Leveraging Relational Autocorrelation with Latent Group Models

Jennifer Neville, David Jensen
Department of Computer Science, University of Massachusetts
Amherst, MA 01003
{jneville|jensen}@cs.umass.edu

Abstract

The presence of autocorrelation provides a strong motivation for using relational learning and inference techniques. Autocorrelation is a statistical dependence between the values of the same variable on related entities and is a nearly ubiquitous characteristic of relational data sets. Recent research has explored the use of collective inference *techniques to exploit this phenomenon. These techniques achieve significant performance gains by modeling observed correlations among class labels of related instances, but the models fail to capture a frequent cause of autocorrelation—the presence of underlying groups that influence the attributes on a set of entities. We propose a latent group model (LGM) for relational data, which discovers and exploits the hidden structures responsible for the observed autocorrelation among class labels. Modeling the latent group structure improves model performance, increases inference efficiency, and enhances our understanding of the datasets. We evaluate performance on three relational classification tasks and show that LGM outperforms models that ignore latent group structure, particularly when there is little information with which to seed inference.*

1. Introduction

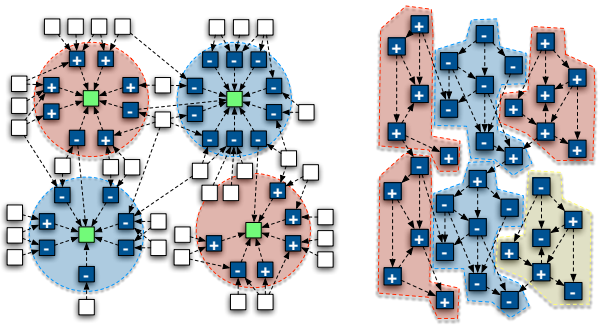
Autocorrelation is a statistical dependence between the values of the same variable on related entities, which is a nearly ubiquitous characteristic of relational datasets. For example, hyperlinked web pages are more likely to share the same topic than randomly selected pages [23], and movies made by the same studio are more likely to have similar box-office returns than randomly selected movies [6]. More formally, autocorrelation is defined with respect to a set of related instance pairs $P_R = \{(o_i, o_j) : o_i, o_j \in O\}$; it is the correlation between the values of a variable X on the instance pairs, (x_i, x_j) $s.t.$ $(o_i, o_j) \in P_R$.

The presence of autocorrelation offers a unique opportunity to improve model performance, as autocorrelation enables inferences about one object to be used to improve inferences about related objects. Indeed, recent work in relational domains has shown that *collective inference* over an entire dataset results in more accurate predictions than conditional inference over each instance independently (e.g., [2, 23, 15]) and that the gains over conditional models increase as autocorrelation increases [7].

Collective models improve classification performance by explicitly representing the autocorrelation dependencies in relational datasets. However, this approach has a number of weaknesses. First, the models do not attempt to discover the true cause of the autocorrelation, which may impair model interpretability and lead to poor domain understanding. Second, during learning the models restrict their attention to dependencies among instances that are closely linked in the data. Modeling the dependencies among more distant neighbors (e.g., [5]) may improve performance by allowing information to propagate in a more elaborate manner during inference. Finally, the models require inference over large, and often cyclic, graphical models, which necessitates the use of computationally complex, approximate inference techniques.

Current collective models, which model autocorrelation dependencies explicitly, fail to capture a frequent cause of autocorrelation—the presence of underlying groups, conditions, or events that influence the attributes on a set of entities. For example, in the cinematic domain, it is likely that studios cause the observed autocorrelation among movie returns. Movie-goers are unlikely to choose movies based on the returns of other movies from the same studio. It is more likely that movie returns are influenced by some unobserved properties of the studio (e.g., advertising budget). In this case, the class labels of movies may be conditionally independent given studio *type* (e.g., high-budget studio). Similarly, in the World Wide Web it is likely that web *communities*—groups of hyperlinked pages that share similar topics—cause the observed autocorrelation among page topics. In this case, we can not directly observe which community a web page belongs to, but it is likely that page topic is influenced by properties of the community (e.g., research groups contain *student, faculty*, and *project* pages), rather than the specific topics of hyperlinked pages. Models that

represent the underlying group structure and the correlation between group properties and member attributes should be able to express the joint distribution more accurately than approaches that only model the observed autocorrelation.



(a) Coordinating object groups. (b) Community groups.

Figure 1. Types of relational group structures.

Figure 1 depicts two potentials causes of observed autocorrelation. First, consider the task of predicting movie box office returns given data from the Internet Movie Database (IMDb). Figure 1a depicts an example fragment of the IMDb database, which contains studios, movies, and actors. There are four studios, each with a different number of movies, and each movie has a number of actors. Movie labels to indicate whether they made more than $2 million in opening-weekend box-office returns.

Returns are correlated among movies made by the same studio—high-grossing movies are clustered at the top-left and bottom-right of the graph. When such autocorrelation is present, the inferences about one movie can be used to improve the inferences about other movies made by the same studio. However, the properties of the studio may be the true cause of the observed autocorrelation and the returns of individual movies may be conditionally independent given studio identity and type. In this example, the studio object is a *coordinating object* that connects the group members (movies), and group type may indicate whether the studio is a high-budget studio or a small art-house studio.

Another example of underlying group structure is illustrated in figure 1b with an example fragment from the Web, where the objects are web pages and the links are hyperlinks among the pages. The pages are labeled according to a binary topic variable, which also exhibits autocorrelation. Again, the autocorrelation may be explained by the underlying group structure. In this case there are no coordinating objects that serve to connect the group members. Instead, the *community* groups can be identified by the pattern of linkage among the pages—the pages within a community are more likely to link to each other than pages in different communities. In this example, group type may indicate the primary interests of the community (e.g., alternative energy sites point to pages on solar and wind power).

In this paper, we propose a *latent group model* (LGM) to incorporate groups and their properties into a model of relational data. LGM is a joint model of links, attributes, and groups for unipartite relational graphs, which addresses the weaknesses of current collective models discussed above. LGMs recover the underlying groups and identify their associated density functions. This attempt to model the true *cause* of autocorrelation will improve domain understanding and motivate development of additional modeling techniques. Group models are also a natural way to allow more elaborate information propagation during inference without fully representing the $O(N^2)$ dependencies between all pairs of instances. Finally, in LGM models the objects are conditionally independent given the underlying group structure, so efficient exact inference techniques are applicable.

Our initial evaluation of LGMs is on *out-of-sample* classification tasks, where the test set is nearly disjoint from the training set. This is in contrast to recent work on *in-sample* classification [19] where the test set links back into the training set and it is possible to improve performance without generalizing about group properties. When the test set is nearly disjoint, it is necessary to both identify groups and generalize about their latent properties. More specifically, we will learn LGMs on training sets where the object attributes and links are observed but group structure and properties are unobserved. The learned models will then be used to classify instances in nearly disjoint test sets, where object class labels, group structure, and group properties are all unobserved. This approach is suited for domains with large, nearly disconnected graph structures and domains with dynamic graph structure, where groups emerge and disband over time. For example, in gene prediction tasks, models of proteins and how they interact to perform certain functions in the cell can be learned in one genome and then applied to classify proteins in new genomes. Also, in fraud detection tasks, which analyze a single dataset that is evolving over time, LGMs could be used to detect group formation and use a few hand-labeled examples to seed inference about the classifications of new group members.

In the remainder of the paper, we outline LGM, our initial algorithms for learning and inference, and related work in statistical relational learning. We present empirical evaluation on three classification tasks to demonstrate the capabilities of the model, showing that LGMs perform better than models that ignore latent groups, particularly when there is little known information with which to seed inference. Finally, we conclude with directions for future work.

2. Latent Group Models

Latent group models (LGMs) specify a generative probabilistic model of the attributes, links, and group structures in a relational dataset. The model posits that the data contains

different types of groups of objects. Membership in these groups influences the observed attributes of objects, as well as the existence of relationships (links) among objects.

For this initial investigation of LGMs, we make several simplifying assumptions about the data. Specifically, we assume a unipartite relational data graph (single object type) with binary, undirected links, and at most one link between any pair of objects. We also assume the number of objects, groups, and group types are fixed and known. However, it is relatively straightforward to extend the model to accommodate deviations from these assumptions.

The model assumes the following generative process for a dataset with N_O objects and N_G groups:

1. For each group g, $1 \leq g \leq N_G$:
   (a) Choose a value for group type t_g from $p(T)$, a multinomial probability with k values.
2. For each object i, $1 \leq i \leq N_O$:
   (a) Choose a group g_i uniformly from the range $[1, N_G]$.
   (b) Choose a class value c_i from $p(C|T_{G_i})$, a multinomial probability conditioned on the object's group type t_{g_i}.
   (c) For each attribute $A \in \mathbf{A_M}$:
      i. Choose a value for a_i from $p(A|C)$, conditioned on the object's class label c_i.
3. For each object j, $1 \leq j \leq N_O$:
   (a) For each object k, $j < k \leq N_O$:
      i. Choose an edge value e_{jk} from $p(E|G_j = G_k, T_{G_j}, T_{G_k})$, a Bernoulli probability conditioned on the two objects' group types and whether they are in the same group.

This generative model specifies that attribute values and link existence are conditionally independent given group membership and type. More specifically, the class labels of objects are conditionally independent. The joint distribution of a dataset D, with N_G groups, L links, and N_O objects, each with M attributes and class label C, is:

$$p(D) = \prod_{g \in N_G} p(t_g) \prod_{i \in N_O} p(c_i|t_{g_i}) \prod_{A \in \mathbf{A_M}} p(a_i|c_i) \cdot \prod_{l_{jk} \in L} p(e_{jk}{=}1|g_j{=}g_k, t_{g_j}, t_{g_k}) \prod_{l_{jk} \notin L} p(e_{jk}{=}0|g_j{=}g_k, t_{g_j}, t_{g_k})$$

Figure 2 depicts the the graphical model representation of an LGM model. The template consists of four plates, which represent replicates of groups, objects, attributes, and potential binary links among objects. Groups each have a type attribute T. Objects have a group membership G, a class label C, and attributes $A_1, ..., A_M$. E is a binary variable indicating link existence among all $\binom{N_O}{2}$ pairs of objects. The conditions on the arcs constrain the manner in which the model is *rolled out* for inference[1]—each E is influenced by two G and two T variables and each C is influenced by a single T variable in the unrolled Bayes net. The LGM model is a form of probabilistic relational model (PRM) [4] that combines a relational Bayesian network, link existence uncertainty, and hierarchical latent variables.

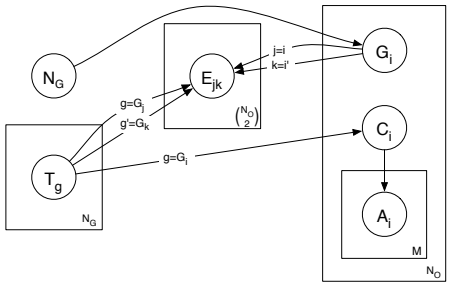


Figure 2. LGM graphical representation.

2.1. Learning

There are two steps to incorporating groups and their properties into a relational model. First, we need to detect the underlying group structure. When the groups are identified by a *coordinating object* (e.g., studio), detection is a relatively easy task—we can identify groups through high degree nodes and their immediate neighbors. When groups consist of *communities*, group detection is more difficult. However, if intra-community links are more frequent than inter-community links, the existing relations can be used as evidence of the underlying community structure, and membership can be inferred from the patterns of relations.

Second, we need to infer latent group types and model their influence on the attributes of group members. When the groups are observed, we can infer the latent types with a relatively simple application of the expectation-maximization (EM) algorithm. A similar approach is used in information retrieval where latent-unigram models represent each document as a *group* of word occurrences[2] that are conditionally independent given the document's *topic* [1]. When group membership is unobserved, the group boundaries and properties must be jointly inferred from the observed relations and attributes. To continue the document retrieval metaphor, it is as if we have word occurrence information (attribute values) and noisy indicators of word co-occurrence within documents (link information), but we do not know the document boundaries or the topic distributions.

When learning an LGM model, both group membership G and group type T are unobserved. Ideally, we could learn the model using a straightforward application of the EM algorithm—iterating between inferring the latent variables (E-step) and estimating the parameters (M-step). Un-

[1] We use *contingent Bayesian network* [14] notation to denote context-specific independence.

[2] However, the "vocabulary" is much smaller in relational domains—generally < 10 class values, compared to thousands of unique words.

fortunately, there are difficulties with this approach. First, there are N_O latent group variables, each with N_G possible values, and N_G latent type variables, each with k possible values. When the average group size is small (i.e., $N_G = O(N_O)$), we expect that EM will be very sensitive to the start state. Furthermore, when group membership is unknown, each of the C and E variables depends on all the T variables—there is no longer context-specific independence to exploit and the E-step will require inference over a large, complex, rolled-out Bayes net where objects' group memberships are all interdependent given the link observations. Exact inference in this situation is impractical. Although approximate inference techniques (e.g., loopy belief propagation) may allow accurate inference, given the number of latent variables and their dependency on sparse link information ($L \ll \binom{N_O}{2}$), we expect that EM will not converge to a reasonable solution due to many local (suboptimal) maxima in the likelihood function.

Our learning algorithm is motivated by the observation that collective models improve performance by propagating information only on existing links. This indicates that autocorrelated groups should have more intra-group links than inter-group links. We exploit this characteristic to decouple the group discovery from the remainder of the estimation process and propose the following approximate learning algorithm:

1. Hypothesize group membership for objects based on the observed link structure alone.
2. Use EM to infer group types and estimate the remaining parameters of the model.

A hard clustering approach in the 1^{st} step, which assigns each object to a single group, greatly simplifies the estimation problem in the 2^{nd} step—we only need to estimate the latent group type variables and parameters of $p(T)$, $p(C|T)$, and $p(A|C)$. To this end, we employ a recursive spectral decomposition algorithm with a norm-cut objective function [20] to cluster the objects into groups with high intra-group and low inter-group linkage.

Spectral clustering techniques partition data into disjoint clusters using the eigenstructure of a similarity matrix. We use the divisive, hierarchical clustering algorithm of [20] on the relational graph formed by the links in the data. The algorithm recursively partitions the graph as follows: Let $\mathbf{E}_{N \times N} = [E(i,j)]$ be the adjacency matrix and let $\mathbf{D}$ be an $N \times N$ diagonal matrix with $d_i = \sum_{j \in V} E(i,j)$. Solve the eigensystem $(\mathbf{D} - \mathbf{E})\mathbf{x} = \lambda \mathbf{D}\mathbf{x}$ for the eigenvector $\mathbf{x}_1$ associated with the 2^{nd} smallest eigenvalue λ_1. Consider m uniform values between the minimum and maximum value in $\mathbf{x}_1$. For each value m: bipartition the nodes into (A, B) such that $\forall v_a \in A \; x_{1a} < m$, and calculate the NCut value for the partition, $NCut(A,B) = \frac{\sum_{i \in A, j \in B} E(i,j)}{\sum_{i \in A} d_i} + \frac{\sum_{i \in A, j \in B} E(i,j)}{\sum_{j \in B} d_j}$. Partition the graph into the (A, B) with minimum *NCut*. If *stability*$(A, B) \leq c$, recursively repartition A and B.[3]

As we will show in section 4, the spectral clustering approach appears to work well in practice. However, refinements that iterate the clustering and EM steps, or incorporate soft clusterings, may improve results even further.

2.2. Inference

LGM models can be used to improve inference on *in-sample* and *out-of-sample* classification tasks. In-sample classification refers to tasks where the test set links back into the groups identified in the training set. In this case there may be enough information about the class distribution associated with the groups in the training set for group membership (i.e., G) to improve inference without inferring group type. Out-of-sample classification refers to tasks where the test set, or identified groups, are nearly disjoint from the training set groups. In this case, it is necessary to generalize about group properties to improve performance.

When applying an LGM model for classification, class labels C, group membership G, and group type T are all unobserved and must be inferred. Our inference algorithm is similar to the sequential learning procedure. We begin by clustering the objects into groups using the observed links. This simplifies inference by partitioning the objects into disjoint sets and fixing their group membership. Then inference can be decomposed into disjoint subtasks, one for each group. Within each group, the class labels are conditionally independent given the group type so we use standard belief propagation to jointly infer group types and class labels.

3. Related Work

3.1. Modeling Coordinating Objects

Slattery and Mitchell [21] use an approach based on the *HITS* algorithm [9] to identify coordinating objects in the data. This approach exploits autocorrelation dependencies by propagating information through these objects, but it does not generalize about group properties.

The social networks community provides latent variable models that cluster objects based on their link structure [18, 5, 25, 8]. These approaches use patterns of relations to identify which *roles* the objects play in the data. Although these models often incorporate richer representations for the relational link structure (e.g., [25]), they do not incorporate attributes into the model. The models are also primarily used for clustering rather than for classification.

PRMs have been used to cluster the objects in relational datasets [24], using latent variables on a subset of object

[3] We use the termination criterion proposed in [20]. *Stability*(A, B) is defined as the ratio of the minimum and maximum bin counts, after the values of $\mathbf{x}_1$ are binned by value into m bins. Unless otherwise noted, we use $m = \lceil log_2(N) + 1 \rceil$ and $c = 0.06$.

types. This approach does not posit group structures, but it is useful in situations where the coordinating objects are fixed and known. For example in the cinematic domain, an PRM with a latent variable on studios could be used to represent the autocorrelation among movie returns.

3.2. Modeling Communities

Popescul and Ungar [19] cluster relational database tables using standard k-means clustering algorithms and then use the cluster IDs as features in *conditional* models that reason about each instance independently. This approach has been shown to improve classification performance, but it can only be employed in situations where the test set instances link into the clusters used during training because the features use the identity of the clusters rather than generalizing over the properties of the groups.

Kubica et al. [10] use a latent variable model to cluster objects into groups based on their attribute values and link structure. Their approach is geared toward clustering data with multiple transactional links (e.g., phone calls, email) where the links patterns are homogeneous with respect to the groups. In other words, it is assumed that all groups have the same distribution of intra- and inter-group linkage. Situations where the patterns of linkage differ among groups are, however, easy to imagine. For example, consider machine learning papers: Reinforcement learning papers tend to cite papers in optimization, operations research, and theory, but genetic algorithm papers cite primarily other genetic algorithm papers. Allowing the link probabilities to vary among groups will be important for modeling group structures in large heterogeneous domains.

3.3. Collective Inference

Many collective models learn a joint distribution of the attributes of set of instances. Relational Markov networks (RMNs) [23] and relational dependency networks (RDNs) [15] model autocorrelation explicitly in the joint distribution. RMNs use clique templates to model the pairwise correlations among class labels of related instances; RDNs use aggregated features. These techniques model the autocorrelation dependencies at a global level—the autocorrelation dependencies are assumed to be uniform across each link in the data and parameters of features are tied across the entire dataset. As such, these models will not be able to distinguish among regions with varying levels of autocorrelation.

4. Experimental Results

The experiments in this section demonstrate the utility of latent group models in relational domains. Using three classification tasks, we evaluate whether LGMs can leverage autocorrelation to improve model accuracy and illustrate the conditions under which LGMs will perform well.

We present results for two LGM variations. The first variation, LGM-k, sets the number of group types to the number of class label values, $k = |C|$ (e.g., for binary tasks, $k = 2$); the second variation, LGM-2k, sets $k = 2|C|$. We compare LGM to four alternative models. The first two are conditional models that reason about each instance independently and do not use the class labels of related instances: The relational probability tree (RPT) model [16] is a decision tree model and the relational Bayesian classifier (RBC) model [17] is a naive Bayes model. The third model is a relational dependency network (RDN) [15] that reasons about networks of instances collectively. The fourth model (RDN-ceil) is a probabilistic ceiling for the RDN model, where we allow the true labels of related instances to be used during inference. This model shows the level of performance possible if the RDN model could infer the true labels of related instances with perfect accuracy.

To limit the confounding effects of feature construction and model selection, we first consider the restricted task of predicting class labels using only the class labels of related instances and/or the group membership. For the RPT and RBC models, we clustered the training and test sets together and used cluster ID as the sole attribute in the model. The performance of these baseline models illustrates the baseline utility of clustering without generalization about group type and serves as a comparison to previous work that clusters the data to generate additional features for classification [19]. For LGM, RDN and RDN-ceil, we used the class label of related instances as the sole attribute available for modeling. We used exact inference for all models except the RDN, which requires an approximate inference. In the RDN experiments, we used Gibbs sampling with chains of length 500 and burn-in of 100. (At this length, area under the ROC curve (AUC) had converged.) During inference we varied the number of known class labels available to seed the inference process. We expect this will be illustrative of performance when other information serves to seed the inference process—either when some labels can be inferred from intrinsic attributes, or when weak predictions about many related instances serve to constrain the system.

A second set of experiments, designed to explore the impact of intrinsic attribute information on performance, includes object attributes in each of the models.

4.1. Data and Tasks

The first data set was collected by the WebKB Project [3]. The data consist of a set of 3,877 web pages from four computer science departments, manually labeled with the categories: course, faculty, staff, student, research project, or other. We considered the unipartite co-citation web graph. The classification task was to predict page category. As in previous work on this dataset, we do not try to predict the category "other"; we remove these instances

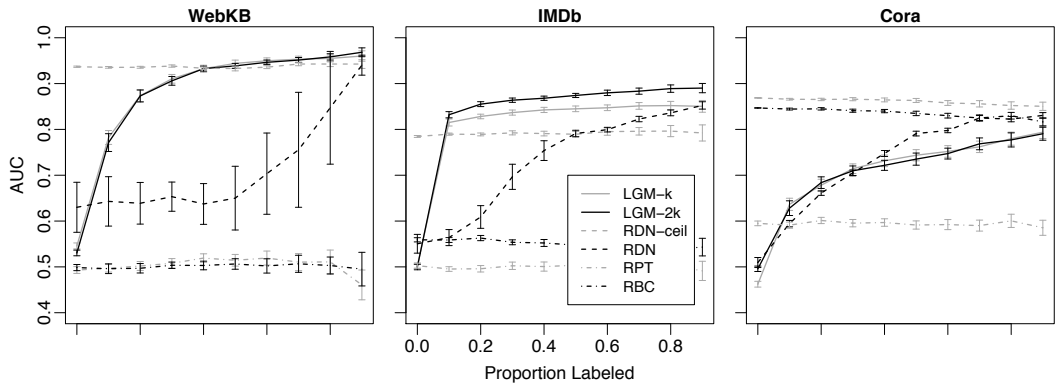


Figure 3. Model performance as the proportion of labeled instances during inference is varied.

from the data after creating the co-citation graph.

The second data set is from the IMDb (www.imdb.com). We used a sample of 1,382 movies released in the U.S. between 1996 and 2001. The binary classification task was to predict movie returns *>$2mil*. Based on past work that showed movie receipts to be autocorrelated through studios [6], we considered a unipartite graph of movies, where links indicate movies that are made by a common studio.

The third data set is drawn from Cora, a database of computer science research papers extracted automatically from the Web using machine learning techniques [13]. We considered the unipartite co-citation graph of 4,330 machine-learning papers. The classification task was to predict one of seven paper topics (e.g., *Neural Networks*).

4.2. Results

Figure 3 shows area under the ROC curve (AUC) results for each of the three classification tasks when the models do not use attributes. The graph shows AUC for the most prevalent class, averaged over all samples. For WebKB, we used cross-validation by department, learning on three departments and testing on the fourth. For IMDb, we used *snowball sampling* to bipartition the data into five training/test samples. For Cora, we used five temporal samples where we learned the model on one year and applied the model to the subsequent year. For each training/test split we ran 10 trials at each level of labeling (on the RDNs we only ran 5 trials due to relative inefficiency of inference). The error bars indicate the standard error of the AUC estimates for a single test set, averaged across the training/test splits. This illustrates the variability of performance within a particular sample, given the random initial labeling.

On WebKb and IMDB, LGM performance quickly reaches performance levels comparable to RDN-ceil, asymptoting at less than 40% known labels. (Note that RDNs and LGMs cannot be expected to do better than random at 0% labeled.) This indicates that the model is able to exploit group structure when there is enough information to accurately infer the group type. RDN performance doesn't converge as quickly to the ceiling. There are two explanations for this effect. First, when there are few constraints on the labeling space (e.g., fewer known labels), RDN inference may not be able to fully explore the space of labelings. Although we saw performance plateau for Gibbs chains of length 500-2000, it is possible that longer chains, or alternative inference techniques, could further improve RDN performance [12]. The second explanation is that joint models are disadvantaged by the data's sparse linkage. When there are few labeled instances, influence may not be able to propagate to distant objects over the existing links in the data. A group model that allow influences to propagate in more elaborate ways may be able to exploit the seed information more successfully. Future work will attempt to quantify the amount of error due to each of these sources.

RPT performance is near random on all three datasets. This is because the RPT algorithm uses feature selection and only a few cluster IDs show significant correlation with the class label. This indicates there is little evidence to support generalization about cluster identities themselves. The RBC does not do any feature selection and uses cluster IDs without regard to their support in the data. On Cora, the RBC significantly outperforms all other models. However, Cora is the one dataset where the test set instances link into the training set. This indicates that the RBC approach may be superior for in-sample classification tasks.

LGM performance does not reach that of RDN-ceil in Cora. Although the LGM outperforms the RDN when there is little know information, eventually the RDN takes over as it converges to RDN-ceil performance. We conjecture that this effect is due to the quality of the clusters recovered in Cora. Alternative clustering techniques (e.g., soft clustering, bi-clustering) may improve performance in these data.

Figure 4 shows average AUC results for each of the three classification tasks when we include attributes in the models. For the WebKB task, we included three page attributes: *school, url-server, url-text*; for IMDb, we included eight binary *movie-genre* attributes; for Cora, we included three pa-

per attributes: *type, year, month*. In all three datasets, the attributes improve LGM performance when there are few known labels. This is most pronounced in the IMDb, where LGM achieves ceiling performance with no class label information and indicates that movie genre is predictive of group type. In contrast, the RDN models are not able to exploit the attribute information as fully. In particular, in the WebKB task, the attributes significantly impair RDN performance. This is due to poor feature selection, which selects biased page attributes over the pairwise autocorrelation features. However, even on the other two tasks where this was not a problem, the RDN did not propagate the attribute information as effectively as the LGM when there were less than 40% known class labels.

5. Discussion

Consider the case where there are k group types, $|C|$ class values, and each object has a latent variable. There is a spectrum of group models ranging from $k=|C|$ to $k=N_G$. Collective models that model autocorrelation with global parameters, reason at one end of the spectrum ($k = |C|$) by implicitly using $|C|$ groups. Techniques that cluster the data for features to use in conditional models (e.g., [19]), reason at the other end of the spectrum ($k = N_G$) by using the identity of each cluster. The approach of [10] uses $k = 1$ in the sense that it ties the parameters of intra- and inter-group link probabilities across all groups.

When group size is large, there may be enough data to reason about each group independently (i.e., use $k = N_G$). For example, once a studio has made a sufficient number of movies, we can accurately reason about the likely returns of its next movie independently. However, when group size is small, modeling all groups with the same distribution (i.e., use $k = |C|$) will offset the limited data available for each group. A model that can vary k may be thought of as a backoff model, with the ability to smooth to the background signal when there is not enough data to accurately estimate a group's type in isolation. LGMs offer a principled framework within which to explore this spectrum.

One of the primary advantages of LGMs is that influence can propagate between pairs of objects that are not directly linked but are close in graph space (e.g., in the same group). In RMNs and RDNs, the features of an object specify its Markov blanket. This limits influence propagation because features are generally constructed over the attributes of objects one or at most two links away in the data. Influence can only propagate farther by influencing the probability estimates of attribute values on each object in a path sequentially. An obvious way to address this issue is to model the $O(N_O^2)$ dependencies among all pairs of objects in the data, but dataset size and sparse link information makes this approach infeasible for most datasets. PRMs with existence uncertainty [11] are the only current models that consider the full range of dependencies and their influence on observed attributes. Because LGMs can aggregate influence over an extended local neighborhood, they are a natural way to expand current representations while limiting the number of dependencies to model.

6. Conclusions

This paper presents a latent group model that reasons jointly about attribute information and link structure to improve reasoning in relational domains. To date, work on statistical relational models has focused on models of attributes conditioned on the link structure (e.g., [23]), or on models of link structure conditioned on the attributes (e.g., [11]). Our initial investigation has shown that modeling the interaction among links and attributes will likely improve model generalization and interpretability.

Latent group models are a natural means to model the attribute and link structure simultaneously. The groups decouple the link and attribute structure, thereby offering a way to learn joint models tractably. Our analysis has shown that group models outperform collective models when there is little information to seed inference. This is likely because a smaller amount of information is needed to infer group type than is needed to propagate information throughout sparse relational graphs. This suggests *active inference* as an interesting new research direction—where techniques choose which instances to label based on estimated improvement to the collective predictions.

Latent group models extend the manner in which collective models exploit autocorrelation to improve model performance. One of the reasons collective inference approaches work is that the class labels are at the "right" level of abstraction—they *summarize* the attribute information that is relevant to related objects [7]. Group models also summarize the information but at higher level of abstraction (i.e., group membership and type). Positing the existence of groups decouples the search space into a set of biased abstractions and could be considered a form of predicate invention [22]. This allows the model to consider a wider range of dependencies to reduce bias while limiting potential increases in variance and promises to unleash the full power of statistical relational models. Indeed, the results we report for LGMs using only the class labels and the link information achieve nearly the same level of performance reported by relational models in the recent literature.

Acknowledgments

The authors acknowledge helpful comments from S. Macskassy, C. Perlich, F. Provost, and the participants of the Dagstuhl Seminar 05051.

This effort is supported by DARPA and NSF under contract numbers IIS0326249 and HR0011-04-1-0013. The

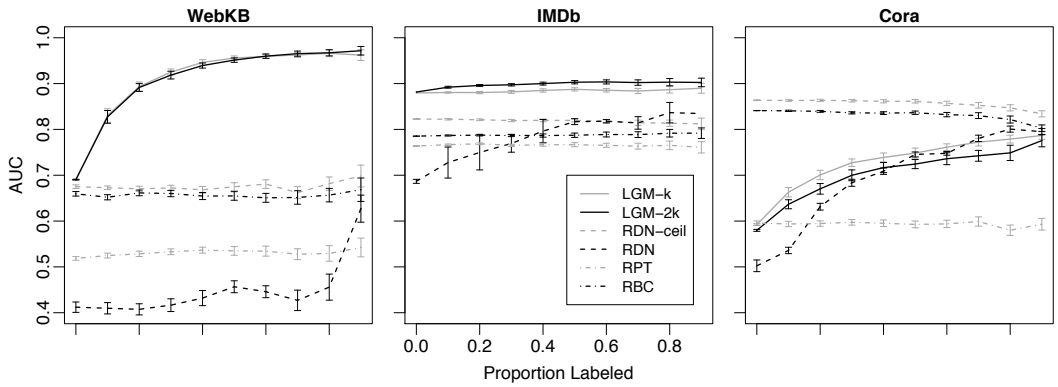


Figure 4. Model performance when attributes are included.

U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied of DARPA, NSF, or the U.S. Government.

References

[1] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[2] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *ACM SIGMOD 1998*, pages 307–318, 1998.

[3] 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 *AAAI-1998*, pages 509–516, 1998.

[4] L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Relational Data Mining*, pages 307–335. Springer-Verlag, 2001.

[5] P. Hoff, A. Raftery, and M. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97:1090–1098, 2002.

[6] D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In *ICML-2002*, pages 259–266, 2002.

[7] D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *ACM SIGKDD 2004*, pages 593–598, 2004.

[8] C. Kemp, T. Griffiths, and J. Tenenbaum. Discovering latent classes in relational data. Technical Report AI Memo 2004-019, Massachusetts Institute of Technology, 2004.

[9] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *the 9th ACM SIAM Symposium on Discrete Algorithms*, pages 25–27, 1998.

[10] J. Kubica, A. Moore, J. Schneider, and Y. Yang. Stochastic link and group detection. In *AAAI-2002*, pages 798–806, 2002.

[11] Q. Lu and L. Getoor. Link-based classification. In *ICML-2003*, pages 496–503, 2003.

[12] S. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. Technical Report CeDER-04-08, Stern School of Business, NYU, 2004.

[13] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. A machine learning approach to building domain-specific search engines. In *IJCAI-1999*, pages 662–667, 1999.

[14] B. Milch, B. Marthi, D. Sontag, S. Russell, D. Ong, and A. Kolobov. Approximate inference for infinite contingent bayesian networks. In *AISTATS-2005*, pages 238–245, 2005.

[15] J. Neville and D. Jensen. Dependency networks for relational data. In *ICDM-2004*, pages 170–177, 2004.

[16] J. Neville, D. Jensen, L. Friedland, and M. Hay. Learning relational probability trees. In *ACM SIGKDD 2003*, pages 625–630, 2003.

[17] J. Neville, D. Jensen, and B. Gallagher. Simple estimators for relational bayesian classifers. In *ICDM-2003*, pages 609–612, 2003.

[18] K. Nowicki and T. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96:1077–1087, 2001.

[19] A. Popescul and L. Ungar. Cluster-based concept invention for statistical relational learning. In *ACM SIGKDD 2004*, pages 665–670, 2004.

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

[21] S. Slattery and T. Mitchell. Discovering test set regularities in relational domains. In *ICML-2000*, pages 895–902, 2000.

[22] I. Stahl. Predicate invention in inductive logic programming. In *Advances in Inductive Logic Programming*, pages 34–47. 1996.

[23] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *UAI-2002*, pages 485–492, 2002.

[24] B. Taskar, E. Segal, and D. Koller. Probabilistic classification and clustering in relational data. In *IJCAI-2001*, pages 870–878, 2001.

[25] A. Wolfe and D. Jensen. Playing multiple roles: Discovering overlapping roles in social networks. In *the Workshop on Statistical Relational Learning, ICML-2004*, 2004.