

# Cluster-based Concept Invention for Statistical Relational Learning

Alexandrin Popescul\*  
Computer and Information Science  
University of Pennsylvania  
Philadelphia, PA 19104  
popescul@cis.upenn.edu

Lyle H. Ungar  
Computer and Information Science  
University of Pennsylvania  
Philadelphia, PA 19104  
ungar@cis.upenn.edu

## ABSTRACT

We use clustering to derive new relations which augment database schema used in automatic generation of predictive features in statistical relational learning. 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 on two tasks, venue prediction and link prediction, using CiteSeer data.

**Categories and Subject Descriptors:** I.2.6 [Artificial Intelligence]: Learning

**General Terms:** Algorithms.

**Keywords:** Relational Learning, Clustering, Feature Generation.

## 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 relational entities participating in the search.

---

\* Now at Ask Jeeves Inc., 1551 S. Washington Avenue, Suite 400, Piscataway, NJ 08854.

In this paper we argue that 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. 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 [3].

## 2. STRUCTURAL LOGISTIC REGRESSION

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 which 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.

In the rest of the section we briefly describe SLR. A more detailed description of the method and the specification of its feature generation algorithm is given in [11]. Section 2.1 describes the similarity measure we use for clustering.

SLR is an extension of logistic regression to modeling relational data. It combines the strengths of classical statistical

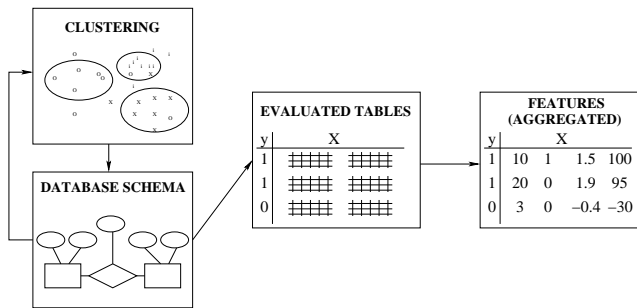


Figure 1: Cluster-relations augment database schema used to produce feature candidates.

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 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 [16, 2] 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. Each node of the refinement graph produces multiple feature candidates. 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.

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.

## 2.1 Similarity Measure used in Clustering

Throughout the experiments in this paper we use  $k$ -means clustering algorithm (e.g. [5]) with vector-space cosine similarity [14]. In the formulas presented here, document  $d$  can stand for any object type we want to cluster, and words  $w$  are the attributes which are used to cluster  $d$ . For example, authors  $d$ , can be clustered using the documents  $w$  they write.

Each document is viewed as a vector whose dimensions correspond to words in the vocabulary; the component magnitudes are the  $tf$ - $idf$  weights of the words.  $Tf$ - $idf$  is the product of term frequency  $tf(w, d)$ —the number of times word  $w$  occurs in the corresponding document  $d$ —and inverse document frequency

$$idf(w) = \log \frac{|D|}{df(w)},$$

where  $|D|$  is the number of documents in a collection and  $df(w)$  is the number of documents in which word  $w$  occurs at least once. The similarity between two documents is then

$$sim(\mathbf{d}_i, \mathbf{d}_j) = \frac{\mathbf{d}_i \cdot \mathbf{d}_j}{\|\mathbf{d}_i\| \|\mathbf{d}_j\|},$$

where  $\mathbf{d}_i$  and  $\mathbf{d}_j$  are vectors with  $tf$ - $idf$  coordinates as described above.

In the experiments reported here we use *binary tf-idf*, where the  $tf$  component is a binary indicator. We use this measure to be consistent with the background relation `HasWord`. Since we are not using counts in the original relation `HasWord`, we do not bring in this additional information into the attributes used to cluster documents. Other derived cluster relations use naturally binary attributes, e.g. citation or authorship based clusters. Other similarity measures and variants of  $tf$ - $idf$  exist.

The search can be extended to include several types of similarity measures and a search over  $k$ , the number of groups in clustering. These would result in a higher number of 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.

## 3. TASKS AND DATA

We explore two tasks using CiteSeer data: classifying documents into their publication venues, conferences or journals, and predicting the existence of a citation between two documents. The target concept pair is `<Document, Venue>` and `<Document, Document>` respectively. In the case of venue prediction, 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. Similarly, in link prediction, value of the response variable is one if there exists a citation between two documents and it is zero otherwise. In both tasks, 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. The following are descriptions of basic relations we use, followed by a description and discussion of the derived cluster relations we use to augment the search space:

- `PublishedIn(doc:Document, vn:Venue)`. Publication venues are extracted by matching information with the DBLP database, <http://dblp.uni-trier.de/>. Publication venues are known for 60,646 CiteSeer documents. All other relations are populated with information about these documents. There are

1,560 unique conferences and journals. Training and test examples are sampled from this background relation. Relation size: 60,646.

- `Author(doc:Document, auth:Person)`. 53,660 out of the total of 60,646 documents have authorship information available; there are 26,740 unique last names of authors. Relation size: 131,582.

- `Citation(from:Document, to:Document)`. This relation contains all citations among our “universe” of 60,646 documents. It contains 42,749 unique citing documents, 31,603 unique cited documents, and the total of 49,398 documents. Relation size: 173,410.

- `HasWord(doc:Document, word:Word)`. This is by far the largest relation even for relatively small vocabularies. It is populated by binary word occurrence vectors, i.e. there is a tuple for each word in the vocabulary if it is contained in a corresponding document. The relation contains word data available for 56,104 documents, the size of vocabulary is 1,000 words. (the vocabulary contains top count words in the entire collection after Porter stemming and stop word removal). Relation size: 6,894,712.

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>`. Cluster relations can be generated lazily, or they can be precomputed and added to the relational schema before feature generation phase. Efficient clustering algorithms can be used for document clustering based on the regularities characteristic of citation structure in corpora of scientific publications [12].

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:

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

- `ClustAuthorsByDocs(auth:Person, clust:Clust1)`  
26,740 authors are clustered based on 53,660 documents they wrote. Relation size: 26,740.

- `ClustDocsByCitingDocs(doc:Document, clust:Clust2)`  
31,603 documents are clustered based on 42,749 documents citing them (the numbers are slightly lower in link prediction where target concept links do not participate in clustering). Relation size: 31,603.

- `ClustDocsByCitedDocs(doc:Document, clust:Clust3)`  
42,749 documents are clustered based on 31,603 documents cited from them (the numbers are slightly lower in link prediction where target concept links do not participate in clustering). Relation size: 42,749.

- `ClustDocsByWords(doc:Document, clust:Clust4)`  
56,104 documents are clustered based on the vocabulary of top 1,000 words. Relation size: 56,104.

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

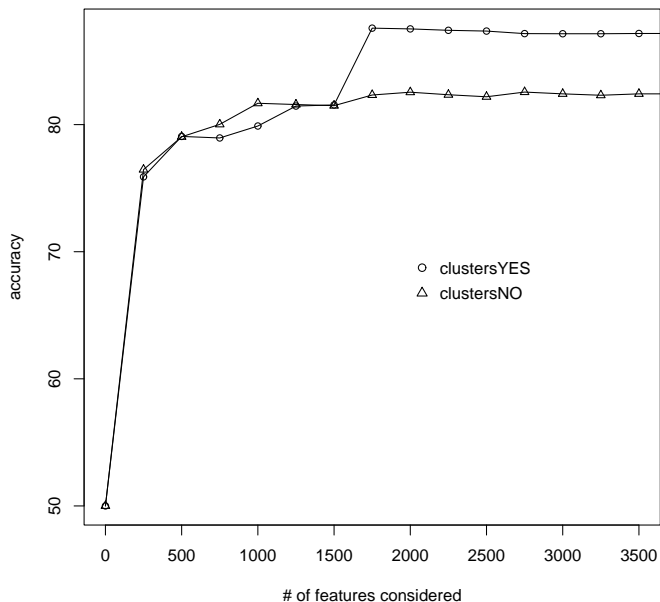
An important aspect of optimizing cluster utility in general, and of the use of cluster relations in our setting in particular, is the choice of  $k$ , the number of groups into which the entities are clustered. In our case, for each potential value of  $k$  we would ideally compute separate clusters. 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, `ClustWordsByDocs`, where the number of clusters is 10. The latter is clustered into fewer groups than the rest of the clusters to reflect the fact that 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. The accuracy of the cluster-based models reported below can potentially be improved even further if one is willing to incur the additional cost of optimizing the choice of  $k$ .

## 4. RESULTS

We compare models learned from the feature space generated from four original non-cluster relations with the models learned from the original four relations plus six derived cluster relations (`clustersNO` and `clustersYES` models). Models are learned with sequential feature selection which uses Bayesian Information Criterion (BIC) [15], i.e. as each feature is generated it is added to the model permanently if the BIC improves, or is permanently dropped otherwise. Sequential feature selection differs from standard step-wise model selection in that the latter requires knowing in advance all features which will be generated; step-wise model selection is much more expensive as it requires the computation of the objective function for all available feature candidates when deciding which one to add or drop next, sequential feature selection, on the other hand, re-trains only one additional model per one generated feature.

The size of feature streams used in training each model is set to 3,500 numerically unique features. A numeric signature of partially evaluated features is maintained to avoid fully generating numerically equivalent (or rather, at least, nearly collinear within hashing error) 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.

We use 10-fold cross validation to show accuracy improvement when using cluster-relations and to derive error bounds on the improvement. All observations are split equally into 10 sets. Each of the sets is used to train a model. Each of the models is tested on the remaining observations. This results in 10 values per each tested level, which are used to derive error bounds. In venue prediction, the total number of observations is 10,000: 5,000 positive examples of `<Document, Venue>` target pairs uniformly sampled from the relation `PublishedIn`, and 5,000 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 document. Sampled positive pairs are removed from the background relation `PublishedIn`, as well as the tuples involving documents sampled for the negative set. The size of the background relation `PublishedIn` reduces by 10,000 after removing tuples involved in training and test

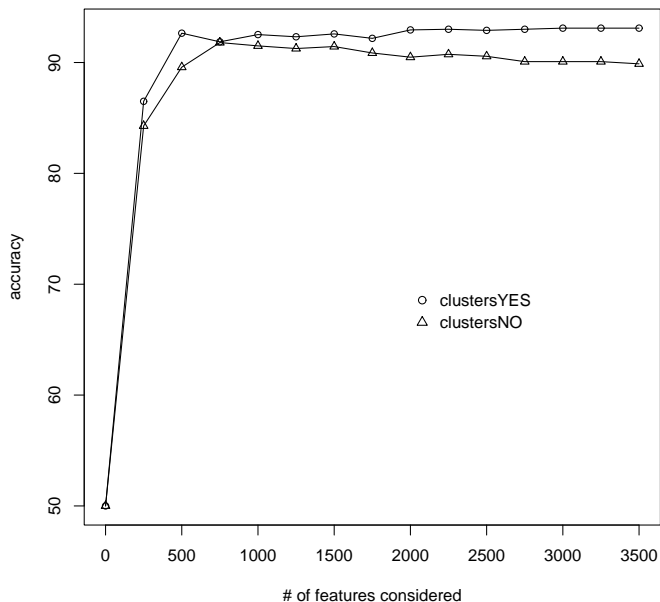


**Figure 2: Learning curves: venue prediction average test set accuracy against the number of features generated from the training sets in 10-fold cross validation (in each of 10 runs  $N_{train} = 1,000$  and  $N_{test} = 9,000$ ). Balanced positive/negative priors**

sets. In link prediction, the total number of observations is 5,000: 2,500 positive examples of  $\langle \text{Document}, \text{Document} \rangle$  target pairs uniformly sampled from the `Citation` relation, and 2,500 negative examples uniformly sampled from empty links in the citation graph. Sampled positive pairs are removed from the background relation `Citation`. The size of the background relation `Citation` reduces by 2,500, the number of sampled positive examples.

Figure 2 and Figure 3 present test accuracy learning curves for models learned with and without cluster relations in venue prediction and link prediction respectively. Curve coordinates are averages over the runs in 10-fold cross validation (see below separate figures and discussion of error bounds of the difference between accuracies in both models). The learning curves show test set accuracy changing with the number of features, in intervals of 250, generated and sequentially selected from the training set. The average test set accuracy of the cluster based models after exploring the entire feature stream is 87.2% in venue prediction and 93.1% in link prediction, which is, respectively, 4.75 and 3.22 percentage points higher than the average accuracy of the models not using cluster relations.

Figure 4 and Figure 5 present Gaussian 95% confidence intervals of the difference in mean test accuracies of `clusterYES` and `clusterNO` models in venue prediction and link prediction respectively. In venue prediction, after exploring approximately half of the feature stream the improvement in accuracy by the cluster-based models is statistically significant at 95% confidence level according to the  $t$ -test (confidence intervals do not intersect with  $y=0$ ). In the early



**Figure 3: Learning curves: link prediction average test set accuracy against the number of features generated from the training sets in 10-fold cross validation (in each of 10 runs  $N_{train} = 500$  and  $N_{test} = 4,500$ ). Balanced positive/negative priors**

feature generation, when considering the streams of about 1,000 features, cluster-based models perform significantly worse: at this learning phase, additional cluster-based features while not yet significantly improving accuracy may delay the discovery of significant non-cluster based features. In link prediction, while the significance of the improvement from cluster-based features is reduced early in the stream, it continuously increases throughout the rest of the stream. At the end of the stream the improvement in accuracy of the cluster-based model is 3.22 percentage points, statistically significant at the 99.8% confidence level. The highest level accuracies (after seeing 750 features by `clustersNO` and after seeing 3500 features by `clustersYES`) also statistically differ: the accuracy improvement in cluster-based models is 1.49 percentage points, significant at the 99.9% confidence level. The average number of features selected in 10 `clusterYES` models is 32.0 in venue prediction and 32.3 in link prediction; respectively, 27.9 and 31.8 features on average were selected into `clusterNO` models from equally many feature candidates (3,500).

The improved accuracy of the cluster-based model in venue prediction comes mostly from a single cluster-based feature. This feature was selected in all cross validation runs. It 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  $\langle D, V \rangle$ , if there exists a document  $D1$  in the cluster where  $D$  belongs such that  $D1$  is published in the same venue as  $D$ . Using a logic based notation, the feature is the following

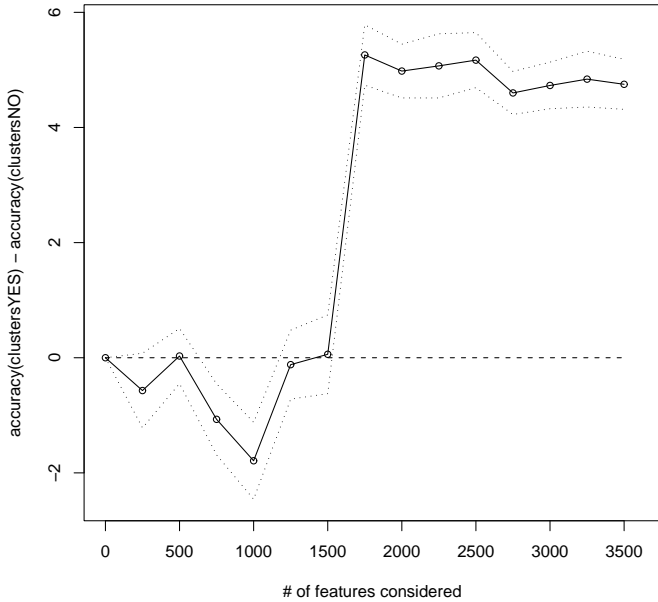


Figure 4: Mean venue prediction accuracy difference,  $accuracy(clustersYES) - accuracy(clustersNO)$  with 95% confidence intervals (bounds based on  $N=10$  points,  $t$ -test distribution)

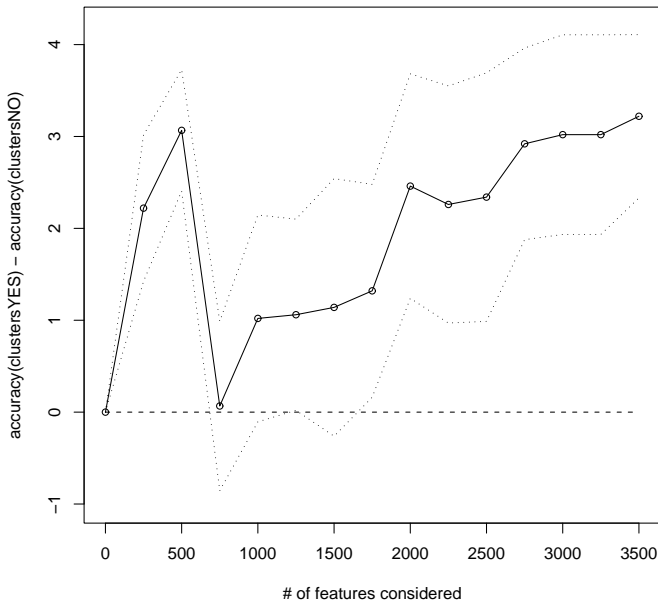


Figure 5: Mean link prediction accuracy difference,  $accuracy(clustersYES) - accuracy(clustersNO)$  with 95% confidence intervals (bounds based on  $N=10$  points,  $t$ -test distribution)

(abbreviate here `clustDocsByWords` by `topic`):<sup>1</sup>

$exists[publishedIn(D1, V), topic(D, C), topic(D1, C)]$ .

The following are examples of other significant features automatically generated and selected in the venue prediction task: document  $D$  is more likely to be published in a conference or journal  $V$ : i) if  $V$  is a large venue, i.e. publishes many papers ( $size[publishedIn(\_, V)]$ ), ii) if document  $D$  cites another document which is published in the same venue  $V$ , iii) if document  $D$  is cited by another document which is published in the same venue  $V$ , iv) if the author of  $D$  published another paper in the same venue  $V$ .

The following three cluster-based features were selected in more than five cross validation runs (9, 9 and 6 times respectively) in the link prediction task (target:  $\langle D1, D2 \rangle$ ):

$exists[docsByCitedDocs(D1, C), docsByCitedDocs(D2, C)]$ ,  
 $exists[docsByWords(D1, C), docsByWords(D2, C)]$ ,  
 $exists[docsByCitingDocs(D1, C), docsByCitingDocs(D2, C)]$ .

## 5. RELATED WORK AND DISCUSSION

Clustering and other latent space modeling methods such as principal components analysis 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 replacing individual words for document classification. Structure together with the flat features can result in more accurate predictive models, for example in the context of maximum entropy modeling [9].

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. [6]). 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, using different similarity 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 [10]. 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 [8]. 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 vari-

<sup>1</sup>Note that  $D1$  is distinct from  $D$  as the tuple with publication venue of document  $D$  is removed from the background relation `PublishedIn`.

ables. 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.

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 [1] which was proposed for learning in hypertext domains, represents classifications produced by Naive Bayes as a new predicate added to FOIL [13]. Our approach differs in that we use statistics rather than logic as a central 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) [4], 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 conducive to searching for complex features, as is done in inductive logic programming and in this paper.

## 6. CONCLUSIONS AND FUTURE WORK

We presented a framework for learning predictive statistical models from relational data where alternative “cluster-relations” are derived from the attributes in the original database schema and included in the feature generation process. The method was used for predicting the publication venue of scientific papers from the CiteSeer data including the citation graph, 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 relations.

Adding cluster-derived concepts as relations to a database schema used to generate features can increase the predictive accuracy of statistical relational models. Section 4 presents experimental results for venue prediction and link prediction. In both tasks, we have shown that models built using a schema augmented with cluster-derived relations result in statistically significant increase in accuracy over the models built using the original relational schema.

Two models, with and without cluster relations, were compared on feature streams of 3,500 unique features. In the middle of the feature generation process for the venue prediction task, the cluster-based feature generation was able to discover a highly significant feature from the clusters of documents grouped by their word content which contributed a mean of 4.75 percentage point increase in the final test set

accuracy. Through the end of the feature generation process the cluster-based model maintained a stable accuracy improvement, and was statistically significant at the 95% confidence level based on the *t*-test over 10-fold cross validation. The average improvement from using cluster-derived features at the end of the link prediction feature stream was 3.22 percentage points, also statistically significant at the 95% confidence level.

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. In contrast to “propositionalization” [7], which implies a decoupling of relational feature generation and modeling, SLR is dynamic and allows for a more natural introduction of this extension.

## 7. REFERENCES

- [1] M. Craven and S. Slattery. Relational learning with statistical predicate invention: Better models for hypertext. *Machine Learning*, 43(1/2):97–119, 2001.
- [2] 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.
- [3] Dean Foster and Lyle Ungar. A proposal for learning by ontological leaps. In *Proc. of Snowbird Learning Conference*, Snowbird, Utah, 2002.
- [4] 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.
- [5] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, 1990.
- [6] 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.
- [7] 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.
- [8] Stephen Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19,20:629–679, 1994.
- [9] Dmitry Pavlov, Alexandrin Popescul, David M. Pennock, and Lyle H. Ungar. Mixtures of conditional maximum entropy models. In *Proc. of ICML-2003*, 2003.
- [10] Claudia Perlich and Foster Provost. Aggregation-based feature invention and relational concept classes. In *Proc. of KDD-2003*, 2003.
- [11] Alexandrin Popescul. *Statistical Learning from Relational Databases, PhD thesis*. University of Pennsylvania, 2004.
- [12] Alexandrin Popescul, Gary Flake, Steve Lawrence, Lyle H. Ungar, and C. Lee Giles. Clustering and identifying temporal trends in document databases. In *Proc. of the IEEE Advances in Digital Libraries*, 2000.
- [13] J.R. Quinlan and R.M. Cameron-Jones. Induction of logic programs: FOIL and related systems. *New Generation Computing*, 13:287–312, 1995.
- [14] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw Hill, 1983.
- [15] Gideon Schwartz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1979.
- [16] E. Shapiro. *Algorithmic Program Debugging*. MIT Press, 1983.