Cluster-based Concept Invention for Statistical Relational Learning

Alexandrin Popescul POPESCUL@CIS.UPENN.EDU
Lyle H. Ungar UNGAR@CIS.UPENN.EDU
Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104

Abstract

We use clustering to derive new relations which augment database schema used in automatic generation of predictive features in statistical relational learning. Clustering improves scalability through dimensionality reduction. More importantly, entities derived from clusters increase the expressivity of feature spaces by creating new first-class concepts which contribute to the creation of new features. For example, in CiteSeer, papers can be clustered based on words or citations giving "topics", and authors can be clustered based on documents they co-author giving "communities". Such cluster-derived concepts become part of more complex feature expressions. Out of the large number of generated features, those which improve predictive accuracy are kept in the model, as decided by statistical feature selection criteria. We present results demonstrating improved accuracy and scalability when predicting publication venues using CiteSeer data.

1. Introduction

Statistical relational learning and related methods search a space of database queries or logic expressions to find those which generate new predictive features. A given schema, describing background data, is used to structure a search over database queries. Each query generates a table, which in turn is aggregated to produce scalar feature candidates. The process produces a stream of features, from which statistically significant predictors are selected. The expressivity of the generated features is determined by the set of central relational entities participating in the search.

Considerably more powerful models can be built when the original schema is augmented with new relations which are derived via clustering (*cluster-relations*). Clustering can be used to create first-class relational concepts which are not derivable otherwise from the original relations. The addition of cluster-relations to the schema results in the creation of richer, more expressive, feature spaces, resulting in more accurate models than those built from the original relational concepts. Perhaps surprisingly, this approach can also lead to the more rapid discovery of predictive features. In addition to summarizing information (e.g. "Is this document on a given topic?"), cluster derived concepts participate in more complex relationships (e.g., "Does the database contain another document on the same topic and published in the same conference?"). The creation of these new high-level concepts allows more accurate and robust modeling from complex data sources not simply through information reduction, but, more importantly, through the increased expressivity of the language used to describe patterns in the data (0).

2. Methodology

We use a form of statistical relational learning which integrates regression with feature generation from relational data. In this paper we use logistic regression, giving a method we call Structural Logistic Regression (SLR). SLR combines the strengths of classical statistical modeling with the high expressivity of features automatically generated from a relational database.

Cluster-relations enter the formulation of the search space used to generate predictive features exactly as the original relations. The original database schema is used to decide which entities to cluster and what sources of attributes to use, for example documents clustered by words or by citations create alternative clusterings of the same objects. Once the schema is expanded by adding derived cluster relations to it, the underlying statistical relational learning methodology is repeated, i.e. database queries of the feature generation search space are evaluated, and the resulting tables per observation are aggregated to produce scalar feature columns, Figure 1. The new relations added

are treated exactly the same as the original relations.

Section 2.1 briefly presents SLR; the reader is referred to (0) for a more detailed description.

2.1. Structural Logistic Regression

SLR is an extension of logistic regression to modeling relational data. It combines the strengths of classical statistical models with the higher expressivity of features automatically generated from a relational database. SLR dynamically couples two main components: generation of feature candidates from relational data and their selection using statistical model selection criteria. Relational feature generation is a search problem. It requires formulation of the search in the space of, possibly complex, queries to a relational database. At each search node, feature candidates are constructed and considered for model inclusion. Thus, the process incrementally learns predictive data patterns, possibly encoding complex regularities in a domain. The process results in a statistical model where each selected feature is the evaluation of a database query encoding a predictive data pattern.

As mentioned above, relational feature generation is a search problem. We use top-down search of refinement graphs (**?**; **?**) as our main search space specification method. Each node in the refinement graph is a database query. The search starts with simpler queries about learning examples and progresses by refining its nodes, i.e. adding more relation instances and conditions to a parent query. Since we are building statistical models, rather than logic clauses as is the case in inductive logic programming where refinement graphs are used, we are not limited to searching in the space of binary logic-valued clauses. In our case, each node of the graph is a query evaluating into a table of all satisfying solutions. Within each node we apply a number of aggregate operators to produce both boolean and real-valued features. Thus, each node of the refinement graph produces 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 or logarithm of their product, we find *count*, *ave*, *max*, *min*, and *empty* to be particularly useful. 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.

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. (**?**; **?**). Equivalence determination based on the homomorphism theorem for *tableau* query formalism, essentially the class of conjunctive queries we look at before aggregation, is explained in detail in (0), page 115. However, deciding the equivalence of two arbitrary queries is different from the simpler problem we face of avoiding duplicates when we have control over the way we structure the search space.

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, ii) a node resulting in an empty table for each observation should not be refined any further as its refinements will also be empty.

3. Task and Data

We explore the task of classifying CiteSeer documents into their publication venues, conferences or journals. The target concept pair is `<Document, Venue>`. The value of the response variable is one if the pair's venue is a true publication venue of the corresponding document and, it is zero otherwise. The search space contains queries based on several relations about documents and publication venues, such as citation information, authorship and word content of the documents. Modeling of latent structure of entities in this domain, such as topics of documents or communities of authors, is capable of producing more accurate predictive models that the original relational representation. Clusters can be derived by clustering entities in the domain based on the variety of alternative sources of attributes.

Publication venues were extracted by matching information with the DBLP database, `http://dblp.uni-trier.de/`. Publication venues are known for 60,646 CiteSeer documents.We use these documents and information about them in the experiments described below. Table XXX shows the basic relations we use,

- `PublishedIn(`*doc*`:Document,` *vn*`:Venue)`. 60,646 (60,646, 1,560)
- `Author(`*doc*`:Document,` *auth*`:Person)`. 131,582 (53,660, 26,740)
- `Citation(`*from*`:Document,` *to*`:Document)`. There are 173,410(42,749,31,603)
- `HasWord(`*doc*`:Document,` *word*`:Word`. 6,894,712 (56,104, 1,000) (The vocabulary was limited to

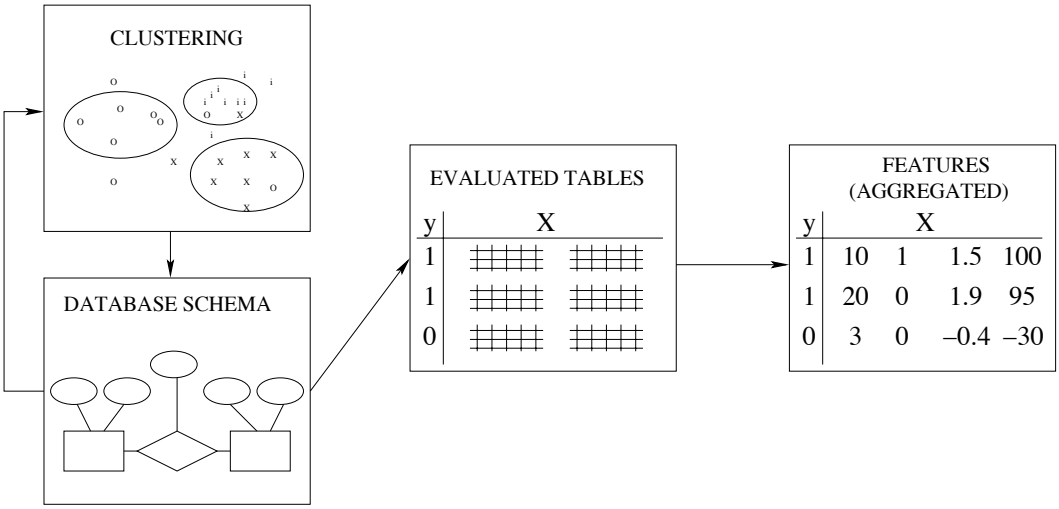

Figure 1. Cluster-relations augment database schema used to produce feature candidates.

the top 1,000 count words in the collection after Porter stemming and stop word removal).

We use k-means to derive cluster relations; any other hard clustering algorithm can be used for this purpose. The results of clustering are represented by binary relations
`<ClusteredEntity, ClusterID>`.

The original database schema contains several entities which can be clustered based on a number of alternative criteria. Each many-to-many relation in the original schema presented above can produce two cluster relations. Three out of four relations are many-to-many (with the exception of `PublishedIn`), this results in six new cluster-relations. The following is the list of these six cluster relations which we add to the relational database schema:[1]

- `Author(`*doc*`:Document,`*auth*`:Person)` produces:

  `ClusterDocsByAuthors(`*doc*`:Document,`*clust*`:Clust0)`
  53,660 documents are clustered based on the identity of their 26,740 authors.

  `ClusterAuthorsByDocs(`*auth*`:Person,`*clust*`:Clust1)`
  26,740 authors are clustered based on 53,660 documents they wrote.

- `Citation(`*from*`:Document,`*to*`:Document)` produces:

  `ClusterDocsByCitingDocs(`*doc*`:Document,`*clust*`:Clust2)`
  31,603 documents are clustered based on 42,749 documents citing them.

  `ClusterDocsByCitedDocs(`*doc*`:Document,`*clust*`:Clust3)`
  42,749 documents are clustered based on 31,603 documents cited from them.

- `HasWord(`*doc*`:Document,`*word*`:Word` produces:

  `ClusterDocsByWords(`*doc*`:Document,`*clust*`:Clust4)`
  56,104 documents are clustered based on the vocabulary of top 1,000 words they contain.

  `ClusterWordsByDocs(`*word*`:Word,`*clust*`:Clust5)`
  The vocabulary of 1,000 words is clustered based on their occurrence in this collection of 56,104 documents.

Throughout the experiments in this paper we use k-means clustering algorithm (e.g. (**?**)) with vector-space cosine similarity (0), a widely used similarity measure in text analysis. The search can be extended to include more similarity measures and a search over k, the number of groups in clustering. These simply results in more features tested in the regression. If needed, on-the-fly optimization using subsampling and efficient linear time clustering algorithm could be used, but in this paper we did not find them necessary.

An important aspect of optimizing cluster utility in general, and of the use of cluster relations in our setting in particular, is the choice of k, the number of groups into which the entities are clustered. In our case, for each potential value of k we would ideally

[1] See Future Work section for the discussion of how the expressivity of the cluster types can be further increased when attribute vectors of clustered entities are derived from compound relationships.

compute separate clusters. For simplicity and speed in the experiments presented here we fix k to be equal to 100 in all cluster relations except for the last one, `ClusterWordsByDocs`, where the number of clusters is 10. The latter is clustered into fewer groups because there is roughly an order of magnitude fewer objects, words, to be clustered; we selected the vocabulary of size 1,000 to make the size of `HasWord` relation smaller and more manageable.

4. Results

The results presented below demonstrate the advantages of including cluster relations in the statistical relational learning framework. First, including cluster relations in the search space increases the expressivity of the feature space and achieves higher classification accuracy. Second, cluster-based features are cheaper to generate since cluster relations contain fewer tuples than the original relations from which they were derived.

The training set contains 1,000 observations: 500 positive examples of `<Document, Venue>` target pairs uniformly sampled from the `PublishedIn` relation, and 500 negative examples where document is uniformly sampled from the remaining documents and the venue is uniformly sampled from the domain of all venues, such that the sampled venue is not a true venue of the corresponding document. Sampled positive pairs are removed from the background relation `PublishedIn`, as well as the tuples involving documents sampled for the negative set. The test set contains 2,000 examples: 1,000 positive and 1,000 negative, sampled and removed from `PublishedIn` in the same manner as the training set examples. We know the citation structure, authorship and content of the documents for which we are learning to predict publication venues.

To demonstrate the utility of using cluster relations within our statistical relational learning framework we compare two models. One model is learned from the feature space generated from four original non-cluster relations, `PublishedIn`, `Author`, `Citation` and `HasWord`. The other model is learned from the original four relations plus six derived cluster relations, `DocsByAuthors`, `AuthorsByDocs`, `DocsByCitingDocs`, `DocsByCitedDocs`, `DocsByWords` and `WordsByDocs`. Models are learned with sequential BIC feature selection, i.e. as each feature is generated it is added to the model permanently if the BIC statistic improves, or is permanently dropped otherwise. Sequential feature selection is different from standard step-wise model selection. Standard step-wise model selection is infeasible within this framework because it is not known in

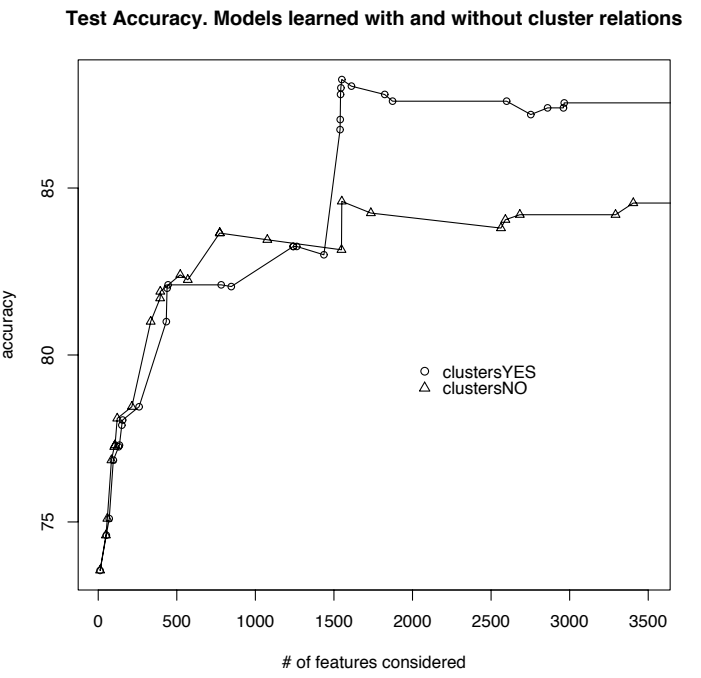

Figure 2. Learning curves show test set (N_{test}=2,000) accuracy changing with the number of features generated and selected from the training set (N_{train}=1,000). Balanced positive/negative priors.

Table 1. Six features in each model which improve test accuracy by at least 1.0 percentage point after being selected. Target: $publishedIn'(D, V)$

Feature	**Model**
$size[publishedIn(\_, V)]$	both
$exists[citation(D, D1), publishedIn(D1, V)]$	both
$exists[citation(D1, D), publishedIn(D1, V)]$	both
$exists[citation(D, D2), citation(D1, D2), publishedIn(D1, V)]$	both
$exists[author(D, A), author(D1, A), publishedIn(D1, V)]$	both
$exists[citation(D, D3), citation(D3, D2), citation(D1, D2), publishedIn(D1, V)]$	`clusterNO`
$exists[publishedIn(D1, V), docsByWords(D, C), docsByWords(D1, C)]$	`clusterYES`

advance which features will be generated next in the feature stream; it is also computationally much more demanding for very large feature streams. Step-wise feature selection needs to check all available feature candidates before adding/dropping one; sequential feature selection re-trains only one additional model per one generated feature.

We learn two models (`clusterNO` and `clusterYES`) using sequential feature selection from the feature streams of size 3,500 of numerically unique features each. A numeric signature of partially evaluated features is maintained to avoid fully generating numerically equivalent (or rather, at least, nearly collinear within hashing error bound) features; note that this is different from avoiding syntactically equivalent nodes of the search space: two different queries can produce numerically equivalent feature columns, e.g. all zeros, which is a common case as feature generation progresses deeper in the search space.

Figure 2 presents learning curves for the two models, learned with and without cluster relations. The curves show test set accuracy changing with the number of features generated and sequentially selected from the training set. The x-axis is the number of features generated from the background relations; points on the curves correspond to the selected features. The model with clusters, `clusterYES`, contains 31 features at the end of the process; 24 features are selected into `clusterNO` from equally many feature candidates (3,500). The number of positive and negative examples is equal in both cases; the curves start from the first added non-intercept feature. Empty models, i.e. only with an intercept, would be 50% accurate. The test set accuracy of the cluster based model after exploring the entire feature stream is 87.55%, which is 3.0 percentage points higher than the accuracy of the model not using cluster relations. The figure suggests that there is no significant overfitting, represented by a decrease in test accuracy when adding a new feature. The largest single drop in accuracy after adding next feature is 0.45 percentage points. Thus, the `clusterYES` model achieves a significant improvement in accuracy and with fewer resources, as the generation of the same number of features for the cluster-based model is cheaper, as the size of cluster relations is smaller that the size of the original relations from which they were derived.

Not all of the cluster relations are equally useful. As Figure 2 shows, the improved accuracy of the cluster-based model comes mostly from a single cluster-based feature, 1540-th in the stream, which made test set accuracy jump by 3.75 percentage points. This feature is a binary feature involving latent document topics, i.e. the cluster relation of documents clustered by their word content. The feature is *ON* for target document/venue pair `<D,V>`, if there exists a document `D1` in the cluster where `D` belongs such that `D1` is published in the same venue as `D`.

Table 1 shows the most significant features learned. *Post factum*, these features are fairly obvious, which is good news for the purposes of validating the methodology and its potential application to less well understood domains. The last item in the table shows the most important cluster-based feature, which translates to English as "if there exists another document which is on the same latent topic as `D` and is published in `V`."

4.1. Cluster-First Search

The cost of database query evaluation used in feature generation dominates the complexity of the SLR methodology. Up to this point we presented models learned when searching the feature space in breadth-first manner. In this section we explore an alternative search strategy which places cluster-based features earlier in the stream. In our venue prediction setting, this strategy achieves the same accuracy as the breadth-first `clusterYES` model with far fewer features tested.

To test the cluster-first search strategy we split the

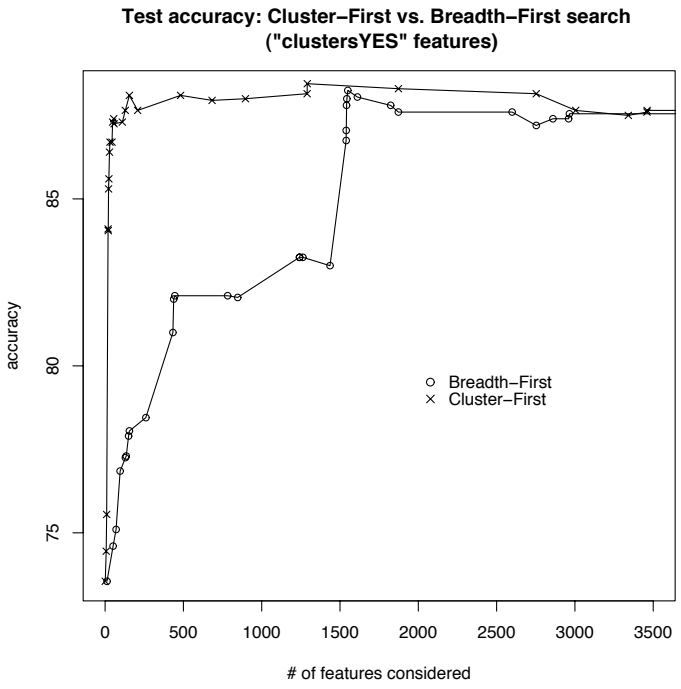

Figure 3. Learning curves show test set (N_{test}=2,000) accuracy changing with the number of features generated and selected from the `clusterYES` training set (N_{train}=1,000) following two search strategies: breadth-first and cluster-first. The latter pushes cluster-based features earlier in the stream. Balanced positive/negative priors.

stream of 3,500 features generated by breadth-first traversal of the `clusterYES` search space into two consecutive substreams. The features of the first "cluster" substream are presented for statistical feature selection first, followed by the features from the second substream. The features in each substream appear in the same relative order as in the original breadth-first stream.

The first stream includes features which involve only cluster-relations and the `PublishedIn` relation. The second stream includes all other features, some of which are based only on `Citation`, `Author` and `HasWord` relations, and others involve cluster relations and `PublishedIn` relation *together* with `Citation`, `Author` and `HasWord` relations. The `PublishedIn` relation is pushed earlier in the stream together with cluster-based relations because of its special status - this relation serves as a *structural core* background relation being of the same type as the response concept. It provides background reference essential for learning, similarly to the role of the `Citation` relation when learning models for link prediction (0; **?**). Because it is not a many-to-many relation there are no "proxy" cluster-relations derived from `PublishedIn`.

Figure 3 presents learning curves for two search strategies, and show test set accuracy a function of the number of features generated and tested in the model.

5. Related Work and Discussion

Clustering and other latent space modeling methods such as PCA often used in propositional predictive modeling as a means for dimensionality reduction. Dimensionality reduction is achieved by replacing the original flat features with the identifiers of clusters they are elements of, or by the coordinates of their projections onto a lower dimensional space. For example, words can be clustered into groups which replace individual words for document classification. structure together with the flat features resulting in more accurate predictive models, for example in the context of maximum entropy modeling (0).

One research direction in relational learning addresses clustering of relational entities with novel distance metrics defined over the interlinked relational representation. Many people have address clustering from relational representation (see e.g. (0)).

It is not our goal to find a single "best" data partitioning. Instead, we identify a number of alternative clusterings which are involved in more complex features improving predictive accuracy of statistical models. Objects may be clustered based on different

attributes, usin different similary measures, and with different numbers of clusters found The usefulness of a grouping can be assessed only in relation to a particular set of predictions being made.

Cluster-based concept and relation invention, as described in this paper, differs importantly from using aggregation, in a sense commonly used in databases, as a means of summarization. Aggregation is essential in statistical relational learning and is also used to create new, rich types of features from relational representation (**?**). Using aggregates creates richer features than modeling a boolean, table empty/non-empty feature as is the case in classical logic-based relational learning approaches (0). The need for aggregates in relational learning comes from the fact that the central type of relational representation is a table (set); the data is represented by a number of tables, and database queries result in tables. Statistical models, on the other hand, work with scalar values, real numbers, integers, or categorical variables. Aggregation allows summarizing information in a table per given observation into a scalar value which can be included in a statistical model, for example, *average* of a word count in all cited documents, or a citing document with *max* number of incoming links. Aggregates are essential to our approach; each node in our search space evaluates into a table, which in turn is aggregated to produce a number of scalar feature candidates. The advantage of clusters comes at another level to create central relational entities from which features are generated; aggregates are applied at the next step to the tables resulting from queries which can involve both the cluster relations and the original relations, for example, the number of documents in the same cluster and published in the same conference is a *count* or *size* aggregate of a corresponding derived view.

The idea of augmenting the existing representation with new relations or predicates is, of course, not new. In inductive logic programming it is known as "predicate invention". For example, Statistical Predicate Invention (0) which was proposed for learning in hypertext domains, represents classifications produced by Naive Bayes as a new predicate added to FOIL (**?**). Statistical Predicate Invention preserves FOIL as the central modeling component and calls statistical modeling from within the inner structure navigation loop to supply new predicates. Our approach differs in that we use statistics rather than logic as a modeling component, and more importantly in this context, we advocate the use of cluster-based relation invention as a means to enrich feature spaces by adding to schema many types of clusters, not only those of a response concept, thus creating first-class relational concepts, such as "topics" or "communities", which have a clean "identity" as the world representation entities.

Concept invention could also, in theory be done in other types of relational learning, such as in those using graphical models, e.g. Probabilistic Relational Models (PRMs) (**?**; 0), which are generative models of joint probability distribution capturing probabilistic influences between entities and their attributes in a relational domain. However, such generative models are not condusive to searching for complex features, as is done in ILP and in this paper.

6. Conclusions and Future Work

We presented a framework for learning predictive statistical models from relational data where new concepts and relations are derived by clustering items in the original database schema. Adding these new relations to the database schema (as opposed to just using them as aggregates to derive new features) allows a more efficient search of a richer feature space. Including new relations such as $docsByWords$ allows discovery of new features such as $exists[publishedIn(D1, V), docsByWords(D, C), docsByWords(D1, C$

We applied cluster-relation invention to the task of predicting the publication venue of scientific papers from the CiteSeer database, which contains citations, paper authorship, and word content. We used clustering to derive new first class relational entities reflecting hidden topics of papers, author communities and word groups. New cluster relations included into the feature generation process, in addition to the original relations, resulted in the creation of richer cluster-based features, where clusters enter into more complex relationships with existing background relations rather than only provide dimensionality reduction. Using relation invention gives more accurate models than those built only from the original relational concepts.

Two models, with and without cluster relations, were compared on feature streams of 3,500 unique features. Relation invention gave accuracy improvement of about 3.0 percentage points over the non cluster-based model. Enriching the schema with more compact cluster relations can also lead to a more rapid discovery of predictive features. Cluster relations are smaller than the original relations from which they are derived; this translates into lower costs of generating cluster-based features. Focussing search on relations containing clusters, we get high accuracy in fewer than 100 features, in contrast to the breadth-first strategy which achieved the same accuracy only after considering more than 1,500 features. This is important, as the

most computationally demanding process in the SLR methodology is database query evaluation for feature generation.

We envision several improvements to the relation invention methodology. Richer types of clusters can be derived from more complex sets of attributes than those immediately available in a single relation. For example, publication venues and authorship data are in two separate relations which both can be used to cluster publication venues based on the authors who publish in them. Also, clustering can be performed lazily as a corresponding depth in the feature search space is reached by the feature generation process. In contrast to "propositionalization" (**?**), which implies a decoupling of relational feature generation and modeling, SLR is dynamic and allows for a more natural introduction of this extension.

References

Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

A. V. Aho, Y. Sagiv, and J. D. Ullman. Equivalences among relational expressions. *SIAM Journal of Computing*, 8(2):218–246, 1979.

M. Craven and S. Slattery. Relational learning with statistical predicate invention: Better models for hypertext. *Machine Learning*, 43(1/2):97–119, 2001.

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.

Dean Foster and Lyle Ungar. A proposal for learning by ontological leaps. In *Proc. of Snowbird Learning Conference*, Snowbird, Utah, 2002.

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proc. of IJCAI99*, pages 1300–1309, Stockholm, Sweden, 1999.

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.

L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, 1990.

Mathias Kirsten, Stefan Wrobel, and Tamas Horvath. Distance based approaches to relational learning and clustering. In Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*, pages 213–230. Springer-Verlag, 2001.

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.

Stephen Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19,20:629–679, 1994.

Werner Nutt, Yehoshua Sagiv, and Sara Shurin. Deciding equivalences among aggregate queries. In *Proc. of PODS-98*, pages 214–223, 1998.

Dmitry Pavlov, Alexandrin Popescul, David M. Pennock, and Lyle H. Ungar. Mixtures of conditional maximum entropy models. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, 2003.

Claudia Perlich and Foster Provost. Aggregation-based feature invention and relational concept classes. In *Proc. of KDD-2003*, 2003.

Alexandrin Popescul and Lyle H. Ungar. Statistical relational learning for link prediction. In *IJCAI Workshop on Learning Statistical Models from Relational Data*, 2003.

Alexandrin Popescul and Lyle H. Ungar. Structural logistic regression for link analysis. In *KDD Workshop on Multi-Relational Data Mining*, 2003.

Alexandrin Popescul, Lyle H. Ungar, Steve Lawrence, and David M. Pennock. Statistical relational learning for document mining. In *Proceedings of IEEE International Conference on Data Mining (ICDM-2003)*, 2003.

J.R. Quinlan and R.M. Cameron-Jones. Induction of logic programs: FOIL and related systems. *New Generation Computing*, 13:287–312, 1995.

G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw Hill, 1983.

E. Shapiro. *Algorithmic Program Debugging*. MIT Press, 1983.