In *Multi-Relational Data Mining Workshop at KDD-2003.*

Structural Logistic Regression for Link Analysis*

Alexandrin Popescul and Lyle H. Ungar

Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
{popescul,ungar}@cis.upenn.edu

Abstract. We present Structural Logistic Regression, an extension of logistic regression to modeling relational data. It is an integrated approach to building regression models from data stored in relational databases in which potential predictors, both boolean and real-valued, are generated by structured search in the space of queries to the database, and then tested with statistical information criteria for inclusion in a logistic regression. Using statistics and relational representation allows modeling in noisy domains with complex structure. Link prediction is a task of high interest with exactly such characteristics. Be it in the domain of scientific citations, social networks or hypertext, the underlying data are extremely noisy and the features useful for prediction are not readily available in a "flat" file format. We propose the application of Structural Logistic Regression to building link prediction models, and present experimental results for the task of predicting citations made in scientific literature using relational data taken from the CiteSeer search engine. This data includes the citation graph, authorship and publication venues of papers, as well as their word content.

1 Introduction

A growing number of machine learning applications involves the analysis of noisy data with complex relational structure. This dictates a natural choice in such domains: the use of relational rather than propositional representation and the use of statistical rather than deterministic modeling. Classical statistical learners provide powerful modeling but are generally limited to a "flat" file propositional domain representation where potential features are fixed-size attribute vectors. The manual process of preparing such attributes is often costly and not obvious when complex regularities are involved.

We present Structural Logistic Regression, an extension of logistic regression to modeling relational data that combines the strengths of classical statistical models with the higher expressivity of features automatically generated from a relational database. Structural Logistic Regression is an "upgrade" method in a sense used in inductive logic programming (ILP) to refer to the methods which extend existing propositional learners to handle relational representation. An "upgrade" implies that modeling and relational structure search are tightly coupled into a single process dynamically driven by the assumptions and model selection criteria of a propositional learner used. This contrasts with "propositionalization" which generally implies a decoupling of relational

* This paper partially overlaps with [29].

feature generation and modeling. Propositionalization has its disadvantages compared to upgrading, as it is difficult to decide a priori what features will be useful for a given propositional learner. Upgrading techniques let their learning algorithms select their own features with their own criteria. In large problems it is impossible to "exhaustively" propositionalize; feature generation should be driven dynamically at the time of learning. An extreme form of propositionalization is generating the full join of a database. This is both impractical and incorrect—the size of the resulting table is prohibitive, and the notion of an object corresponding to an observation is lost, being represented by multiple rows. Moreover, the entries in the full join table will be "atomic" attribute values, rather than values resulting from complex queries, what we desire for our features. The dynamic approach of coupled feature generation and modeling is more time and space efficient than its static alternative: it avoids making premature decisions about the depth of the feature space to explore; it generates only the features which will be looked at by the model selection, thus avoiding over-generation or under-generation; it avoids storing the entire table containing all feature candidates; it allows for flexible search strategies, and invites a number of statistical optimizations, for example sampling from subspaces of features before or instead of fully evaluating them.

Structural Logistic Regression integrates classical logistic regression with feature generation from relational data. We formulate the feature generation process as search in the space of relational database queries, based on the top-down search of refinement graphs commonly used in ILP [9], and extend it to include aggregate operators. Statistical information criteria are used dynamically during the search to determine which features are to be included into the model.

We propose the application of Structural Logistic Regression to link prediction and argue that the characteristics of the method and the task form a good match. Link analysis is an important problem arising in many domains. Web pages, computers, scientific publications, organizations, people and biological molecules are interconnected and interact in one way or another. Being able to predict the presence of links or connections between entities in a domain is both important and difficult to do well. We emphasize three key characteristics of such domains: i) their nature is inherently multi-relational, making the standard "flat" file domain representation inadequate, ii) such data is often very noisy or partially observed, and iii) the data are often extremely sparse. For example, in the domain of scientific publications, documents are cited based on many criteria, including their topic, conference or journal, and authorship, as well as the extremely sparse citation structure. In one example given below only one link exists in more than 7,000 potential link possibilities. All attributes contribute, some in fairly complex ways. The characteristics of the task suggest: i) using relational data model as a natural way to represent and store such data, ii) using statistical learning for building robust models from noisy data, and iii) using focused feature generation to produce complex, possibly deep, but local regularities, to be combined in a single discriminative model instead of trying to produce a full probabilistic model of the entire domain.

We present experimental results for citation prediction task in the domain of scientific publications. Link prediction models in this domain can be used to recommend citations to users who provide the abstract, names of the authors and possibly a partial reference list of a paper in progress. In addition to prediction, the learned features have

an explanatory power, providing insights into the nature of the citation graph structure. We use data from CiteSeer (a.k.a. ResearchIndex), an online digital library of computer science papers [22] (`http://citeseer.org/`). CiteSeer contains a rich set of relational data, including the text of titles, abstracts and documents, citation information, author names and affiliations, conference or journal names.[1]

2 Methodology

Our method couples two main components: generation of feature candidates from relational data and their selection with statistical model selection criteria (Figure 1). Relational feature generation is a search problem. It requires formulation of the search in the space of queries to a relational database. We structure the search space based on the formulation widely used in inductive logic programming for learning logic descriptions, and extend it to include other types of queries as the use of statistics relaxes the necessity of limiting the search space to only boolean values.

The language of non-recursive first-order logic formulas has a direct mapping into SQL and relational algebra (e.g. [5]), which can be used as well for the purposes of our discussion, e.g. as we do in [29]. Our implementation uses SQL for efficiency reasons providing connectivity with relational database engines.

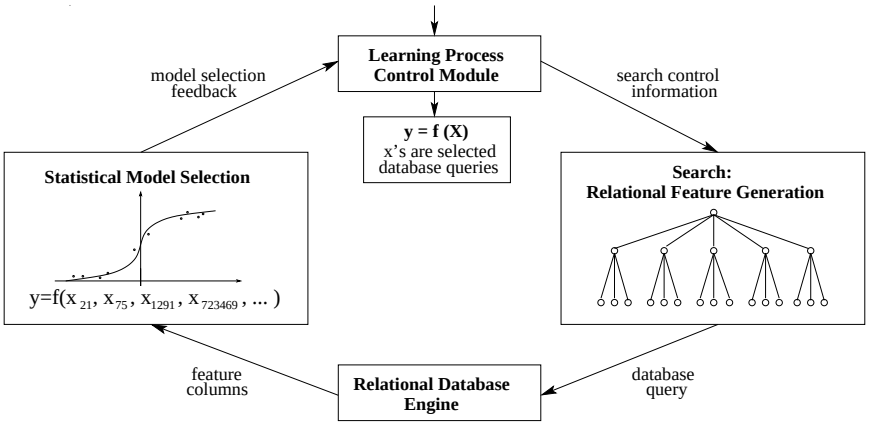

Fig. 1. The search in the space of database queries involving one or more relations produces feature candidates one at a time to be considered by the statistical model selection component. The process results in a statistical model where each selected feature is the evaluation of a database query encoding a predictive data pattern in a given domain

Logistic regression is used for classification problems. It is a discriminative model, i.e. it models conditional class probabilities without attempting to model the marginal

[1] Publication venues are extracted by matching information with the DBLP database: `http://dblp.uni-trier.de/`

distribution of features. Model parameters/regression coefficients are learned by maximizing conditional likelihood function. More complex models result in higher likelihood values, but at some point will likely overfit the data, resulting in poor generalization. A number of criteria aiming at striking the balance between likelihood optimization and model complexity have been proposed. Among the more widely used is the Bayesian Information Criterion (BIC) [33], which works by penalizing the likelihood by a term that depends on model complexity. We use sequential model selection to find a model which generalizes well by adding predictors resulting in improved BIC one at a time until the search space exploration is completed.

We introduce the notation first. Throughout this section we use the following relations: $cites(FromDoc, ToDoc)$, $author(Doc, Auth)$, $published\_in(Doc, Venue)$, and $word\_count(Doc, Word, Int)$. For compactness we use extended logic-based notation. First-order expressions are treated as database queries resulting in a table of all satisfying variable bindings, rather than a boolean value. The extended notation is necessary to introduce aggregations over entire query results or over their individual columns. Aggregate operators are subscripted with the corresponding variable name if applied to an individual column, or are used without subscripts if applied to the entire table. For example, an average count of the word "learning" in documents cited from a learning example document d, a potentially useful type of feature in document classification, is denoted as:

$$class(d) \quad \sim \quad ave_C \, [cites(d, \, D), \, word\_count(D, \, learning, \, C)],$$

where $\sim$ denotes "modeled using", i.e. the right hand side of the expression is one of the features in a statistical model of $class$. The duplicates in the column of satisfying bindings of C are not eliminated, unless an explicit projection of that column is performed before aggregation takes place. When necessary, we will borrow the projection operator notation (π) from the relational algebra.

The following is an example of a feature useful in link prediction; here the target concept is binary and the feature is a database query about both target documents $d1$ and $d2$:

$$cites(d1, \, d2) \quad \sim \quad count \, [cites(d1, \, D), \, cites(d2, \, D)]$$

is the number of common documents that both $d1$ and $d2$ cite.[2] Queries may be just about one of the documents in a target pair. For example:

$$cites(d1, \, d2) \quad \sim \quad count \, [cites(D, \, d2)]$$

is the number of times document $d2$ is cited. Larger values of this feature increase the prior probability of $d1$ citing $d2$, regardless of what $d1$ is.

2.1 Feature Generation

We define the search space based on the concept of "refinement graphs" [34] and expand it to include aggregate operators. Top-down search of refinement graphs is widely

[2] The database query is the right hand side of the expression.

used in inductive logic programming to search the space of first-order logic clauses. The search of refinement graphs starts with most general clauses and progresses by refining them into more specialized ones. Refinement graphs are directed acyclic graphs specifying the search space of the first-order logic queries. The space is constrained by specifying legal clauses (e.g. disallowing recursion and negation), and then structured by partial ordering of clauses, using a syntactic notion of generality (θ-subsumption [28]). Typically, a search node is expanded via a "refinement operator" to produce its most general specializations. Inductive logic programming systems using refinement graph search, usually apply two refinement operators: i) adding a predicate to the body of a clause, involving one or more variables already present, and possibly introducing one or more new variables, or ii) a single variable substitution (see e.g. [9]). We use a single refinement operator which combines the two: it adds (joins) one relation to a query expanding it into the nodes accounting for *all* possible configurations of equality conditions of new attributes with either a new or an old attribute[3], such that i) each refinement contains at least one equality condition with an old attribute, and ii) the types are taken into account to avoid adding equalities between attributes of different types. This refinement operator is complete. Not all refinements it produces are the most general refinements of a given query, however, we find that this definition simplifies pruning of equivalent subspaces; it has to account only for the type and the number of relations joined in a query.

In contrast to learning logic programs, we are not limited to searching in the space of boolean-valued expressions when building statistical models. At a high level we use refinement graphs to structure the search space. Each node of the graph is a query evaluating into a table of all satisfying variable binding. Within each node we perform a number of aggregations to produce both boolean and real-valued features. Thus, each node of the refinement graph can produce multiple feature candidates. Although there is no limit to the number of aggregate operators one may try, e.g. square root of the sum of column values, logarithm of their product etc., we find a few of them to be particularly useful. We use the following typical to SQL aggregate operators: $count$, ave, max, min and $empty$. Aggregations can be applied to a whole table or to individual columns, as appropriate given type restrictions, e.g. ave cannot be applied to a column of a categorical type. Adding aggregate operators results in a much richer search space. Binary logic-based features are also included through the aggregate operator $empty$. The situations when an aggregation is not defined, e.g. the average of an empty set, are resolved by introducing an interaction term with a 1/0 (not-defined/defined) feature.

The second aspect of defining search once the search space is structured is choosing search strategy. The present implementation performs the breadth-first search. In this setting, it is not necessary to specify the depth of the search prior to learning: intermediate models at any point of the search are usable. The decision to continue the exploration of deeper subspaces will depend on the available resources as well the expectation of how likely a model is to improve significantly if the search continues.

Search space potentially can be made arbitrarily complex. Richer queries, not necessarily involving only conjuncts and equality conditions, can also be made part of the

[3] In the experiments reported in this paper, we do not use conditions of equality with a constant.

search space. A key question for the future is how best to define such search spaces and how to control the search space complexity and search space bias.

The use of aggregate operators in feature generation makes pruning of the search space more involved. Currently, we use a hash function of partially evaluated feature columns to avoid fully recomputing equivalent features. In general, determining equivalence among relational expressions is known to be NP-complete. Polynomial algorithms exist for restricted classes of expressions, e.g. [1] (without aggregates) and [25] (with aggregates). However, deciding the equivalence of two arbitrary queries is different from avoiding duplicates when we have control over the way we structure the search space. The latter is simpler and should be the subject of future improvements.

Top-down search of refinement graphs allows a number of optimizations, e.g. i) the results of queries (prior to applying the aggregations) at a parent node can be reused at the children nodes; certainly, this needs to be balanced with the space requirements needed to store the views, and ii) a node which query results are empty for each observation should not be refined further as its refinements will also be empty.

3 Tasks and Data

Learning link prediction from relational data differs in several important aspects from other learning settings. Relational learning, in general, requires a quite different paradigm from "flat" file learning. The assumption that the examples are independent is violated in the presence of relational structure; this can be addressed explicitly [13], [14], or implicitly, as we do here, by generating more complex features which capture relational dependencies. When the right features are used, the observations are conditionally independent given the features, eliminating the independence violation.

In our link prediction setting, a class label of a learning example indicating the presence of a link between two documents is information of the same type as the rest of the link structure. Target links are not included in the background knowledge. This setting combines modeling-based and memory-based learning. We build a formal statistical model, but prediction of future data points requires database access, as each selected feature is a database query. Thus, an important aspect, more so than in attribute-value learning, is what information about new examples will be available at the time of prediction and how missing or changing background information would affect the results.

Consider the following two link prediction scenarios:

- The identity of all objects is known. Only *some* of the link structure is known. The goal is to predict unobserved links, from existing link structure alone or also using information about other available object attributes.
- New objects arrive and we want to predict their links to other existing objects. What do we know about new objects? Perhaps, we know some of their links, and want to predict the other. Alternatively, we might not know any of the links, but know some other attributes of the new objects.

In the latter case, when none of the new objects' links is known, and prediction is based solely on other attributes, e.g. only authorship and word content, feature generation would have to be controlled to not produce features based on immediate links, but use them when referring to the links in already existing background knowledge.

In this paper, we perform experiments for the first scenario. The data for our experiments was taken from CiteSeer [22]. CiteSeer catalogs scientific publications available in full-text on the web in PostScript and PDF formats. It extracts and matches citations to produce a browsable citation graph. The data we used contains 271,343 documents and 1,092,200 citations.[4] Additional information includes authorship and publication relations.[5] We use the following schema:

$cites(Document,\ Document),$
$author(Document,\ Person),$
$published\_in(Document,\ Venue).$

The training and test sets are formed by sampling citations (or absent citations for negative examples) from the citation graph. We perform learning on five datasets. Four of the datasets include links among all documents containing a certain query phrase, and the fifth data set covers the entire collection. Note that the background knowledge in the first four datasets also includes all other links in the full collection; only training and test links are sampled from the subgraph induced by document subsets. Table 1 contains the summary of the datasets.

The detailed learning setting is as follows:

- Populate three relations $cites$, $author$ and $published\_in$ initially with *all* data.
- Create training and test sets of 5,000 examples each by i) randomly sampling 2,500 citations for training and 2,500 citations for testing from those in column `# Links` of the Table 1; and ii) creating negative examples by sampling from the same subgraph also 2,500/2,500 train/test of "empty" citations, i.e. pairs of documents not citing each other.
- *Remove* test set citations from the $cites$ relation; but not the other information about the documents involved in the test set citations. For example, other citations of those documents are not removed.
- *Remove* training set citations from the $cites$ relation, so as not to include the actual answer in the background knowledge.
- Learning is performed i) using $cites$ relation only, or ii) using all three relations $cites$, $author$ and $published\_in$.

The positive and negative classes in this task are extremely unbalanced. We ignore the lack of balance at the training phase; at the testing phase we perform additional *precision-recall* curve analysis for larger negative class priors. The next section reports experimental results.

[4] This data is part of CiteSeer as of August 2002. Documents considered are only non-singleton documents out of the total of 387,703. Singletons are documents which both citation indegree and outdegree registered in CiteSeer are zero.

[5] The authorship information is known for 218,313 papers, and includes 58,342 authors. Publication venues are known for 60,646 documents. The set of venues consists of 1,560 conferences and journals.

Table 1. Number of documents, number of citations and citation graph density in each dataset. Density is the percentage of existing citations out of the total number of possibilities, (# Docs)^2

Dataset	# Docs	# Links	Density (10^{-2}%)
"artificial intelligence"	11,144	16,654	1.3
"data mining"	3,424	6,790	5.8
"information retrieval"	5,156	8,858	3.3
"machine learning"	6,009	11,531	3.2
entire collection	271,343	1,092,200	0.1

4 Results

We start by presenting the results for the balanced class priors test scenario, and continue with the analysis of the unbalanced class settings. Two sets of models are learned for each dataset: i) using only $cites$ relation, and ii) using all three relations $cites$, $author$ and $published\_in$.

When only $cites$ is used the average test set accuracy in five datasets is 88.73% and when all three relations are used the average increases to 90.90%.[6] In both cases the search explored features involving joins of up to three relations. It is not unreasonable to expect that even better models can be built if we allow the search to progress further. Table 2 details the performance in each dataset. The largest accuracy of 93.22% is achieved for the entire CiteSeer dataset. Even though this is the largest and the most sparse dataset, this is not surprising because, since the features are not domain specific and rely on the surrounding citation structure, this dataset retains more useful "supporting link structure" after some of them are removed to serve as training and testing examples (Section 3).

Table 2. Training and test set accuracy (%) of the models learned from the $cites$ alone, and from the $cites$, $author$ and $published\_in$. 5,000 train/test examples; balanced priors

Dataset	with $cites$		with all data	
	Train	Test	Train	Test
"artificial intelligence"	90.24	89.68	92.60	92.14
"data mining"	87.40	87.20	89.70	89.18
"information retrieval"	85.98	85.34	88.88	88.82
"machine learning"	89.40	89.14	91.42	91.14
entire collection	92.80	92.28	93.66	93.22

In the experiments using only the $cites$ relation the average number of features selected is 32; 13 of the selected features are the same across all five datasets. When

[6] The predicted probability of 0.5 was used as the decision cut-off in logistic regression.

all three relations $cites$, $author$ and $published\_in$ are used the average number of selected features is 40, with 14 features common to all five datasets. In addition to more obvious features, such as $d1$ is more likely to cite $d2$ if $d2$ is frequently cited, or if the same person co-authored both documents, or if $d1$ and $d2$ are co-cited, or cite the same papers[7], we learned some more interesting features. For example, a document is more likely to be cited if it is cited by frequently cited documents. Locally, this effectively learns the concept of an authoritative document [17], [26]. Or, the following feature, selected in all models:

$$cites(d1,\ d2) \ \sim \ count\ [\pi_{D2}\ (cites(D1,\ d2),\ cites(D1,\ D2))]$$

increases the probability of a citation if $d2$ is co-cited with many documents. Since this feature is selected in addition to the simple citation count feature, it could mean that either $d2$ appears more often in reviews, which tend to have longer lists of references, or it is cited from documents having smaller overlap among their references, which is more probable if they belong to different communities.

We compare the above results to the models trained on only binary features, i.e. when using only the $empty$ aggregate operator on the entire table. Such features are the logic-based features from the original formulation of refinement graph search. The binary features resulted in models with lower out-of-sample accuracies in all datasets. On average the accuracy with only binary features is 2.52 percentage points lower in models using $cites$ relation, and 2.20 percentage points lower in models using all three relations. The decrease of accuracy is significant at the 99% confidence level in both cases according to the t-test. We are currently unable to make experimental comparisons with a representative of classical ILP, FOIL; it runs out of memory on the system on which we have performed the rest of our experiments.

The class priors in our data are extremely unbalanced, due to the sparsity of the citation structure. The citation graph of the "artificial intelligence" dataset, for example is only 1.34×10^{-4} dense; that means that for one citation between two documents there are more than 7,000 non-existing citations, thus there are more than 7,000 times as many negative examples as there are positive. We perform the precision-recall curve analysis of our models trained with balanced class priors for testing situations with increased negative class proportions.

We vary the ratio k of the number of negative to the number positive examples used at testing. The ratio of one corresponds to the initial balance. We use for illustration the "artificial intelligence" dataset and the model trained using all three relations. New larger sets of negative examples are sampled with replacement from all "non-existing" links between documents in this dataset. Figure 2 presents precision-recall curves for k = 1, 10 and 100. As k increases the precision falls for the same levels of recall. Reducing the negative class prior should be performed when possible by filtering out obviously negative examples, for example by using a text based similarity, or other measures appropriate for a given task.

In application to citation recommendation, when only a few citations need to be recommended, we should care only about the high precision mode performance. In the case

[7] These two features correspond to the concepts of co-citation and bibliographic coupling used in bibliometrics.

of the precision recall curve for k = 100, for example, 10% of citations can be recalled for recommendation with 91% precision. This is an overall measure of performance—some users can receive more than enough recommendations, and others none. When we want to recommend a fixed number of citations to *every* user, the CROC performance metric should be used [32].

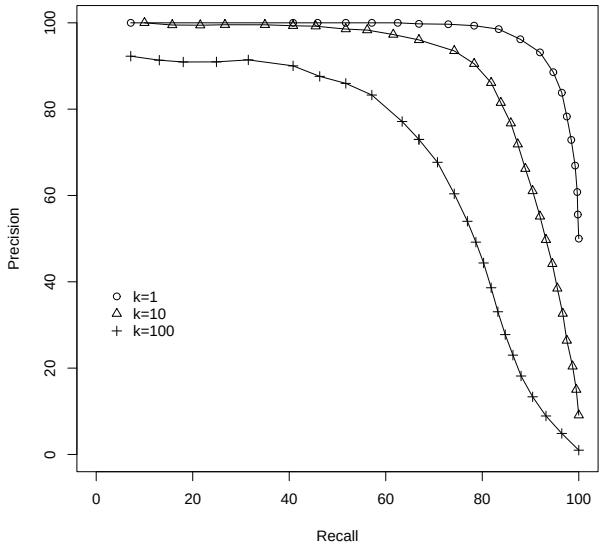

Fig. 2. Precision-recall curves for the "artificial intelligence" dataset with different class priors. k is the ratio of the number of negative to the number of positive examples used at testing

5 Related Work and Discussion

A number of approaches "upgrading" propositional learners to multi-relational domains have been proposed in the inductive logic programming community [21]. Often, these approaches upgrade learners most suitable to binary attributes. TILDE [4] and WARMR [8], for example, upgrade decision trees and association rules, respectively. S-CART [20] upgrades CART, a propositional algorithm for learning classification and regression trees. Upgrading implies that generation of relational features and their modeling are coupled into a single loop. Structural Logistic Regression presented in this paper can be categorized as an "upgrade".

Another approach is "propositionalization" (see e.g. [19]). In a broader sense "propositionalization" refers to any process of transforming a first-order representation into a propositional representation to make it suitable for a propositional learning algorithm.

In a narrower sense, as used more often in the literature, it is defined as "decoupling" of feature generation from modeling, where the features are first constructed from relational representation and then presented to a propositional algorithm. While in the former case we can view "upgrading" also as a form of "propositionalization", the latter emphasizes that the native propositional algorithm's model selection criteria do no take part in feature construction.

One form of "decoupled propositionalization" is to learn a logic theory with an ILP rule learner and then use the bodies of learned clauses as binary features in a propositional learner. For example, Srinivasan and King [35] use linear regression to model features constructed from the clauses returned by Progol. Decoupling feature construction from modeling, however, retains the inductive bias of the technique used to construct features, and better models can potentially be built if we allow a propositional learner itself to select its own features based on its own criteria. First Order Regression System (FORS) [15] more closely integrates feature construction into regression modeling, but does so using a FOIL-like covering approach for feature construction. Additive, or cumulative, models, such as linear or logistic regression, have different criteria for feature usefulness; fusing of feature construction and model selection into a single process is advocated in this context in [3] and [30].

Aggregate operators present an attractive way of extending feature spaces. For example, Knobbe et al. [18] use aggregates in propositionalization by first constructing a single table involving aggregate summaries of relational data and then using a standard propositional learner on this table. Perlich and Provost present a detailed discussion of aggregation in relational learning in [27].

An approach explicitly addressing numerical reasoning limitations of classical ILP is proposed in [36]. This approach augments the search space within P-Progol with clausal definitions of numerical, including statistical, functions which are lazily evaluated during the search. Our framework allows for the implementation of this extension to supplement the numerical reasoning capability we achieve via aggregate operators. The choice of numerical functions to be included into the search formulation on a par with the original relations should certainly be driven by the user insights into the application domain, as including too many augmenting functions makes the size of the search space prohibitive in large problems.

A number of learning models have been proposed which combine the expressivity of first-order logic with probabilistic semantics to address uncertainty. For example, "Stochastic Logic Programs" [23] model uncertainly from within the ILP framework by providing logic theories with a probability distribution; "Probabilistic Relational Models" (PRMs) [11] are a relational "upgrade" of Bayesian networks. Other examples include "Bayesian Logic Programs" [16], PRISM [31], "Relational Markov Networks" [37] and "Relational Markov Models" [2]. The marriage of richer representations and probability theory makes resulting formalisms extremely powerful, and inevitably a number of equivalences among them can be observed. In addition to a fundamental question of semantic and representational equivalence, it is useful to also consider the differences in how models are built, i.e. what objective function is optimized, what training algorithm is used to optimize that function, what is done to avoid over-fitting,

what simplifying assumptions are made. We believe that answering these questions in a single study will greatly enhance the discussion of these models.

A conflict exists between the two goals: i) probabilistically characterizing the whole domain at hand and ii) building a model that would address a specific question only, such as classification or regression modeling of a single response variable. This distinction typically leads to two philosophies in probabilistic/statistical machine learning: "generative" modeling and "discriminative" modeling. Generative models would attempt to model the distribution of their features, while discriminative models, like ours, do not do that, accepting them as they are and solely focusing on modeling the response variable distribution given these features, thus making it easier to learn by reducing the degrees of freedom that need to be estimated. For this reason, our method allows inclusion into the model of arbitrarily complex features without trying to do the impossible in large and sparse environments—estimating their distribution.

PRMs, for example, are generative models of joint probability distribution of entities and their attributes in a relational domain. Being a joint probability model of the entire domain, PRMs can provide answers to a large number of questions, including class labels, latent groupings, changing beliefs given new observations. An important limitation, however, of generative modeling is that in reality there is rarely enough data to reliably estimate the entire model. Generative modeling does not allow searching for complex features arbitrarily deep. One can achieve superior performance when focusing only on a particular question, e.g. class label prediction, and training models discriminatively to answer that question. Taskar et al. [37] propose "Relational Markov Networks" (RMNs)—a relational extension of discriminatively trained Markov networks. In RMNs, however, the structure of a learning domain, determining which direct interactions are explored, is prespecified by a relational template; this precludes the discovery of deeper and more complex regularities made possible by more focused feature construction advocated in this paper. Certainly, classification and regression do not exhaust potential applications. Generative modeling can prove useful in other problems, e.g. a formulation similar to PRMs, but semantically different, called a "Statistical Relational Model"—a statistical model of a particular database instantiation—was proposed for optimizing relational database queries [12].

Link analysis plays an important role in the hypertext domains, a notable example being Google, which uses the link structure of the Web by employing a link based concept of page authority in ranking search results [26]. In addition to knowing the authoritative documents, it is often useful to know the web pages which point to authorities on a topic, the so called called "hub" pages [17], which correspond to the concept of review papers in the scientific literature domain. A technique called "Statistical Predicate Invention" [7] was proposed for learning in hypertext domains, including learning certain types of relations between pages. It combines statistical and relational learning by using classifications produced by Naive Bayes as predicates in FOIL. Statistical Predicate Invention preserves FOIL as the central modeling component and calls Naive Bayes to supply new predicates. Neville and Jensen [24] propose an iterative technique based on a Bayesian classifier that uses high confidence inferences to improve class inferences for linked objects at later iterations. Cohn and Hofmann propose a joint probability model of document content and connectivity in [6].

6 Conclusions and Future Work

We presented Structural Logistic Regression, an "upgrade" of standard logistic regression to handle relational data representation. We demonstrate the advantages of using richer first-order representation with classical statistical modeling by applying the method to link prediction in the domain of scientific literature citations. The link prediction task is inherently relational, noisy and extremely sparse, thus suggesting relational representation and the use of discriminative statistical modeling of complex features obtained by a tightly coupled search in the space of database queries and selected with model's native information criteria. Focused search allows generation of complex features and avoids their manual "packaging" into a single table, a process that can be expensive and difficult. Discriminative modeling, such as in logistic regression, does not require modeling the distribution of individual features, an impossible task in sparse domains with many potential predictors; it focuses instead solely on modeling the response variable's distribution given these features. Our method extends beyond classical ILP because statistics allows generation of richer features, better control of search of the feature space, and more accurate modeling in the presence of noise. On the other hand, our method differs from relational probabilistic network models, such as PRMs and RMNs, because these network models, while being good at handling uncertainly, do not attempt to learn and model new complex relationships. Other regression models, such as linear regression for modeling continuous outcomes or Poisson regression for modeling count data can be used within our framework provided a common package is used—they all fall into the category of generalized linear models.

In the citation prediction task explored here, the learned models have explanatory as well as predictive power. Selected features provide insights into the nature of citations; some features "re-discovered" common concepts in link analysis and bibliometrics, such as bibliographic coupling, co-citation, authoritative and "hub" documents. Other linked environments, such as the Web, social networks, and biological interactions, e.g. protein interactions, we believe, can be explored with this methodology.

We are extending this work in three main directions: better feature selection, better search, and incorporation of relations derived from clustering into the search space.

Learning takes place with an exponential number of potential feature candidates, only relatively few of which are expected to be useful. Instead of avoiding large feature spaces because of the dangers of overfitting, we should rather learn to deal with them with more sophisticated model selection criteria. Feature selection methods recently derived by statisticians give promising results for handling this potentially infinite stream of features with only a finite set of observations.

Our formulation supports the introduction of sophisticated procedures for determining which subspaces of the query space to explore. Intelligent search techniques which combine feedback from the feature selection algorithms, other information such as sampling from feature subspaces to determine their promise and using database meta-information will help scale to truly large problems.

We advocate the use of clustering to extend the set of relations used in feature generation. Clusters improve modeling of sparse data, improve scalability, and produce richer representations [10]. New cluster relations can be derived using attributes in other relations. For example, one can cluster documents based on words, giving "topics", or

authors based on co-authorship, giving "communities". Once clusters are formed, they represent new relationships which can be added to the relational database schema, and then used interchangeably with the original relations.

Acknowledgments

We thank Steve Lawrence, David Pennock and the anonymous reviewers for their help.

References

1. A. V. Aho, Y. Sagiv, and J. D. Ullman. Equivalences among relational expressions. *SIAM Journal of Computing*, 8(2):218–246, 1979.
2. Corin Anderson, Pedro Domingos, and Dan Weld. Relational Markov models and their application to adaptive web navigation. In *Proc. of KDD-2002*, pages 143–152, 2002.
3. Hendrik Blockeel and Luc Dehaspe. Cumulativity as inductive bias. In P. Brazdil and A. Jorge, editors, *Workshops: Data Mining, Decision Support, Meta-Learning and ILP at PKDD-2000*, pages 61–70, 2000.
4. Hendrik Blockeel and Luc De Raedt. Top-down induction of logical decision trees. *Artificial Intelligence*, 101(1-2):285–297, 1998.
5. Stefano Ceri, Georg Gottlob, and Letizia Tanca. *Logic Programming and Databases.* Springer-Verlag, Berlin, 1990.
6. David Cohn and Thomas Hofmann. The missing link - A probabilistic model of document content and hypertext connectivity. In *NIPS*, volume 13. MIT Press, 2001.
7. M. Craven and S. Slattery. Relational learning with statistical predicate invention: Better models for hypertext. *Machine Learning*, 43(1/2):97–119, 2001.
8. L. Dehaspe and H. Toivonen. Discovery of frequent datalog patterns. *Data Mining and Knowledge Discovery*, 3(1):7–36, 1999.
9. Saso Dzeroski and Nada Lavrac. An introduction to inductive logic programming. In Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*, pages 48–73. Springer-Verlag, 2001.
10. Dean Foster and Lyle Ungar. A proposal for learning by ontological leaps. In *Proc. of Snowbird Learning Conference*, Snowbird, Utah, 2002.
11. L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*, pages 307–338. Springer-Verlag, 2001.
12. Lise Getoor, Daphne Koller, and Benjamin Taskar. Statistical models for relational data. In *Workshop on Multi-Relational Data Mining at KDD-2002*, pages 36–55, 2002.
13. P.D. Hoff. Random effects models for network data. In *Proc. of the National Academy of Sciences: Symposium on Social Network Analysis for National Security*, 2003.
14. D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In *Proc. of ICML*, pages 259–266. Morgan Kaufmann, 2002.
15. Aram Karalic and Ivan Bratko. First order regression. *Machine Learning*, 26:147–176, 1997.
16. K. Kersting and L. De Raedt. Towards combining inductive logic programming and Bayesian networks. In *Proc. of ILP-2001*, 2001.
17. Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
18. Arno J. Knobbe, Marc De Haas, and Arno Siebes. Propositionalisation and aggregates. In De Raedt and A. Siebes, editors, *PKDD, LNAI 2168*, pages 277–288. Springer-Verlag, 2001.

19. S. Kramer, N. Lavrac, and P. Flach. Propositionalization approaches to relational data mining. In Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*, pages 262–291. Springer-Verlag, 2001.
20. Stefan Kramer and Gerhard Widmer. Inducing classification and regression trees in first order logic. In Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*, pages 140–159. Springer-Verlag, 2001.
21. W. Van Laer and L. De Raedt. How to upgrade propositional learners to first order logic: A case study. In Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*, pages 235–261. Springer-Verlag, 2001.
22. Steve Lawrence, C. Lee Giles, and Kurt Bollacker. Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6):67–71, 1999.
23. S. Muggleton. Stochastic logic programs. In L. De Raedt, editor, *the 5th Int'l Workshop on Inductive Logic Programming*, 1995.
24. J. Neville and D. Jensen. Iterative classification in relational data. In *Workshop on Learning Statistical Models from Relational Data, AAAI*, pages 13–20. AAAI Press, 2000.
25. Werner Nutt, Yehoshua Sagiv, and Sara Shurin. Deciding equivalences among aggregate queries. In *Proc. of PODS-98*, pages 214–223, 1998.
26. Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the Web. Technical report, Computer Science Department, Stanford University, 1998.
27. Claudia Perlich and Foster Provost. Aggregation-based feature invention and relational concept classes. In *Proc. of KDD-2003*, 2003.
28. G. Plotkin. A note on inductive generalization. In B. Meltzer and D. Michie, editors, *Machine Intelligence 5*, pages 153–163. Edinburgh University Press, 1969.
29. Alexandrin Popescul and Lyle H. Ungar. Statistical relational learning for link prediction. In *Workshop on Learning Statistical Models from Relational Data at IJCAI-2003*, 2003.
30. Alexandrin Popescul, Lyle H. Ungar, Steve Lawrence, and David M. Pennock. Towards structural logistic regression: Combining relational and statistical learning. In *Workshop on Multi-Relational Data Mining at KDD-2002*, pages 130–141, 2002.
31. Taisuke Sato. A statistical learning method for logic programs with distribution semantics. In *Proceedings of the 12th Int'l Conference on Logic Programming (ICLP95)*, pages 715–729, 1995.
32. Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. Methods and metrics for cold-start recommendations. In *Proc. of the 25'th Conference on Research and Development in Information Retreival (SIGIR 2002)*, 2002.
33. Gideon Schwartz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1979.
34. E. Shapiro. *Algorithmic Program Debugging*. MIT Press, 1983.
35. A. Srinivasan and R. King. Feature construction with inductive logic programming: A study of quantitative predictions of biological activity aided by structural attributes. *Data Mining and Knowledge Discovery*, 3(1):37–57, 1999.
36. Ashwin Srinivasan and Rui C. Camacho. Numerical reasoning with an ILP system capable of lazy evaluation and customized search. *J. of Logic Programming*, 40(2-3):185–213, 1999.
37. Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proc. of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI-2002)*, 2002.