in link data in Section 4. In Section 5, we describe how the parameter estimation is performed for the model. Experiments and results are presented in Section 6. We conclude the paper in Section 7.

2 Related Work

In this section, we give a brief review of related work. To do so, we begin with some notations. We denote a variable by an italic uppercase token (e.g., A and X_i) and a value of that variable by the same token in lowercase (e.g., a, x_i, and θ). We denote matrices and sets by bold-face capitalized tokens (e.g., $\mathbf{A}$ and Σ) and vectors by bold-face lower case tokens (e.g., $\mathbf{a}$, $\mathbf{a}_i$, and σ). Finally, we use calligraphic tokens (e.g., $\mathcal{G}$ and $\mathcal{M}$) to denote statistical models and graphs.

A link data set can be represented as a directed (or undirected) graph $\mathcal{G}_{\mathrm{D}} = (\mathcal{O}_{\mathrm{D}}, \mathcal{L}_{\mathrm{D}})$ (called link graph), where the node $o_i \in \mathcal{O}_{\mathrm{D}}$ denotes an object (e.g., authors and papers) and the edge $o_i \rightarrow o_j \in \mathcal{L}_{\mathrm{D}}$ denotes a link from o_i to o_j (e.g., author-of and citation). More generally, link data can be viewed as an instantiation of a relational schema $\mathcal{S}$, where entities are interconnected. A schema $\mathcal{S}$ specifies a set of object types $\mathbf{T}$. Each object type $t \in \mathbf{T}$ is associated with a set of attributes. For example, consider the link data in Fig. 1a. A link graph for all objects is shown in Fig. 1b, where each object is associated with a type $\mathbf{T}(o_i) = t_{o_i}$ (e.g., *paper* and *author*). If a paper is represented as a bag of words, the type *paper* would have a set of real-value attributes $Words_k$ indicating the frequency that the word k occurs in the paper. It would also have some other attributes such as *Title* and *Journal/BookTitle*. For classification, each object o_i is characterized by a set of variables that include a single probabilistic variable C_i (a class label) and several other attributes whose values are known with certainty. For example, *paper* has the label attribute *Topic* and *author* has the label attribute *ResearchInterest*. Clearly, different object types may have different sets of attributes and different value domains of class labels. Sometimes, we are only interested in the link data with a special type of objects. For example, the link graph for objects with $\text{type} = paper$ is shown in Fig. 1c, which is also known as the citation graph in the bibliographic domain. In [22], this kind of link data is referred to as *networked data*.

In general, attributes of an object in link data can depend probabilistically on other attributes of the same object and on attributes of other linked objects in $\mathcal{G}_{\mathrm{D}}$. For example, the topic of a paper may be influenced by other attributes of the paper (e.g., *Words* and *Journal/BookTitle*), attributes (e.g., *Topic*) of other papers that are cited by or cite the paper, as well as attributes (e.g., *ResearchInterests*) of its authors. Furthermore, objects not directly linked may be related by chains of links, which suggests that it may be beneficial to use *collective inference* procedure [30] to make simultaneous statistical judgments regarding the values of attributes for multiple objects in $\mathcal{G}_{\mathrm{D}}$ for which some attribute values are not known. Then, the collective classification task is to construct a joint probability model over all the values of the class labels.

Recently, many algorithms and models (e.g., [4], [5], [6], [20], [21], [27]) have been proposed for classification of link data by incorporating relational feature representation and generation into traditional machine learning algorithms. Since they apply combinations of classifiers and relational feature extractors in a procedural way, we refer to them as combinative relational classifiers (CRCs) in this paper. Several researchers also proposed statistical relational

models (SRMs)[1] (e.g., [7], [8], [10], [12], [19]) to characterize the correlation between link data. As mentioned by Taskar et al. [8], none of CRCs provide a coherent model for the correlations between link data. Nevertheless, CRCs are often used as the baseline models for link-based classification due to the implementation simplicity. Thus, we simply review the two types of models in the following subsections.

2.1 Combinative Relational Classifiers (CRCs)

For link-based classification, the watershed paper of Chakrabarti et al. [4] studied classifying Web pages based on the text and (possibly inferred) class labels of neighboring pages, using naïve Bayes local and relational classifiers. It exploited system-predicted labels of linked documents to iteratively relabel each document in the test set, achieving a significant improvement compared to a baseline of using the text in each document alone. Due to the critical role, this paper uses this model as one baseline for link-based classification, and refers to it as the neighborhood iterative classification (NIC) model for convenience. Notice that NIC is also used as one of relational classifiers in the NetKit-SRL toolkit [22].

By explicitly learning how the link distribution affects the category, Lu and Getoor [6] proposed a linkage logistic regression (LLR) model that supports classification using discriminatively trained models. That is, the relational component of LLR is a logistic regression model based on a vector of aggregations of attributes of neighboring nodes linked with different types of links (in, out, colinks). Since LLR is a typical representative of discriminatively trained CRC models, this paper also uses it as another baseline link-based model.

Some other researchers also proposed different CRC models by integrating relational feature extractors into different machine-learning algorithms. For example, structural logistic regression (SLR) [21] integrates regression with feature generation from relational data; relational Bayesian classifier (RBC) [27] extends a simple Bayesian classifier to relational data by using four approaches (i.e., average value, random value, independent value, and average probability) to estimate conditional probabilities for each attribute of each type of objects; relational probability tree (RPT) [5] estimates probability distributions over class label values in the same manner as conventional classification trees, but also considers the effects of attributes in the local relational neighborhood on the probability distribution; relational Neural Network (RNN) [20] extends the feed-forward networks for relational learning by taking the attribute values of not only the entities itself, but also of sets of related entities as inputs. There are certainly many more CRC models, which cannot be listed here completely.

2.2 Statistical Relational Models (SRMs)

Possibly, the best well-known SRM up to now is the probabilistic relational model (PRM) proposed by Getoor et al. [3], [7]. PRMs extend Bayesian networks (BNs) to support reasoning in relational domains, allowing the rich relational information to be incorporated into the dependency structure of traditional BNs. Given a link graph, a PRM induces a larger BN over that graph, and defines a full joint probability distribution over all attributes of the objects. PRMs can also model uncertainty over both objects and link existence. However, the application of PRMs to some domains such as Web pages is problematic since there are many cycles in the graph, leading to cycles in the induced BN [8]. Moreover, PRMs require specifying a complete conditional model for each attribute of each entity type, which in large complex domains can be quite burdensome. In the same way that PRMs extend BNs, Heckerman et al. [19] also extended the entity-relationship (ER) model in database domain to the probabilistic entity-relationship (PER) model for relational data. However, they did not provide further details on issues of learning such models from data or of performing probabilistic inference with such models.

To avoid the acyclicity constraint, Taskar et al. [8] proposed a discriminatively trained undirected graphical model, called relational Markov networks (RMNs), to model the dependencies in relational data. By introducing a "clique" between the labels of two endpoints for each link, a RMN specifies the cliques and potentials between attributes of related entities and then defines a conditional distribution over all labels of all entities in a graph. To further improve the scalability of RMNs, Markov Logic Networks (MLNs) [12] combine the first-order logic and probabilistic graphical models in a single representation by associating a weight with each formula that reflects how strong a constraint it is. However, the clique potentials of Markov networks are generally difficult to inspect and understand in many relational data sets [9]. Moreover, the noisy links can be troublesome for discriminative models [21].

Neville and Jensen extended dependency networks (DNs) to relational settings and proposed the relational dependency networks (RDNs) as a collective classification model [9], [10]. RDNs use a selective relational classification algorithm (i.e., RPT) to learn a set of CPDs, and use Gibbs sampling for inference. Their experiments showed that RDNs could effectively infer labels for a network of instances. However, RDNs did not incorporate the semantic information of links into the probabilistic models. As a result, there is a difficulty in applying RDNs for classifying data sets containing some irrelevant or noisy links.

Although graphically quite different, these models are similar in their ability to represent or express probabilistic relationships in relational data and allow attributes of an entity to depend probabilistically on attributes of other related entities. Compared to CRCs, SRMs provide a coherent model for the correlations between link data, and thus can be used for a wider range of learning tasks on link data.

2.3 Noisy Links and Motivation to CDNs

The link regularities in many real-world data are very complex. Yang et al. [2] identified six hypertext regularities that might hold in a particular hypertext corpus. We focus on three regularities and extend them for the generic link data domain:[2]

1. Several previous papers (e.g., [10]) used the term PRM to denote the general statistical models for relational data. In this paper, we use PRM to denote a special type of model proposed by [3], [7], and use SRM to denote the general statistical models for relational data.

2. The other three regularities are *no regularity*, *preclassified regularity*, and *metadata regularity* [2]. If a data set has no regularity, then we do not expect to be able to use links to build better classifiers. And, the preclassified and metadata regularities can be found more often in a hypertext corpus. Therefore, we do not include them in this paper.

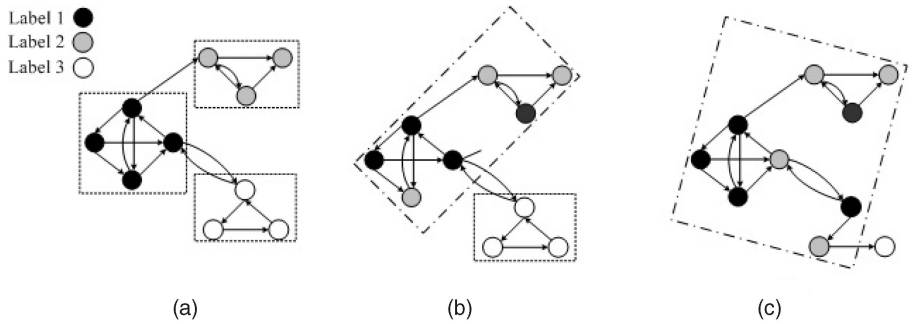


Fig. 2. The link data with (a) "encyclopedia" regularity, (b) "coreferencing" regularity, and (c) "partial coreferencing" regularity. In the figure, the linked nodes in a dotted rectangle have the same class label, and the linked nodes in a dashdotted rectangle may have semantically related class labels.

- **"Encyclopedia" regularity.** The class of an object is the same as the class of the majority of the linked objects.
- **"Coreferencing" regularity.** Objects with the same class tend to link to objects not of that class, but which are semantically related to each other.
- **"Partial coreferencing" regularity.** Objects with the same class tend to link to objects that are semantically related to each others, but also link to a wide variety of other objects without semantic reason.

Fig. 2 compares the three kinds of link regularities graphically. Clearly, links of the first two types convey more reinforcing information for the objects on the links, whereas links of the third, partial coreferencing regularity type are less informative. However, even when links from a Web object point at other objects of unrelated classes, these links can sometimes also provide additional information for classification of the objects in question. Instead of eliminating these links outright, the approach that we take in CDNs is to weigh these links differently through contextual dependency functions. We can learn the parameters of these functions from the training data set, rather than deleting those links. In order to distinguish the difference between these three kinds of links, we extend RDNs to contextual dependency networks (CDNs) for modeling the contextual dependencies in the link structure. Compared with RDNs, our CDN model has the following characteristics. First, in CDN models, we use the appropriately shaped dependency functions to weight dependencies among objects in the link structure in order to reduce the effect of irrelevant links on the classification. Second, we use CDNs to define the dependency structures at both class-based and instance-based levels, allowing them to model complex relational patterns in link graphs for a wide range of link data. Third, CDNs can be easily extended to a dynamic model for modeling the phenomena that relations among objects change over time [26].

3 Contextual Dependency Networks

As pointed out by Neville and Jensen [10], several characteristics of DNs are particularly desirable for modeling relational data, including the ability to represent cyclic dependencies, the simple technique for parameter estimation and structure learning, and the ability to easily interpret and understand. In this section, we begin by reviewing the details of DNs and then describe how to extend them to contextual dependency networks (CDNs).

3.1 Dependency Networks

Dependency networks (DNs) introduced by Heckerman et al. [13] are graphical models of probabilistic relationships that are similar to BNs. They differ in that the graphical structures of DNs are not required to be acyclic. A DN $\mathcal{D} = (\mathcal{G}, \mathbf{P})$ encodes the conditional independence constraints that each node is independent of all other nodes in $\mathbf{X}$ given its parents, where the direct graph $\mathcal{G}$ encodes the dependency structure of the model and $\mathbf{P}$ is a set of CPDs satisfying

$$p(X_i|\mathbf{Pa}_i) = p(X_i|\mathbf{X}\backslash X_i) \quad (1)$$

for each node $X_i \in \mathbf{X}$, where $\mathbf{Pa}_i$ specifies the parents of variable X_i. Notice that each node in $\mathcal{G}$ corresponds to a variable in $\mathbf{X}$. The edges connect a node (that is, a variable) to the other connected variables in a CPD to define the probabilistic dependency among them.

An advantage of DNs is that they are generally easier to learn from complete data than BNs. Namely, we can learn the CPD for each node independently using any standard classification or regression algorithm [13]. Both structure learning and parameter estimation of DNs are accomplished through learning the CPD of each variable. However, learning each CPD independently may result in an inconsistent network. That is, there may be no joint distribution from which each CPD can be obtained using the rules of probability. For example, a DN that contains a directed edge from X_i to X_j, but not from X_j to X_i, is inconsistent—X_i and X_j are dependent but X_j is not represented in the CPD for X_i. However, Heckerman et al. [13] show that DNs are "nearly" consistent if they are learned from reasonably large data sets because each DN is learned from the same set of joint data. For inference on DNs, Gibbs sampling can be used to recover a full joint distribution for $\mathbf{X}$ and extract probabilities of interest. For more details about DNs, we refer the reader to [13].

RDNs [9], [10] extend DNs to a relational setting. RDNs use a bidirected model graph $\mathcal{G}_{\mathrm{M}}$ with a set of CPDs $\mathbf{P}$. Each node in $\mathcal{G}_{\mathrm{M}}$ corresponds to an attribute object A_i^t and is associated with a CPD $p(a_i^t|\mathbf{Pa}(a_i^t))$. The RDN learning algorithm is much like the DN learning algorithm, except it uses RPTs to learn a set of CPDs [10]. RPTs can adjust for biases towards particular features due to autocorrelation in

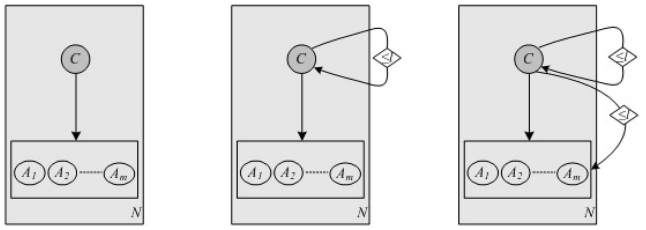


Fig. 3. The relational models for collective inference on link data (Modified from [30]). (a) The intrinsic model. (b) The collective inference (CI) model that allows interdependence among class labels of linked objects. (c) The relational collective inference (RCI) model that extends the CI model by allowing dependency among the class label of an object and the attributes of related objects. In the figure, N denotes the size of the data set, and l denotes the predefined link length.

relational data [5]. In the inference, the set of template CPDs in $\mathbf{P}$ is rolled-out over the link graph. Each object-attribute pair gets a separate, local copy of the appropriate CPD. Thus, given $\mathcal{G}_{\mathrm{M}}$ and $\mathbf{P}$, the full joint distribution over the unknown label variables in $\mathcal{G}_{\mathrm{D}}$ can be expressed as follows (via Gibbs sampling):

$$P(\mathcal{G}_{\mathrm{D}}|\mathcal{M}) = \prod_{t\in\mathbf{T}} \prod_{o_i\in\mathbf{I}(t)} P(C_i|\mathbf{Pa}(C_i)), \tag{2}$$

where $\mathcal{M} = (\mathcal{G}_{\mathrm{M}}, \mathbf{P})$ is the RDN model, $\mathbf{T}$ is the type set of objects in $\mathcal{G}_{\mathrm{D}}$, and $\mathbf{I}(t)$ is the object set of type $t \in \mathbf{T}$ in $\mathcal{G}_{\mathrm{D}}$. Notice that for different types of objects o_i and o_j, C_i and C_j may have different value domains.

However, we observe that the link structure is not a part of the probabilistic model of RDNs. As a matter of fact, by incorporating the links into the probabilistic model, we can both predict links and, more importantly, use the links to help us make prediction about other attributes in the model [7]. Consider the example illustrated in Fig. 1. The number and semantic strength (i.e., from very relevant to irrelevant) of citations made by each paper is outside the probabilistic model, but they are important for inferring the topic of the citing paper. Thus, in order to differentiate between the impacts of links on the classification, a more flexible strategy should be introduced to incorporate the semantics of links into the probabilistic model or, equivalently, capture the different strengths of contextual dependencies among linked objects in the model.

3.2 CDN Models

We now describe CDN models formally. To do so, we first see Fig. 3, which depicts graphically the relational models for collective inference on link data by using the plate notation. As shown in Fig. 3a, the intrinsic model expresses the probabilistic relationships among the class label of an object and its attributes for the i.i.d data (called *intrinsic dependency*). Compared with the intrinsic model, there are two kinds of relational models to capture the relational dependency in link data, i.e., *collective inference* (CI) model and *relational collective inference* (RCI) model [30]. Both models allow interdependence among class labels of (directly or indirectly) connected objects ($\leq l$ links away). The difference among them is that the RCI model adds dependency between the class label of an object and the attributes of related objects.

Typically, RDNs specify a single CPD for the class label of an object with type $t \in \mathbf{T}$ given both other attributes of that object and class labels (and attributes) of other related objects. As an alternative, CDNs define two CPDs for the class label of an object with type $t \in \mathbf{T}$: one for capturing intrinsic dependency and the other for capturing relational dependency. More formally,

$$\begin{aligned} P(C_i|\mathbf{Pa}(C_i)) = {} & \alpha_i P(C_i|\mathbf{Pa}^{(\mathrm{L})}(C_i)) \\ & + (1-\alpha_i) P(C_i|\mathbf{Pa}^{(\mathrm{N})}(C_i)), \end{aligned} \tag{3}$$

where $\mathbf{Pa}^{(\mathrm{L})}(C_i)$ denotes the "local" parents of C_i (i.e., attributes in $\mathbf{Pa}^{(\mathrm{L})}(C_i)$ are associated with object o_i), $\mathbf{Pa}^{(\mathrm{N})}(C_i)$ denotes the "networked" parents of C_i (i.e., attributes in $\mathbf{Pa}^{(\mathrm{N})}(C_i)$ are associated with objects in $\mathcal{G}_{\mathrm{D}}$ that are related to o_i). For convenience, we refer to $P(C_i|\mathbf{Pa}^{(\mathrm{L})}(C_i))$ as *intrinsic* CPDs, $P(C_i|\mathbf{Pa}^{(\mathrm{N})}(C_i))$ as *relational* CPDs, and accordingly $P(C_i|\mathbf{Pa}(C_i))$ as *full* CPDs or directly CPDs for short. As in [24], α_i is a scalar with $0 \leq \alpha_i \leq 1$ that measures how self-reliant o_i is. Namely, α_i captures the strength of the intrinsic dependency for each object o_i. To make the parameters tractable, we often define a single self-reliant factor α_t for each type $t \in \mathbf{T}$ of objects.

The second assumption of CDNs is to introduce some parameters to directly capture the different strengths of contextual dependencies among linked objects. We refer to these parameters as *dependency functions*, which will be described in the next section. Thus, the relational CPD $P(C_i|\mathbf{Pa}^{(\mathrm{N})}(C_i))$ is then expressed as

$$P(C_i|\mathbf{Pa}^{(\mathrm{N})}(C_i)) = \sum_{o_{ik}\in\mathbf{Pa}(o_i)} \sigma_{i,ik} P(C_i|\mathbf{Pa}_{ik}^{(\mathrm{N})}(C_i)), \tag{4}$$

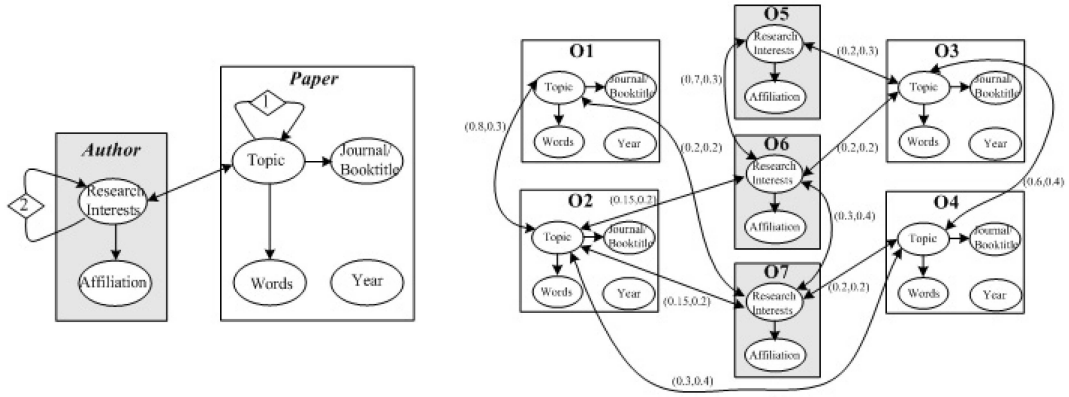
where $\mathbf{Pa}(o_i)$ is the parent set of object o_i in $\mathcal{G}_{\mathrm{D}}$, $\mathbf{Pa}_{ik}^{(\mathrm{N})}(C_i)$ is the parent set of C_i in attributes of object $o_{ik} \in \mathbf{Pa}(o_i)$, and $\sigma_{i,ik}$ is the dependency function of o_i on o_{ik} with $\sigma_{i,ik} \geq 0$ and $\sum_{o_{ik}\in\mathbf{Pa}(o_i)} \sigma_{i,ik} = 1$. Here, $\sigma_{i,ik}$ is used to measure how much $\mathbf{Pa}_{ik}^{(\mathrm{N})}(C_i)$ affects the distribution of C_i, which is controlled by $P(C_i|\mathbf{Pa}_{ik}^{(\mathrm{N})}(C_i))$. Notice that when using the CI model shown in Fig. 3b, (4) can be further simplified as

$$P(C_i|\mathbf{Pa}^{(\mathrm{N})}(C_i)) = \sum_{o_{ik}\in\mathbf{Pa}(o_i)} \sigma_{i,ik} P(C_i|C_{ik}). \tag{5}$$

Let the link graph $\mathcal{G}_{\mathrm{D}} = (\mathcal{O}_{\mathrm{D}}, \mathcal{L}_{\mathrm{D}})$ be an instantiation of a relational schema $\mathcal{S}$, thus we can formally define a CDN model for the relational schema $\mathcal{S}$ as follows:

Definition 1. *For the relational schema $\mathcal{S}$, a CDN model $\mathcal{M} = (\mathcal{G}_{\mathrm{M}}, \mathbf{P}, \boldsymbol{\Theta})$ defines:*

1. *a directed model graph $\mathcal{G}_{\mathrm{M}}$ in which each node corresponds to an attribute of objects with type $t \in \mathbf{T}$ and each edge represents the dependency among attributes,*
2. *a set of template CPDs $\mathbf{P} = \mathbf{P}^{(\mathrm{L})} \cup \mathbf{P}^{(\mathrm{N})}$, where $\mathbf{P}^{(\mathrm{L})}$ and $\mathbf{P}^{(\mathrm{N})}$ are the intrinsic and relational CPDs, respectively, and*



(a) (b)

Fig. 4. (a) The model graph and (b) the inference graph for the link graph shown in Fig. 1b. The decimal over each bidirected edge indicates the dependency function values between two linked objects. For example, the decimal over edge (O1, O2) means $(\sigma_{1,2}, \sigma_{2,1})$.

3. *a set of parameters* Θ *that are used to specify dependency functions among linked objects in any link graph that is defined by the schema* $\mathcal{S}$.

The structure of the model graph $\mathcal{G}_{\mathrm{M}}$ is defined by the components of CPDs. The nodes of $\mathcal{G}_{\mathrm{M}}$ correspond to the variables in $\mathbf{P}$, and are associated with an intrinsic CPD and a relational CPD. The edges of $\mathcal{G}_{\mathrm{M}}$ connect a node to each of the variables in its intrinsic and relational CPDs. The parameter set Θ will be further discussed in Section 5.

For a given link graph $\mathcal{G}_{\mathrm{D}}$, a CDN model uses the $\mathcal{G}_{\mathrm{M}}$ and $\mathcal{G}_{\mathrm{D}}$ to instantiate an inference graph $\mathcal{G}_{\mathrm{I}} = (\mathcal{V}_{\mathrm{I}}, \mathcal{E}_{\mathrm{I}})$ during inference. $\mathcal{G}_{\mathrm{I}}$ represents the probabilistic dependencies among all the object variables in a single test set [10]. The structure of $\mathcal{G}_{\mathrm{I}}$ is determined by $\mathcal{G}_{\mathrm{D}}$ and $\mathcal{G}_{\mathrm{M}}$. That is, the class-level dependencies in the CDN are instantiated according to the link structure of $\mathcal{G}_{\mathrm{D}}$, to define object-level dependencies. Let n_t denote the number of edges in $\mathcal{G}_{\mathrm{M}}$ that represent intrinsic dependencies for the class label C_t for object type $t \in \mathbf{T}$ (i.e., the number of variables in C_t's intrinsic CPD), then the total number of nodes in $\mathcal{G}_{\mathrm{I}}$ will be $|\mathcal{V}_{\mathrm{I}}| = \sum_{t \in \mathbf{T}} (n_t + 1)|\mathbf{I}(t)|$, where $\mathbf{I}(t)$ is the number of objects with type $t \in \mathbf{T}$ in $\mathcal{G}_{\mathrm{D}}$. Accordingly, the number of edges in $\mathcal{G}_{\mathrm{I}}$ that represent intrinsic dependencies is $|\mathcal{E}_{\mathrm{I}}^{(\mathrm{L})}| = \sum_{t \in \mathbf{T}} n_t |\mathbf{I}(t)|$. Let $n_{t1 \to t2}$ denote the number of edges in $\mathcal{G}_{\mathrm{M}}$ that represent relational dependencies among the class label C_{t1} and the attributes of object type $t2 \in \mathbf{T}$, $L_{t1 \to t2}$ denote the number of edges ($\leq l$ links away) in $\mathcal{G}_{\mathrm{M}}$ among objects with type $t1$ and objects with type $t2$, then the number of (bidirectional) edges in $\mathcal{G}_{\mathrm{I}}$ that represent relational dependencies is $|\mathcal{E}_{\mathrm{I}}^{(\mathrm{N})}| = \sum_{t1,t2 \in \mathbf{T}} n_{t1 \to t2} L_{t1 \to t2}$. Fig. 4a illustrates the model graph for the link graph as shown in Fig. 1b, and Fig. 4b illustrates the corresponding inference graph. It can be easily shown that $|\mathcal{V}_{\mathrm{I}}| = 18$, $|\mathcal{E}_{\mathrm{I}}^{(\mathrm{L})}| = 11$, and $|\mathcal{E}_{\mathrm{I}}^{(\mathrm{N})}| = 11$ in this example (Note that the node *Year* of the type *Paper* is not taken into account since there is no dependency between *Year* and *Topic*). Given a CDN model $\mathcal{M}$, the full joint distribution over the unknown label variables in $\mathcal{G}_{\mathrm{D}}$ can be approximately expressed as follows:

$$
\begin{aligned}
P(\mathcal{G}_{\mathrm{D}}|\mathcal{M}) &= \prod_{t \in \mathbf{T}} \prod_{o_i \in \mathbf{I}(t)} P(C_i|\mathbf{Pa}(C_i)) \\
&= \prod_{t \in \mathbf{T}} \prod_{o_i \in \mathbf{I}(t)} \\
&\quad \left[\alpha_i P(C_i|\mathbf{Pa}^{(\mathrm{L})}(C_i)) + (1-\alpha_i) \sum_{o_{ik} \in \mathbf{Pa}(o_i)} \sigma_{i,ik} P(C_i|\mathbf{Pa}_{ik}^{(\mathrm{N})}(C_i)) \right] \\
&= \prod_{t \in \mathbf{T}} \prod_{o_i \in \mathbf{I}(t)} \left[\sum_{o_{ik} \in \{o_i\} \cup \mathbf{Pa}(o_i)} \widetilde{\sigma}_{i,ik} P(C_i|\mathbf{Pa}_{ik}^{*}(C_i)) \right],
\end{aligned}
\tag{6}
$$

where

$$
\widetilde{\sigma}_{i,ik} = \begin{cases} \alpha_i, & o_{ik} = o_i, \\ (1-\alpha_i)\sigma_{i,ik}, & o_{ik} \in \mathbf{Pa}(o_i), \\ 0, & \text{otherwise.} \end{cases}
$$

and

$$
\mathbf{Pa}_{ik}^{*}(C_i) = \begin{cases} \mathbf{Pa}_{ik}^{(\mathrm{L})}(C_i), & o_{ik} = o_i, \\ \mathbf{Pa}_{ik}^{(\mathrm{N})}(C_i), & o_{ik} \in \mathbf{Pa}(o_i). \end{cases}
$$

More generally, the joint distribution over all attributes of all objects in $\mathcal{G}_{\mathrm{D}}$ can be expressed as:

$$
P(\mathcal{G}_{\mathrm{D}}|\mathcal{M}) = \prod_{t \in \mathbf{T}} \prod_{A \in \mathbf{A}^t} \prod_{o_i \in \mathbf{I}(t)} \left[\sum_{o_{ik} \in \{o_i\} \cup \mathbf{Pa}(o_i)} \widetilde{\sigma}_{i,ik} P\big(o_i.A|\mathbf{Pa}_{ik}^{*}(o_i.A)\big) \right]. \tag{7}
$$

As a form of DNs, CDNs first approximate the full joint distribution for an entire collection of related objects with a set of CPDs. Then, each CPD can be further modeled as a linear combination of an intrinsic CPD and a set of relational CPDs with the weights represented by dependency functions. That is, the CDN allows one to decompose a CPD into smaller pieces. This can be exploited in the CDN

model to provide savings in the presentation of a CPD, ease of knowledge acquisition and domain modeling, and computational savings in the inference process.

3.3 Discussions

The key motivation of CDNs is to introduce dependency functions to directly capture the strengths of the contextual dependencies among linked objects. We can also introduce another kind of contextual dependency to capture deeper probabilistic relationships in relational data, namely, context-specific independence (CSI). CSI refers to conditional independencies that hold only in specific contexts [23]. Notice that here a context on variables is a specific assignment of values to certain variables. It has been shown that CSI can also allow one to decompose a CPD into smaller pieces. For a generic node, however, this decomposition is not particularly useful [23]. That is, only when $\mathbf{X}$ exhibits a significant amount of CSI can this decomposition result in a far more compact representation. Thus, the current implementation of RDNs does not consider CSI.

4 Modeling Linkage Contextual Dependency

In this section, we describe how to quantify dependency functions among linked objects. Our basic idea is that the dependency function can be viewed as a quantitative measure of the contextual dependency among objects in the link structure. From the viewpoint of cognitive science [14], the semantics of an object are usually defined by its content and context. For example, bank has (at least) two different senses in English, as in the Commercial Bank of England (a financial institution) or the bank of the river Thames in England (a hydraulic engineering artifact). Here, the word river or commercial (rather than England) might be used as the contextual object of word bank. In the relational domain, a context for a given object denotes a collection of relevant objects in the link structure that make the semantics of that object unique and comprehensible. For a given object $o_i \in \mathcal{O}_{\mathrm{D}}$, let $\mathbf{N}(o_i)$ denote its context, then $\mathbf{N}(o_i) \subseteq \mathbf{Pa}(o_i)$. Usually, significant dependency exists among o_i and objects in $\mathbf{N}(o_i)$, but not among o_i and objects in $\mathbf{Pa}(o_i) - \mathbf{N}(o_i)$. Here, we refer to such contextual dependency among linked objects as *linkage contextual dependency*. Obviously, the linkage contextual dependency can be quantified from both the link structure and the semantic correlation among objects.

The link structure provides a wealth of information for revelation of contextual dependency among linked objects. The simplest way is to use link features. Let $\varphi_{i,j}$ be the link features between o_i and o_j. $\varphi_{i,j}$ may be binary, representing the presence/absence of a link among them; $\varphi_{i,j}$ may be set to $\varphi_{i,j} = w_{i,j} + w_{j,i}$, where $w_{i,j}$ indicates the number of links from o_i to o_j; $\varphi_{i,j}$ may also be set to be the frequency of linkage modes between o_i and o_j, where the linkage modes denote the important link relations that are likely to convey explicit semantic meaning, such as in-link, out-link, cocitation, and coreference. All the link features among objects in a graph $\mathcal{G}_{\mathrm{D}}$ allow us to construct a link feature matrix (denoted by $\mathbf{F} = [\mathbf{f}_1, \cdots, \mathbf{f}_N]$). Using this representation, we can easily calculate the similarity of two objects in $\mathcal{G}_{\mathrm{D}}$. For example, the similarity of objects o_i and o_j can be calculated by the dot product $\langle \mathbf{f}_i, \mathbf{f}_j \rangle$. More generally, we can use *linkage kernels* to capture the informative, high-order features of some complex, nonlinear relationships from the link structure [18]. The simplest linkage kernel matrix is the cocitation matrix $\mathbf{F}^{\mathrm{T}}\mathbf{F}$, which can be constructed directly by the dot product. Let $K(o_i, o_j)$ denote the linkage kernel among o_i and o_j. In the following, we will adopt several popular kernel functions to construct linkage kernels that can be used in our settings.

On the other hand, the linkage contextual dependency can also be measured by the semantic correlation among linked objects. It is well known that the autocorrelation is a common characteristic of many relational data sets, which measures a statistical dependency between the values of the same variable on linked objects [11]. For two linked objects, here we use mutual information to measure the semantic correlation among them. The higher the mutual information $I(o_i; o_j)$ between two objects o_i and o_j, the easier it is to estimate one object given the other, or vice versa. According to the definition [25], the calculation of $I(o_i; o_j)$ assumes that the class labels of related objects o_i and o_j are known. However, the prediction of the class labels of o_i and o_j is one of the main goals in calculating $I(o_i; o_j)$; thus, this creates a circular argument. A possible solution is to use a bootstrap step and the Gibbs sampling framework (See Section 5.2 for more details). Let $p(C_i|\mathcal{M})$ be the posterior probabilities of the class label of o_i given the model $\mathcal{M}$ that can be obtained by using Gibbs sampling. Assume that the types of o_i and o_j are t and t' where $t, t' \in \mathbf{T}$, then we use c_i^t and $c_j^{t'}$ to denote one possible value of C_i and C_j, respectively. Thus, $I(o_i; o_j)$ can be calculated as follows:

$$\begin{aligned} I(o_i; o_j) &= \mathbf{E}_{C_i, C_j}\left[\log \frac{P(C_i, C_j|\mathcal{M})}{P(C_i|\mathcal{M})P(C_j|\mathcal{M})}\right] \\ &= \frac{1}{\mathrm{Z}} \sum_{c_i^t} \sum_{c_j^{t'}} \frac{p(C_i = c_i^t|\mathcal{M}) p(C_j = c_j^{t'}|\mathcal{M})}{p(c_i^t)} \\ &\quad p(c_i^t|c_j^{t'}) \left[\log \frac{p(c_i^t|c_j^{t'})}{p(c_i^t)} - \log \mathrm{Z}\right], \end{aligned} \tag{8}$$

where $\mathbf{E}$ is mathematical expectation, Z is a normalization factor, $p(c_i^t|c_j^{t'})$ is the conditional probability of c_i^t given $c_j^{t'}$, $p(c_i^t)$, and $p(c_j^{t'})$ are their priors, and $P(c_i^t, c_j^{t'}|\mathcal{M})$ is the joint posterior probability,

$$P(c_i^t, c_j^{t'}|\mathcal{M}) \approx \frac{1}{\mathrm{Z}} \frac{p(c_i^t|\mathcal{M}) p(c_j^{t'}|\mathcal{M})}{p(c_i^t)} p(c_i^t|c_j^{t'}).$$

Notice that this equation not only contains the terms regarding the local information about o_i and o_j (i.e., $p(C_i = c_i^t|\mathcal{M})$ and $p(C_j = c_j^{t'}|\mathcal{M})$), but also the terms regarding the relational autocorrelation information about the entire link data set.

Therefore, we have the following definition of *dependency functions*:

Definition 2. *A function $\sigma(o_i, o_j)$ is called a* dependency function *of object $o_i \in \mathcal{O}_{\mathrm{D}}$ on object $o_j \in \mathcal{O}_{\mathrm{D}} (o_i \neq o_j)$ in the graph $\mathcal{G}_{\mathrm{D}}$ if it satisfies: 1) $\sigma(o_i, o_j) \geq 0$, 2) $\sum_{o_j \in \mathbf{Pa}(o_i)} \sigma(o_i, o_j) = 1$, and 3) the function $\sigma(o_i, o_j)$*

consists of at least two components: the mutual information $I(o_i; o_j)$ and the linkage kernel $K(o_i, o_j) = f(\varphi_{i,j})$. For $\forall o_j$,

$$o_k \in \mathbf{Pa}(o_i), o_k \neq o_j, I(o_i; o_j) \geq I(o_i; o_k),$$
$$K(o_i, o_j) = K(o_i, o_k) \Longrightarrow \sigma(o_i, o_j) \geq \sigma(o_i, o_k), \text{ and}$$
$$K(o_i, o_j) \geq K(o_i, o_k),$$
$$I(o_i; o_j) = I(o_i; o_k) \Longrightarrow \sigma(o_i, o_j) \geq \sigma(o_i, o_k).$$

For simplicity, we use $\sigma_{i,j}$ to denote $\sigma(o_i, o_j)$ in the following discussion.

Notice that here we are modeling only positive interactions between objects. If $o_j \notin \mathbf{Pa}(o_i)$, we set $\sigma_{i,j} = 0$. Moreover, we assume that the contextual dependency is a symmetric correlation between two objects, but the *strengths* of the bidirectional contextual dependencies are likely to be unequal and thus are measured by two different dependency functions.

There may be many forms of dependency functions that we can choose from different linkage kernel functions. Here, we present three special forms, as follows:

1. **Polynomial function model:**

$$\sigma_{i,j} = \frac{1}{\mathrm{Z}}\left(\langle \mathbf{f}_i, \mathbf{f}_j \rangle + 1\right)^{\beta_{\mathrm{I}}} I(o_i; o_j), \text{ where } \beta_{\mathrm{I}} = 1, 2, \text{ or } 3. \tag{9}$$

2. **Exponential function model:**

$$\sigma_{i,j} = \frac{1}{\mathrm{Z}} \exp\left(-\frac{|\mathbf{f}_i - \mathbf{f}_j|^2}{2\beta_{\mathrm{II}}}\right) I(o_i; o_j). \tag{10}$$

3. **Sigmoid function model:**

$$\sigma_{i,j} = \frac{1}{\mathrm{Z}} \tanh\left(\langle \mathbf{f}_i, \mathbf{f}_j \rangle + \beta_{\mathrm{III}}\right) I(o_i; o_j), \tag{11}$$

In the three cases, Z is a normalization constant. We can define the *dependency vector* of o_i, i.e., $\boldsymbol{\sigma}_i = \left[\sigma_{i,1}, \cdots, \sigma_{i,N}\right]^{\mathrm{T}}$, and the *dependency matrix* $\Sigma = [\boldsymbol{\sigma}_1, \cdots, \boldsymbol{\sigma}_N]$ for the link graph $\mathcal{G}_{\mathrm{D}}$, where $N = |\mathcal{O}_{\mathrm{D}}|$ is the number of objects in $\mathcal{G}_{\mathrm{D}}$.

By using dependency functions, CDNs can differentiate the impacts of the related objects on the classification. Moreover, we may directly apply dependency functions to optimize the objects' relational neighborhoods (possibly with irrelevant neighboring objects). Intuitively, some neighbors with relatively lower dependency function values would more likely be "irrelevant" neighbors for the given object o_i, or at least have less impact on the classification of o_i. If we keep the elements of the dependency vector $\boldsymbol{\sigma}_i$ in the nonincreasing order (i.e., $\sigma_{i,j1} \geq \cdots \geq \sigma_{i,j_{M_i}} \geq \sigma_{i,j_{M_i+1}} = \cdots = \sigma_{i,j_N} = 0$), it is possible to choose a proper K_i such that the last $|\mathbf{Pa}(o_i)| - K_i$ dependency function values are much smaller than the first K_i values and the first K_i neighbors in $\mathbf{Pa}(o_i)$ that correspond to these K_i values dominate the influence on the classification of o_i. In practice, the most appropriate K_i value is chosen such that $\left(\sum_{k=1}^{K_i} \sigma_{i,k}\right) \geq \sigma_{\perp}$ and $\left(\sum_{k=1}^{K_i-1} \sigma_{i,k}\right) < \sigma_{\perp}$ for a given threshold $\sigma_{\perp}$(e.g., $\sigma_{\perp} = 0.90$). Notice that K_i may be different for different $o_i \in \mathcal{O}_{\mathrm{D}}$.

5 Learning the CDN Models

In general, there are two components to learning any graphical model: structure learning and parameter estimation. Much like DNs and RDNs, both the structure and parameters of CDN models are determined through learning a set of CPDs. Thus, we discuss how to estimate the parameters of CDNs in this section. The inference in CDNs is also described here, since it will be used in CDN learning.

5.1 Parameter Estimation

The CDN model has two sets of parameters: 1) a set of CPDs $\mathbf{P} = \mathbf{P}^{(\mathrm{L})} \cup \mathbf{P}^{(\mathrm{N})}$ where $\mathbf{P}^{(\mathrm{L})}$ and $\mathbf{P}^{(\mathrm{N})}$ are the intrinsic CPDs and relational CPDs, respectively, and 2) a set of parameters $\boldsymbol{\Theta}$ that are used to specify dependency functions among linked objects. According to the discussion in the previous section, the parameter set $\boldsymbol{\Theta}$ should include the self-reliant factor α_t, the parameter β_t of linkage kernels, the priors $\pi_t = \left\{p_i^t = P(c_i^t)\right\}$, and the transition probability

$$\left\{p(c_i^t | c_j^{t'}) \mid t' \in \mathbf{T}\right\}$$

for each type $t \in \mathbf{T}$. Notice that when using the CI model,

$$p(c_i^t | c_j^{t'}) = p\left(C_i = c_i^t | \mathbf{Pa}_j^{(\mathrm{N})}(C_i) = c_j^{t'}\right), \tag{12}$$

but when using the RCI model,

$$p(c_i^t | c_j^{t'}) = \sum_{\widetilde{\mathbf{Pa}}_j^{(\mathrm{N})}(C_i)} p\left(C_i = c_i^t | \widetilde{\mathbf{Pa}}_j^{(\mathrm{N})}(C_i), C_j = c_j^{t'}\right) * p\left(\widetilde{\mathbf{Pa}}_j^{(\mathrm{N})}(C_i) | C_j = c_j^{t'}\right), \tag{13}$$

where $o_j = \mathbf{Pa}_j^{(\mathrm{N})}(o_i)$, and $\widetilde{\mathbf{Pa}}_j^{(\mathrm{N})}(C_i) = \mathbf{Pa}_j^{(\mathrm{N})}(C_i) - \{C_j\}$. Here, $p\left(\widetilde{\mathbf{Pa}}_j^{(\mathrm{N})}(C_i) \mid C_j = c_j^{t'}\right)$ can be calculated by using the intrinsic CPDs. These two equations mean that the transition probabilities can be obtained by using the intrinsic and relational CPDs. Thus, the parameter-estimation task is to learn a set of model parameters $\left\{\mathbf{P}_t^{(\mathrm{L})}, \mathbf{P}_t^{(\mathrm{N})}, \alpha_t, \beta_t, \pi_t\right\}_{t \in \mathbf{T}}$ from a training set $\mathcal{G}'_{\mathrm{D}} = (\mathcal{O}'_{\mathrm{D}}, \mathcal{L}'_{\mathrm{D}})$. The learned parameters are then applied to a separate testing set $\mathcal{G}_{\mathrm{D}}$.

Similar to RDNs [10], the CDN learning algorithm is, in principle, based on pseudolikelihood techniques, which estimate a set of conditional distributions independently. This approach avoids the complexities of estimating a full joint distribution and can incorporate existing techniques for learning probability distributions of relational data [10]. Specifically, three different methods are used in this paper to learn the parameters of CDNs.

Given a training set $\mathcal{G}'_{\mathrm{D}}$, we can use standard statistical learning methods to learn the parameters of the intrinsic CPDs $\mathbf{P}_t^{(\mathrm{L})}$, and the priors π_t for each type $t \in \mathbf{T}$ by assuming that the objects in $\mathcal{O}'_{\mathrm{D}}$ are independent: 1) The prior p_i^t can be estimated simply by the relative frequencies of the objects with the class label c_i^t in $\mathcal{O}'_{\mathrm{D}}$. 2) The intrinsic

CPDs $\mathbf{P}_t^{(\mathrm{L})}$ can be estimated by any probabilistic classification or regression techniques (called *intrinsic models*) such as naïve Bayes (NB) (e.g., [4]), logistic regression (e.g., [6]), or probabilistic support vector machine (SVM) (e.g., [29]).

For the relational CPDs $\mathbf{P}^{(\mathrm{N})}$, however, we cannot directly use the standard statistical learning methods, since the labels of instances are correlated. Instead, we can employ RPTs, which can adjust for biases toward particular features due to autocorrelation in relational data [10]. However, this learning algorithm, if used for CDNs, does not take into account the influence of dependency functions on the learning of $\mathbf{P}^{(\mathrm{N})}$. As mentioned above, $\mathbf{P}^{(\mathrm{N})}$ is also a set of control parameters in the calculation of dependency functions. Here, we model the learning process of $\mathbf{P}^{(\mathrm{N})}$ as a dynamic interacting process of multiple homogenous Markov chains. Then, we can use the self-mapping transformation algorithm [15] to learn a set of relational CPDs. The learning process of $\mathbf{P}^{(\mathrm{N})}$ includes the following steps:

Given a training set $\mathcal{G}'_{\mathrm{D}}$,

1. Initialize $\mathbf{P}^{(\mathrm{N})}$ randomly, and calculate Σ using the initial values of $\mathbf{P}^{(\mathrm{N})}$.
2. Partition the graph $\mathcal{G}'_{\mathrm{D}}$ into N' subgraphs, each of which contains an object o_i and its parents $\mathbf{Pa}(o_i)$. For each subgraph $\mathcal{G}'_{\mathrm{D}_i} = (\mathcal{O}'_{\mathrm{D}_i}, \mathcal{L}'_{\mathrm{D}_i})$, the log-likelihood function can be expressed from (6) as

$$\begin{aligned} Q &= \log P(\mathcal{G}'_{\mathrm{D}_i} | \mathbf{P}^{(\mathrm{N})}) \\ &= \sum_{o_j \in \mathcal{O}'_{\mathrm{D}_i}} \log \left[\sum_{o_k \in \mathbf{Pa}(o_j)} \sigma_{j,k} P(C_j | \mathbf{Pa}_k^{(\mathrm{N})}(C_j)) \right] \\ &= \sum_{o_j \in \mathcal{O}'_{\mathrm{D}_i}} \log \langle \boldsymbol{\sigma}_j, \mathbf{p}_j \rangle, \end{aligned} \tag{14}$$

where $\mathbf{p}_j = \left[P(C_j \mid \mathbf{Pa}_k^{(\mathrm{N})}(C_j)) \right]^{\mathrm{T}}_{o_k \in \mathbf{Pa}(o_j)}$. Then, we can use the self-mapping transformation method [15] to estimate $\mathbf{P}^{(\mathrm{N})}$ in this subgraph $\mathcal{G}'_{\mathrm{D}_i}$. First, we form the Lagrangian function

$$\begin{aligned} L_Q &= Q + \sum_{c_s} v_s \left[\sum_{\mathbf{c}_t} p_{st} - 1 \right] \\ &= \sum_{o_j \in \mathcal{O}'_{\mathrm{D}_i}} \log \langle \boldsymbol{\sigma}_j, \mathbf{p}_j \rangle + \sum_{c_s} v_s \left[\sum_{\mathbf{c}_t} p_{st} - 1 \right], \end{aligned} \tag{15}$$

where $p_{st} = P(C_j = c_s \mid \mathbf{Pa}_k^{(\mathrm{N})}(C_j) = \mathbf{c}_t)$, and v_s is the lagrange undetermined multiplier. It can be shown that the log-likelihood function is locally maximized when [15]

$$p_{st} = \frac{p_{st} \partial Q / \partial p_{st}}{\sum_{\mathbf{c}_t} p_{st} \partial Q / \partial p_{st}}, \tag{16}$$

where

$$\frac{\partial Q}{\partial p_{st}} = \sum_{o_j \in \mathcal{O}'_{\mathrm{D}_i}} \frac{\sum_{o_k \in \mathbf{Pa}(o_j)} \delta_{(j,k)}^{(t,s)} \sigma_{j,k}}{\langle \boldsymbol{\sigma}_j, \mathbf{p}_j \rangle}$$

and

$$\delta_{(j,k)}^{(t,s)} = \begin{cases} 1, & \text{if } C_j = c_s,\ \mathbf{Pa}_k^{(\mathrm{N})}(C_j) = \mathbf{c}_t, \\ 0, & \text{otherwise.} \end{cases}$$

Repeat this process for all subgraphs

$$\mathcal{G}'_{\mathrm{D}_i}\ (i = 1, \cdots, N').$$

3. Recalculate Σ using the current values of $\mathbf{P}^{(\mathrm{N})}$.
4. Repeat Steps 2 and 3 until convergence.

Let $C_i^{(n)}$ be the label variable of o_i after the nth iteration, then the sequence $C_i^1, \cdots, C_i^{(n)}$ can be viewed as a homogenous Markov chain. Different Markov chains influence one another at each iteration step and form a coupled Markov model in which the interinfluencing relationship is characterized by Σ and $\mathbf{P}^{(\mathrm{N})}$. This also enlightens us that the CDN model can be naturally extended to model dynamic relational data in the similar way that DPRMs extend PRMs [26].

For the parameters α_t and β_t for $t \in \mathbf{T}$, we can set the appropriate values by the cross-validation methodology. Sometimes, we can also set the value of α_t empirically. For example, α_t is set to be $0.7 \sim 0.8$ for type=*paper* and $0.4 \sim 0.5$ for type=*author* in the citation data.

It should be noted that during learning, relational models consider a large number of features, thus simple and efficient learning techniques are advantageous, particularly for joint models [10]. The CDN learning algorithm is much like the DN learning algorithm, except we use a self-mapping transformation algorithm to learn a set of relational CPDs. In practice, this learning procedure has good performance.

5.2 Inference

For a given link graph $\mathcal{G}_{\mathrm{D}}$, a CDN model uses the $\mathcal{G}_{\mathrm{M}}$ and $\mathcal{G}_{\mathrm{D}}$ to instantiate an inference graph $\mathcal{G}_{\mathrm{I}}$ during inference. Notice that the rollout process of a CDN model includes two operations: 1) As in many other SRMs, each object-attribute pair gets a separate, local copy of the appropriate CPD (including an intrinsic CPD and a relational CPD). 2) Calculate the dependency functions for the linked object-pairs in $\mathcal{G}_{\mathrm{I}}$ using the parameter set $\boldsymbol{\Theta}$ of the CDN model. An example of the CDN inference graph is shown in Fig. 4b.

In general, the CDN inference graph can be fairly complex. Clearly, exact inference over this complex network is impractical, so we must resort to approximate inference. In addition, cyclic dependencies also necessitate the use of approximate inference. As in DNs and RDNs, we also use ordered Gibbs sampling for approximate inference over CDN inference graphs.

For classification tasks, the first issue is how to initialize the values of the unknown label variables. Here, a bootstrap step is used to assign an initial label for each unlabeled object using only the intrinsic models. That is, $p(C_i|\mathcal{M})$ can be initialized as $p(C_i|\mathbf{Pa}^{(\mathrm{L})}(C_i))$, i.e., the intrinsic probabilistic dependency given the other attributes of o_i. Given the initial labels for all objects, we can construct an initial CDN inference graph $\mathcal{G}_{\mathrm{I}}^{(0)}$ over the link graph $\mathcal{G}_{\mathrm{D}}$. Gibbs inference then proceeds iteratively, estimating the joint posterior distribution over the unknown label variables given the

TABLE 1
Summarization of the Details of the Data Sets Used for Classification Tasks

		Cora	WebKB
Base Subset	Object Num	4331	1041
	Link Num *	11873(11873)	1120(1253)
	Link Density	ρ=2.74 Γ=1.3×10^{-3} Γ_ω=1.3×10^{-3}	ρ=1.08 Γ=2.1×10^{-3} Γ_ω=2.3×10^{-3}
	Category Distribution	*Case*(489), *GA*(626) *NN*(1388), *ProbM*(659) *ReinL*(354), *RuleL*(282) *Theory*(533)	*Course*(244) *Faculty*(153) *Project*(86) *Student*(558)
Extended Subset	Max Num of *other* category	1115	2788
	Max Num of *misc* links *	2639(2639)	3553(3792)
	δ_{MAX}	0.18	0.76

* The link number is denoted by $L(L_\omega)$, where L is the edge number of the link graph, and L_ω is the sum of edge weights.

data. For the variable of each unlabeled object, the *influence propagation* step is performed to return a refined posterior probability $p(C_i|\mathcal{M})$ given both other attributes of that object (i.e., $\mathbf{Pa}^{(\mathrm{L})}(C_i)$) and class labels (and attributes) of other related objects (i.e., $\mathbf{Pa}^{(\mathrm{N})}(C_i)$). Generally speaking, the influence propagation over a CDN inference graph $\mathcal{G}_\mathrm{I}$ is a process that probabilistic influence flows through active link paths in the network, allowing beliefs about one object to influence others to which it is related (directly or indirectly). This process is repeated for each unknown variable in the graph $\mathcal{G}_\mathrm{I}$. After a sufficient number of iterations, the values will be drawn from a stationary distribution [13].

Several criteria may be used to determine whether the iteration process of the Gibbs sampler will be terminated, e.g., the convergence of the log-likelihood over all unobserved label variables, the consistency of the *maximum a posterior* (MAP) estimates among two consecutive iterations and a predefined iteration upper bound. This paper adopts a mixed policy of these criteria.

6 Experiments

6.1 Data Sets

In this paper, we use two real-world data sets, each of which can be viewed as a link graph. Table 1 summarizes the details about them.

Cora [16]. The whole data set consists of about 37,000 papers, all of which have been categorized into hierarchical categories such as */Artificial_Intelligence/Machine_Learning/Case-Based/*. In common with many other works (e.g., [6]), we use the 4,331 papers in the seven subcategories of *Machine Learning*. In this collection, the papers are cited by or cite other papers through 11,873 citations. Here, we directly use Cora_0 to denote this *baseline* subset of 4,331 papers and 11,873 citations. In this case, however, we need to ignore about 2,639 citations that point from the Cora_0 set to papers with a wide variety of other topics. These citations may be potentially useful for classifying the papers in Cora_0, and on the other hand may also be useless or even produce a negative influence. In contrast to the 11,873 within-collection links, we refer to these citations as *miscellaneous links* (sometimes also directly as *noisy links*). To evaluate the robustness of the CDN models in the data sets with complex regularities, we also construct several extended data sets, denoted by Cora_δ, through adding into Cora_0 different amounts of such miscellaneous links that are randomly selected from the 2,639 citations. A parameter δ is used to control the ratio of these new citations in Cora_δ. We also use the category *other* to denote the label attributes of the papers that are pointed by these new citations.

WebKB [17]. The WebKB data set contains approximately 4,100 pages from four computer science departments, with a five-valued attribute representing their types (i.e., *faculty*, *student*, *project*, *course*, and *other*), and 10,400 links between pages. In this collection, the category *other* is a grab-bag of pages of many different types and contains a large number of pages (about 74.5 percent). We could restrict attention to just the pages with the other four labels, but in a relational classification settings, the deleted Web pages might be useful in terms of their interactions with other pages [8]. Thus, we also refer to the *base* subset of pages with the four labels as WebKB_0, and then construct the extended WebKB_δ sets by appending some links that point to these *other* pages. Similarly, we also refer to these links as the miscellaneous links, and a parameter δ is used to control the ratio of these new links in WebKB_δ.

The two data sets may exhibit different link regularities. To further evaluate this assertion, here we use the *link density* and the *maximal miscellaneous link ratio* δ_{MAX} to quantitatively measure such link regularities. As in [28], the link density can be measured by the average number of links per object $\rho = \frac{L}{N}$ and the graph sparseness $\Gamma = \frac{L}{N \cdot T}$ (or $\Gamma_\omega = \frac{L_\omega}{N \cdot T}$). Here, $N = |\mathcal{O}_\mathrm{D}|$, $L = |\mathcal{L}_\mathrm{D}|$, $T = |\{o_j | o_i \rightarrow o_j \in \mathcal{L}_\mathrm{D}, \forall o_i, o_j \in \mathcal{O}_\mathrm{D}\}|$ is the number of cited objects, and L_ω is the sum of edge weights. In Cora_0, $L = L_\omega = 11,873$ whereas in WebKB_0, $L_\omega = 1,253$, $L = 1,120$. Using these measures, Cora_0 has a link density of $\Gamma = \Gamma_\omega = 1.3 \times 10^{-3}$ and $\rho = 2.74$, whereas those of WebKB_0 are $\Gamma = 2.1 \times 10^{-3}$, $\Gamma_\omega = 2.3 \times 10^{-3}$ and $\rho = 1.08$. Clearly, compared with WebKB_0, Cora_0 not only has more links per document, but also has a more uniform link distribution among different documents. On the other hand, it can be easily calculated that $\delta_{MAX} = 0.76$ for WebKB and $\delta_{MAX} = 0.18$ for Cora. This means that WebKB has more miscellaneous links despite having fewer links per document. Thus, in this paper, we simulate the data sets with different link regularities by adjusting different δ values for Cora and WebKB. For simplicity, we set δ to values in {0,0.05,0.10,0.15,0.18} for the two data sets.

6.2 Experiments and Results

6.2.1 Overview

Two sets of experiments were designed over the two real-world data sets. The objective of the first set of experiments was to validate our fundamental conjecture in this paper that noisy links have high influence on link-based classification.

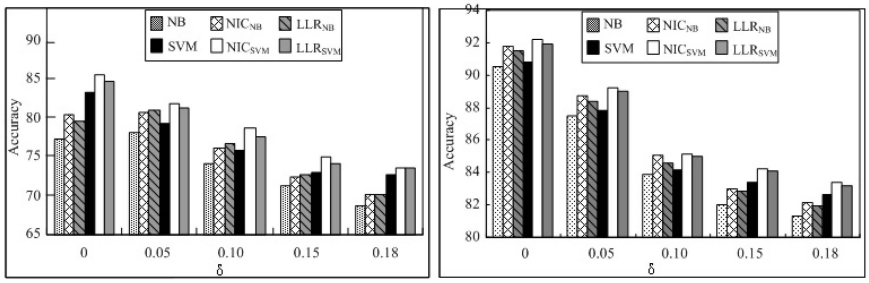


Fig. 5. Comparison of classification accuracies of all baseline models on (a) WebKB and (b) Cora. In our experiments, we use NBs and SVMs as the baseline intrinsic models, and use NICs and LLRs as baseline link-based models. We also reconstruct NICs and LLRs with NBs and SVMs as their intrinsic models, which are denoted by $\mathrm{NIC_{NB}}$, $\mathrm{NIC_{SVM}}$, $\mathrm{LLR_{NB}}$, and $\mathrm{LLR_{SVM}}$, respectively.

So, this set of experiments was performed only on the baseline models. We use NBs and SVMs as the baseline intrinsic models, and use NICs and LLRs as the baseline link-based models, where

- NICs classify documents based on the text and (possibly inferred) class labels of neighboring documents, using relaxation labeling paired with naïve Bayes relational classifiers [4], and
- LLRs use a logistic regression model for link-based classification, based on a vector of aggregations of attributes of neighboring nodes linked with different types of links (in, out, and co-links) [6].

Note that the two link-based models do not have the ability to automatically identify which links are most relevant to the classification tasks.

Our main goal is to demonstrate the robustness of our CDN model in link-based classification on noisy data sets. Thus, the second set of experiments was designed to compare the collective classification performance of CDNs with those of NICs, LLRs, and RDNs. We will experimentally demonstrate that our explicit inclusion of the strengths of linkage contextual dependencies overcomes many difficulties faced by RDNs, NICs and LLRs in dealing with noisy links.

To evaluate the performance of link-based models over different intrinsic models, we reconstruct these link-based models, respectively, with NBs and SVMs as their intrinsic models. For convenience, they are denoted by $\mathrm{NIC_{NB}}$, $\mathrm{NIC_{SVM}}$, $\mathrm{LLR_{NB}}$, $\mathrm{LLR_{SVM}}$, $\mathrm{RDN_{NB}}$, $\mathrm{RDN_{SVM}}$, $\mathrm{CDN_{NB}}$, and $\mathrm{CDN_{SVM}}$, respectively.

In all experiments, k-fold cross-validation methodology was used. That is, each data set was partitioned into k folds; for each of k experiments, we used $k-1$ folds for training and the remaining one for testing. In general, with a large number of folds, the bias of the true error rate estimator will be small but the computational time will be very large as well. In practice, a common choice for k-fold cross validation is $k=10$. In the relational domain, however, the choice of the number of folds and the corresponding data set split method depend on the size and linkage properties (e.g., link density) of the data sets. For WebKB, we used the standard split along different schools. Accordingly, 4-fold cross-validation tests were performed on each WebKB_δ data set. We partitioned each Cora_δ data set into 10 equally sized parts by time and performed 10-fold cross-validation tests. However, the average link density of test sets in 10-fold cross-validation is much less than that in 4-fold cross-validation, which might in turn influence the performance of link-based models. We will validate this conjecture in the last set of experiments.

6.2.2 Results on Baseline Models

To validate our conjecture that noisy links have great influence on the link-based classification, the first set of experiments compared the classification performance of all the baseline models, i.e., the intrinsic models NBs and SVMs, the link-based models NICs and LLRs. The results, as illustrated in Fig. 5, show that, with a few exceptions, the accuracies of these models decline gradually with increasing the parameter δ for the two data sets. This means that the classification performance of these models is heavily influenced by the appending *other* objects and miscellaneous links. By our assumption, these *other* objects and miscellaneous links are used to simulate the noisy information. Thus, it seems safe to conclude that, in most cases, such noisy information provide no significant predictive information for categorization.

We also compared the boosts in accuracy of the baseline link-based models over the corresponding intrinsic models (called *relative accuracies*). The results are shown in Fig. 6. From this figure, we can more easily see the effects of noisy links on link-based classification:

- In most cases, the miscellaneous links do not provide significantly predictive information, or even produce a negative influence on the categorization. In other words, the baseline link-based models do not demonstrate robustness in the data sets with a few noisy links.
- In some cases, the miscellaneous links may be exploited to improve the classification performance, but with such links the baseline link-based models cannot make sure which are helpful for classification and which are not. For example, we can find an improvement in the relative accuracy for $\mathrm{LLR_{SVM}}$ from $\mathrm{WebKB_0}$ to $\mathrm{WebKB_5}$, but their improvements decline gradually from $\mathrm{WebKB_5}$ to $\mathrm{WebKB_{18}}$.

Therefore, we need more robust models for noisy link data. That is, link-based models should have selectivity for

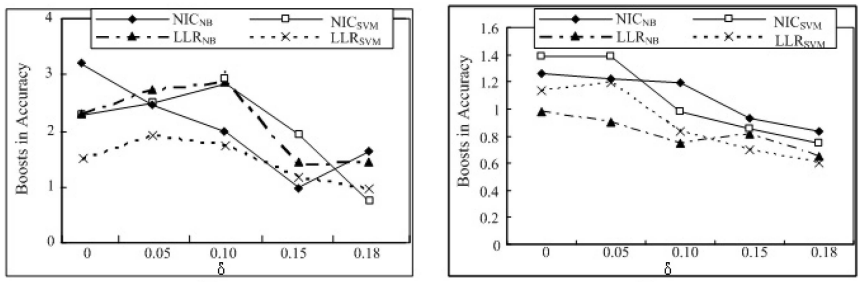


Fig. 6. Comparison of the relative accuracies of the baseline link-based models over the corresponding intrinsic models on (a) WebKB and (b) Cora. Here, the relative accuracy is computed by subtracting the accuracy of the corresponding intrinsic model from that of the link-based model.

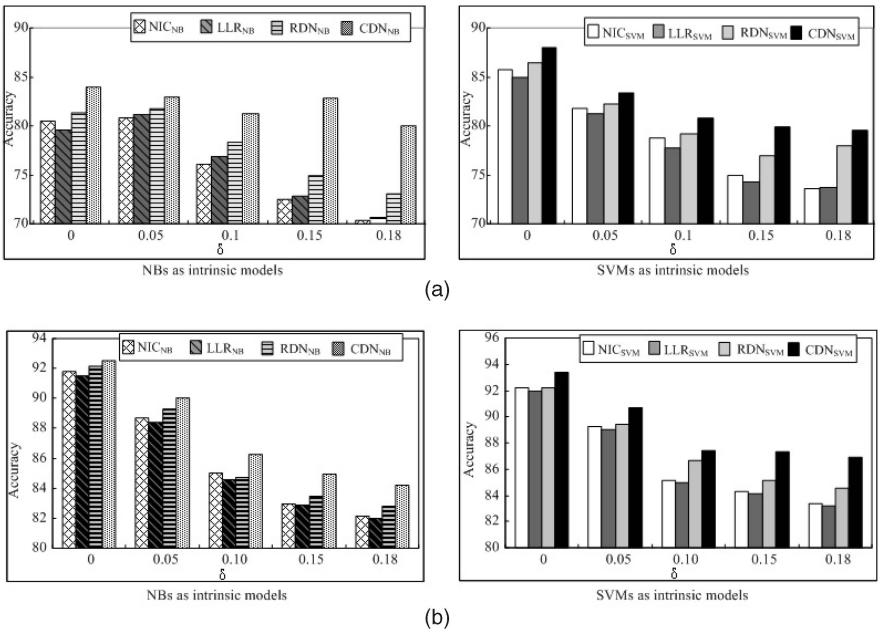


Fig. 7. Comparison of classification accuracies of all link-based models on (a) WebKB and (b) Cora.

different links in the data, since noisy links can be widely found in today's Web data and other link data sets.

Another interesting observation is that the relative accuracies of link-based models are more significant on WebKB than on Cora, despite the fact that citations in Cora may encode stronger semantic correlation. A possible reason is that the intrinsic models have already worked very well on Cora, which leaves little room for improvement.

6.2.3 Results on Collective Classification

The second set of experiments was to compare the classification performance of CDNs with NICs, LLRs, and RDNs in the collective classification scenario. In this scenario, we would have a whole collection of unlabeled instances that are linked together. Thus, the iterative classification algorithm (for NICs and LLRs) or Gibbs inference (for CDNs and RDNs) was performed for all unobserved objects in the test collection simultaneously. Fig. 7 shows the classification accuracies of all these link-based models on WebKB and Cora.

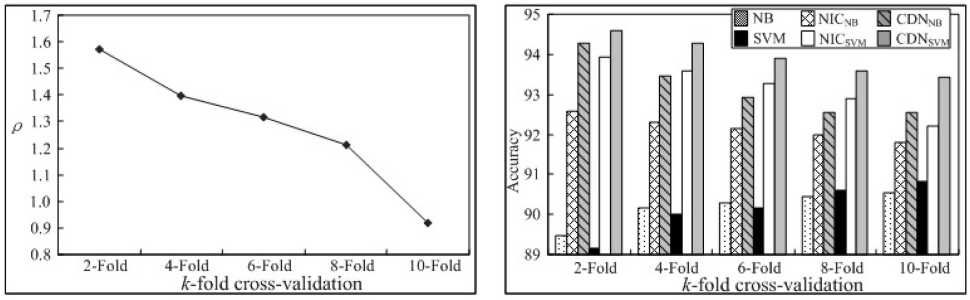
On average, $\mathrm{CDN_{NB}}$ outperformed $\mathrm{NIC_{NB}}$ and $\mathrm{LLR_{NB}}$, respectively, by about 6.15 percent and 6.08 percent on WebKB, and about 1.46 percent and 1.73 percent on Cora; $\mathrm{CDN_{SVM}}$ outperformed $\mathrm{NIC_{SVM}}$ and $\mathrm{LLR_{SVM}}$, respectively, by about 3.33 percent and 3.94 percent on WebKB, and about 2.32 percent and 2.51 percent on Cora. More importantly, the relative accuracies of CDNs do not decline along with increasing the parameter δ for the two data sets. In other words, CDNs can effectively exploit the miscellaneous links to improve the classification performance. We also noted one exception where the CDN models performed poorly on $\mathrm{WebKB_5}$. This indicates that the accuracy improvements of CDNs might be not significant when the data sets have only fewer noisy links. We will further validate this possibility using t-tests below.

On the other hand, $\mathrm{CDN_{NB}}$ outperformed $\mathrm{RDN_{NB}}$ on average by about 5.62 percent on WebKB and about 1.13 percent on Cora; $\mathrm{CDN_{SVM}}$ outperformed $\mathrm{RDN_{SVM}}$ averagely by about 2.71 percent on WebKB and about 1.57 percent on Cora. Obviously, although RDNs can use the selective relational classification algorithms (e.g., RPTs) to learn a set of CPDs, their performance is also affected by the noisy links in the inference phase. This also enlightens us that the selectivity of link features should be directly

TABLE 2
The p-Values of t-Test Results of NIC, LLR, RDN, and CDN Models

	WebKB					Cora				
Pairs	$WebKB_0$	$WebKB_5$	$WebKB_{10}$	$WebKB_{15}$	$WebKB_{18}$	$Cora_0$	$Cora_5$	$Cora_{10}$	$Cora_{15}$	$Cora_{18}$
(CDN_{NB}, NIC_{NB})	0.0341	**0.1165**	0.0026	0.0009	0.0057	0.0001	0.0036	0.0019	0.0000	0.0019
(CDN_{NB}, LLR_{NB})	0.0551	0.0962	0.0076	0.0018	0.0055	0.0085	0.0027	0.0001	0.0078	0.0054
(CDN_{SVM}, RDN_{SVM})	0.0064	**0.1084**	**0.1843**	**0.1276**	0.0625	**0.0882**	0.0013	0.0058	0.0091	0.0385
(CDN_{SVM}, NIC_{SVM})	0.0129	**0.1213**	**0.1193**	0.0057	0.0005	0.0001	0.0011	0.0001	0.0000	0.0000
(CDN_{SVM}, LLR_{SVM})	0.0606	0.0691	0.0928	0.0035	0.0299	0.0102	0.0201	0.0000	0.0018	0.0039
(CDN_{SVM}, RDN_{SVM})	0.0219	**0.2656**	**0.1984**	0.0017	0.0463	0.0094	0.0309	0.0017	**0.0696**	0.0091

* Results are computed using the results of the k-fold cross-validation tests on each sub-dataset (k=4 for WebKB and k=10 for Cora).



(a) (b)

Fig. 8. Experimental results of k-fold cross-validation tests on Cora_0, respectively, with $k = 2$, 4, 6, 8, 10: (a) Average link densities (e.g., ρ-values) of test sets. (b) Classification accuracies of NB, SVM, NIC, and CDN models.

encoded in the relational model itself such that the learned model can stay robust in different link data.

The differences in accuracy between the NIC, LLR, RDN, and CDN models may indicate that the improvements are not significant. To investigate this possibility, we performed two-tailed, paired t-tests to assess the significance of the results obtained from the four-validation tests. The null-hypothesis H_0 is that there is no difference in the accuracies of the two models, whereas the H_1 hypothesis is that there is a difference. The closer the resulting p-value is to zero, the more confident we can be that the null-hypothesis is not true. Table 2 shows the p-value results of $(\mathrm{CDN}, \mathrm{NIC})$, $(\mathrm{CDN}, \mathrm{LLR})$, and $(\mathrm{CDN}, \mathrm{LLR})$ pairs. We can see that, with a few exceptions, CDNs outperform NICs and LLRs at the 90 percent (averagely 97.7 percent) significance level on both WebKB and Cora, and outperform RDNs at the 80 percent (averagely 93.6 percent) significance level on the two data sets. The results support our conclusions that the classification performance of CDNs is significantly better than NICs, LLRs, and RDNs. We also notice that in some cases such as WebKB_5 and WebKB_{10}, the p-values of the $(\mathrm{CDN}, \mathrm{RDN})$ pair are slightly larger than those in the other cases. This means that there is still some room for improvement for CDN models when they are used in collective classification tasks.

We also performed a set of experiments to investigate the influence of the choice of the number of folds in cross-validation tests. As shown in Fig. 8a, the average link densities (e.g., ρ-values) of test sets decline gradually with increasing the number of folds. This means that with a larger number of folds, less link information can be utilized by link-based models for classifying objects in test sets. Fig. 8b shows the corresponding classification accuracies of NB, SVM, NIC, and CDN models. We can see that with increasing the number of folds, the intrinsic models (i.e., NB and SVM) performed slightly better but the classification performance of link-based models (i.e., NIC and CDN) was downing gradually.

In summary, the experimental results are generally positive, but in some cases, the improvements are not so significant. However, we can safely conclude from these results that the CDN models show relatively high robustness in the link data with a few noisy links.

We also examine the efficiency of Gibbs inference over the CDN models. Despite the fact that we used 200 as the iteration upper bound, most runs of Gibbs inference in our experiments converged within the first 100 iterations. To investigate the convergence rate, we tracked accuracy throughout the Gibbs inference procedure. Fig. 9 shows the accuracy curves throughout the Gibbs inference procedure on WebKB_{18} and Cora_{18}. Accuracy improves very quickly, leveling within the first 10 iterations. This shows that the Gibbs inference employed by the CDN model may be quite efficient to use in practice.

7 Conclusion

Many link data such as Web pages are often accompanied with a few noisy links. Such noisy links do not provide the predictive information for categorization. To capture such complex regularities in link data, this paper proposes a

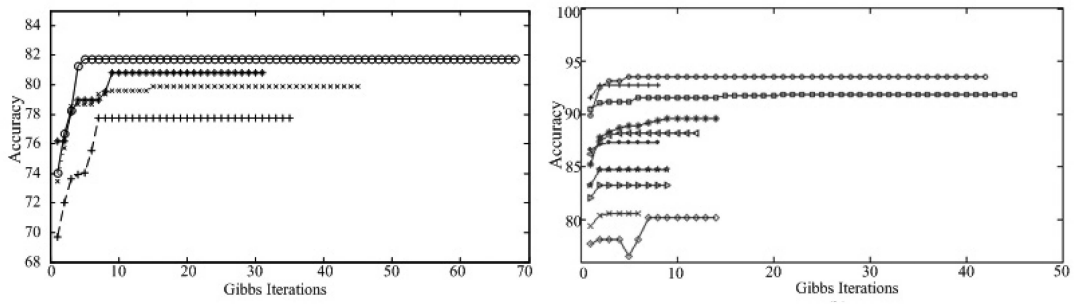


Fig. 9. Accuracy curves throughout the Gibbs inference procedure on (a) WebKB_{18} and (b) Cora_{18}. The curves in each subgraph represent a test run in cross-validation tests (results were computed on the average of three separate trials.)

contextual dependency network (CDN) model. We summarize the main contributions of this work as follows:

- The notion of context is explicitly introduced into the link domain. The appropriately shaped dependency functions can be used to weight contextual dependencies among objects in the link structure so that the learned relational models can reduce the effect of the noisy links on the classification.
- By extending the DN framework to model the linkage contextual dependencies, CDNs provide a robust model for link-based collective classification. The CDN model directly encodes the selectivity of link features, and also offers simple parameter estimation techniques. Moreover, the model can be easily extended to model dynamic relational data.
- On two real-world data sets, several experiments were designed to evaluate the effects of noisy links on different link-based classification models. Experimental results showed that the CDN model can demonstrate high robustness in these link data sets, and provide good prediction for the attributes of linked objects.

Currently, we are experimenting with Web image classification tasks to explore more interesting applications of the relational models. Our basic premise is that Web images which are cocontained in the same pages or contained in cocited pages are likely to be related to the same topic. We can build a robust image classification model by using visual, textual, and link information. We expect the CDN model to have higher accuracy over the SVM classifiers that only uses visual and textual features.

ACKNOWLEDGMENTS

The first, third and last authors are supported by the National Hi-Tech Research and Development Program (863) of China (grant No. 2003AA119010), and China-America Digital Academic Library (CADAL) project (grant No. CADAL2004002). Qiang Yang is supported by a grant from Hong Kong RGC (RGC Project HKUST 6187/04E). Charles X. Ling is supported by an NSERC grant in Canada. The authors would like to thank the anonymous reviewers who provided helpful comments on the earlier version of this paper.

REFERENCES

[1] L. Getoor, "Link Mining: A New Data Mining Challenge," *ACM SIGKDD Explorations Newsletter*, vol. 5, no. 1, pp. 84-89, 2003.
[2] Y. Yang, S. Slattery, and R. Ghani, "A Study of Approaches to Hypertext Categorization," *J. Intelligent Information System*, vol. 18, nos. 2/3, pp. 219-241, 2002.
[3] L. Getoor, E. Segal, B. Taskar, and D. Koller, "Probabilistic Models of Text and Link Structure for Hypertext Classification," *Proc. 17th Int'l Joint Conf. Artificial Intelligence Workshop Text Learning: Beyond Supervision*, pp. 24-29, 2001.
[4] S. Chakrabarti, B. Dom, and P. Indyk, "Enhanced Hypertext Categorization Using Hyperlinks," *Proc. ACM SIGMOD '98*, L.M. Haas and A. Tiwary, eds., pp. 307-318, 1998.
[5] J. Neville, D. Jensen, L. Friedland, and M. Hay, "Learning Relational Probability Trees," *Proc. Ninth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, pp. 625-630, 2003.
[6] Q. Lu and L. Getoor, "Link-Based Classification," *Proc. 12th Int'l Conf. Machine Learning*, pp. 496-503, 2003.
[7] N. Friedman, D. Koller, and B. Taskar, "Learning Probabilistic Models of Relational Structure," *J. Machine Learning Research*, pp. 679-707, 2002.
[8] B. Taskar, P. Abbeel, and D. Koller, "Discriminative Probabilistic Models for Relational Classification," *Proc. Uncertainty on Artificial Intelligence*, pp. 485-492, 2001.
[9] J. Neville and D. Jensen, "Collective Classification with Relational Dependency Networks," *Proc. Second Multi-Relational Data Mining Workshop KDD-2003*, pp. 77-91, 2003.
[10] J. Neville and D. Jensen, "Dependency Networks for Relational Data," *Proc. IEEE Int'l Conf. Data Mining*, pp. 170-177, 2004.
[11] D. Jensen and J. Neville, "Linkage and Autocorrelation Cause Feature Selection Bias in Relational Learning," *Proc. Ninth Int'l Conf. Machine Learning*, pp. 259-266, 2002.
[12] M. Richardson and P. Domingos, "Markov Logic Networks," *Machine Learning*, vol. 62, nos. 1-2, pp. 107-136, 2005.
[13] D. Heckerman, D. Chickering, C. Meek, R. Rounthwaite, and C. Kadie, "Dependency Networks for Inference, Collaborative Filtering, and Data Visualization," *J. Machine Learning Research*, vol. 1, pp. 49-75, 2001.
[14] P. Brézillon, "Context in Problem Solving: A Survey," *The Knowledge Eng. Rev.*, vol. 14, no. 1, pp. 1-34, 1999.
[15] S. Zhong, J. Ghosh, "A New Formulation of Coupled Hidden Markov Models," technical report, Dept. of Electrical and Computer Eng., Univ. of Texas at Austin, 2001.
[16] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the Construction of Internet Portals with Machine Learning," *Information Retrieval J.*, vol. 3, pp. 127-163, 2000.
[17] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitechell, K. Nigam, and S. Slattery, "Learning to Extract Symbolic Knowledge from the World Wide Web," *Proc. 15th Nat'l Conf. Artificial Intelligence*, pp. 509-516, 1998.
[18] Y.H. Tian, T.J. Huang, and W. Gao, "Latent Linkage Semantic Kernels for Collective Classification of Link Data," *J. Intelligent Information Systems*, 2006.
[19] D. Heckerman, C. Meek, and D. Koller, "Probabilistic Models for Relational Data," Technical Report, MSR-TR-2004-30, Microsoft Research, 2004.

[20] W. Uwents and H. Blockeel, "Classifying Relational Data with Neural Networks," *Proc. 15th Int'l Conf. Inductive Logic Programming*, S. Kramer and B. Pfahringer, eds., pp. 384-396, 2005.
[21] A. Popescul, L.H. Ungar, S. Lawrence, and D. M. Pennock, "Statistical Relational Learning for Document Mining," *Proc. IEEE Int'l Conf. Data Mining*, pp. 275-282, 2003.
[22] S.A. Macskassy and F. Provost, "NetKit-SRL: A Toolkit for Network Learning and Inference," *Proc. Ann. Conf. North Am. Assoc. Computational Social and Organizational Science (NAACSOS)*, 2005.
[23] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller, "Context-Specific Independence in Bayesian Networks," *Proc. 12th Conf. Uncertainty in Artificial Intelligence (UAI-96)*, E. Horvitz and F. Jensen, eds., pp. 115-123, 1996.
[24] M. Richardson and P. Domingos, "Mining Knowledge-Sharing Sites for Viral Marketing," *Proc. Eighth Int'l Conf. Knowledge Discovery and Data Mining*, pp. 61-70, 2002.
[25] R. M. Gray, *Entropy and Information Theory*. New York: Springer-Verlag, 1990.
[26] S. Sanghai, P. Domingos, and D. Weld, "Dynamic Probabilistic Relational Models," *Proc. 18th Int'l Joint Conf. Artificial Intelligence*, pp. 992-997, 2003.
[27] J. Neville, D. Jensen, and B. Gallagher, "Simple Estimators for Relational Bayesian Classifiers," *Proc. Third IEEE Int'l Conf. Data Mining*, pp. 609-612, 2003.
[28] M.J. Fisher and R.M. Everson, "When Are Links Useful? Experiments in Text Classification," *Proc. Advances in Information Retrieval, 25th European Conf. IR Research*, pp. 41-56, 2003.
[29] P. Sollich, "Probabilistic Methods for Support Vector Machines," *Proc. Advances in Neural Information Processing Systems*, vol. 12, pp. 349-355, 2000.
[30] D. Jensen, J. Neville, and B. Gallagher, "Why Collective Inference Improves Relational Classification," *Proc. 10th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, pp. 593-598, 2004.

Yonghong Tian received the MSc degree from the School of Computer Science, University of Electronic Science & Technology of China in 2000, and the PhD degree from the Institute of Computing Technology, Chinese Academy of Sciences in 2005. He is currently a faculty member at Digital Media Lab, Institute of Computing Technology, Chinese Academy of Sciences. His research interests include machine learning, data mining, semantic-based multimedia analysis, and retrieval. See http://www.jdl.ac.cn/user/yhtian/index.htm for more info. He is a member of the IEEE.

Qiang Yang received the PhD degree from the University of Maryland, College Park. He is a faculty member at the Hong Kong University of Science and Technology's Department of Computer Science. His research interests are AI planning, machine learning, case-based reasoning, and data mining. He is a senior member of the IEEE and an associate editor for the *IEEE Transactions on Knowledge and Data Engineering* and *IEEE Intelligent Systems*. See http://www.cs.ust.hk/~qyang.

Tiejun Huang received the BSc and MSc degrees from the Department of Automation, Wuhan University of Technology in 1992, and the PhD degree from the School of Information Technology & Engineering, Huazhong University of Science and Technology, China, in 1999. He was a postdoctorial researcher from 1999 to 2001 and a research faculty member at the Institute of Computing Technology, Chinese Academy of Sciences. He was also the associated director (from 2001 to 2003) and the director (from 2003 to 2006) of the Research Center for Digital Media in Graduate School at the Chinese Academy of Sciences. He is currently a faculty member in Institute of Digital Media, Peking University. His research interests include digital media technology, digital library, and digital rights management. He is a member of the IEEE.

Charles X. Ling received the MSc and PhD degrees from the Department of Computer and Information Science at the University of Pennsylvania in 1987 and 1989, respectively. Since then, he has been a faculty member of computer science at the University of Western Ontario, Canada. His main research areas include machine learning (theory, algorithms, and applications), cognitive modeling, and AI in general. He has published more than 100 research papers in journals (such as *Machine Learning*, *JMLR*, *JAIR*, *Bioinformatics*, the *IEEE Transactions on Knowledge and Data Engineering* (IEEE TKDE), and *Cognition*) and international conferences (such as IJCAI, AAAI, ICML, KDD, and ICDM). He has been an associate editor for the *IEEE TKDE*, and guest editor for several journals. He is also the director of the Data Mining Lab, which leads data mining development in CRM, Bioinformatics, and the Internet. He has managed several data mining projects for major banks and insurance companies in Canada. See http://www.csd.uwo.ca/faculty/cling for more info. He is a member of the IEEE.

Wen Gao (M'92-SM'05) received the BSc and MSc degrees in computer science from the Harbin University of Science and Technology and the Harbin Institute of Technology, China, in 1982 and 1985, respectively, and the PhD degree in electronics engineering from the University of Tokyo, Japan, in 1991. He was with the Harbin Institute of Technology from 1985, served as lecturer, professor, and head of the Department of Computer Science until 1995. He was with the Institute of Computing Technology, Chinese Academy of Sciences, from 1996 to 2005. During his professor career at the Chinese Academy of Sciences, he was also appointed as the director of the Institute of Computing Technology, executive vice president of Graduate School, as well as the vice president of the University of Science and Technology of China. He is currently a professor in the School of Electronics Engineering and Computer Science, Peking University, China. He is the editor-in-chief of the *Journal of Computer* (in Chinese), associate editor of the *IEEE Transactions on Circuit System for Video Technology*, and editor of the *Journal of Visual Communication and Image Representation*. He published four books and more than 300 technical articles in refereed journals and proceedings in the areas of multimedia, video compression, face recognition, sign language recognition and synthesis, image retrieval, multimodal interface, and bioinformatics. He earned the Chinese National Award for Science and Technology Achievement in 2000, 2002, 2003, and 2005, respectively. He is a senior member of the IEEE.