Scalable Relational Learning for Sparse and Incomplete Domains

Yi Huang, Volker Tresp yihuang@siemens.com, volker.tresp@Siemens.com
Siemens AG, Corporate Technology, Information and Communications, Learning Systems, Munich, Germany

Markus Bundschus bundschu@dbs.ifi.lmu.de
Ludwig-Maximilians University Munich, Germany

Achim Rettinger achim.rettinger@cs.tum.edu
Technical University of Munich, Germany

1. Introduction

The Semantic Web (SW) presents new challenges to statistical relational learning. One of the main features of SW data is that it is notoriously incomplete. Consider friend-of-a-friend (FOAF) data. The purpose of the FOAF project is to create a web of machine-readable pages describing people, their relationships, and people's activities and interests, using SW technology. Obviously people vary in their motivation to communicate private information such that person-specific information varies from almost nothing to great detail. Another feature of SW data is the great sparsity of relational information such as friendship: listed friends in the knowledge base are a small subset of all potential friends. Finally, there is a variety of additional untested and potentially conflicting schema information specified by users. We are expecting that the data situation in the FOAF setting will be characteristic for many other SW domains. Naturally, missing information can be supplemented. First, deductive reasoning can be used to complement factual knowledge based on axioms. The advantage is that deduced facts can be queried with powerful query languages such as SQL for relational data bases and SPARQL for SW data. A common problem is that, generally, valid axioms are rare and many true facts in an application cannot be derived. Inductive reasoning, the focus of this paper, is the second way for gaining additional knowledge by exploiting regularities in the data. The advantage of inductive reasoning is that one can perform inference on a large number of statements. Disadvantages are that only the *probability* of the truth values of statements can be derived and that machine learning needs to be performed by machine learning experts. The goal of the work presented here is to apply statistical relational learning in sparse and unreliable domains such as the SW. Several requirements need to be fulfilled. The statistical relational learning algorithms need to be able to deal with sparse data typical for SW domains, they must be scalable, they need to be easily configurable, finally it should be possible to integrate the learning results into querying. For the user there should be almost no difference between querying facts, deduced facts and induced information.

We define an extension of SPARQL, which allows the integration of the learned probabilistic statements into querying. Statements that can be inferred via *logical* reasoning can readily be integrated into learning and querying. We study learning algorithms that are suitable for the resulting high-dimensional sparse data matrix. We present experimental results using a friend-of-a-friend data set.

The work on inductive databases (Imielinski & Mannila, 1996; Raedt et al., 2002; Kramer et al., 2006) pursues similar goals but is focussed on the less-problematic data situation in relational databases. In (Kiefer et al., 2008) the authors describe SPARQL-ML, a framework for adding data mining support to SPARQL but without considering the special data situation on the SW.

2. Learning Procedure

To define an appropriate statistical setting, we require the user to define key entities (i.e., statistical units) and a population. Key entities are the entities (e.g. persons) that are the source of the variables or features of interest. A population is the set of key entities, for which statistical inference is performed. The population might be defined in various ways. For example, it might concern all persons in a particular country or, alternatively, all female students at a particular university. In a statistical analysis only a subset of the population is made available for investigation, i.e., a

sample. A data matrix is generated in which the rows correspond to the key entities in the sample and the columns correspond to features that represent actual and potential relations of the key entities.[1]

The resulting data matrix is typically quite large, binary and sparse. A *one* stands for a statement known to be true and a *zero* for a statement whose truth value is unknown. Such a data situation has been studied in various context in the past and a number of matrix completion methods have proven to be successful in this context. We investigate matrix completion based on an eigenvector analysis of the data matrix (SVD), matrix completion based on non-negative matrix factorization (NNMF) (Lee & Seung, 1999) and matrix completion using latent Dirichlet allocation (LDA) (Blei et al., 2003). All three approaches estimate unknown matrix entries via a low-rank matrix approximation. SVD is based on a singular value decomposition and NNMF is a decomposition under the constraints that all terms in the factoring matrices are non-negative. LDA is based on a Bayesian treatment of a generative topic model. After matrix completion, the entries are interpreted as certainty values that the corresponding statements are true. After training, the models can be applied to key entities in the population outside the sample.

3. SPARQL Extension

```
PREFIX ex: http://example.org/
SELECT ?actor
WHERE
  { ?movie  ex: FilmedIn  ?city
    ?city   ex: InCountry ex: Italy
    ?actor  ex: ActIn     ?movie
  }
```

```
PREFIX ex: http://example.org/
SELECT ?actor
 WHERE
  { ?movie  ex: FilmedIn  ?city
    ?city   ex: InCountry ex: Italy
    ?actor  ex: ActIn     ?movie  WITH PROB
?prob
  }
ORDER BY ?prob
```

Figure 1. Top: A SPARQL query. Bottom: An extended SPARQL query that includes probabilistic information.

We now discuss how SPARQL needs to be extended to be able to incorporate the derived probabilities. As an example, we consider a challenge that was posed at a recent LarKC (LarKC, 2008) consortium meeting. First consider a regular SPARQL query that *finds all actors that act in movies that are filmed in an Italian city* (Figure 1, Top). With learned probabilistic triples we can pose the question: *Find all actors that are likely to act in movies that are filmed in an Italian city* (Figure 1, Bottom). Note that we have added the keywords `WITH PROB`. The variable `?prob` assumes the value 1 for explicit triples or triples derived from ontological reasoning and assumes the estimated probabilities for the learned triples. `ORDER BY` returns first the actors, for which it is known for certainty that they have acted in movies that are filmed in an Italian city and then returns actors sorted by the probabilistic labels `?prob`. The keyword `DISTINCT` can be employed to remove redundancy.

4. Experiments

4.1. Data Set and Set Up

Data Set: The experiments are based on friend-of-a-friend (FOAF) data. We selected 636 persons with a "dense" friendship information. On average, a given person has 18 friends. Numerical values such as *date of birth* or the *number of blog posts* were discretized. The resulting data matrix, after pruning columns with few *ones*, has 636 persons (rows) and 491 columns.[2] 462 of the 491 columns (friendship attributes) refer to the property *knows* . The remaining columns (general attributes) refer to general information about age, location, number of blog posts, attended school, etc.

Evaluation Procedure and Evaluation Measure: The task is to predict potential friends of a person. For each person in the data set, we randomly selected one known friendship statement and set the corresponding matrix entry to zero, to be treated as unknown (test statement). In the test phase we then predict all unknown friendship entries, including the entry for the test statement. The test statement should obtain a high likelihood value, if compared to the other unknown friendship entries. Here we use the normalized discounted cumulative gain (NDCG) (Jarvelin & Kekalainen, 2000) to evaluate a predicted ranking. To focus more on the top-ranked items, we also consider the *NDCG@n* which only counts the top n items in the rank list. These scores are averaged over all functions for comparison.

Benchmark methods: *Baseline:* Here, we create a random ranking for all unknown triples, i. e. every unknown triple gets a random probability assigned. *SVM:* We use the one-class support vector machine. Two different input feature sets were examined: one

[1] An example is given in the experimental section.

[2] Columns with fewer than 3 ones were removed.

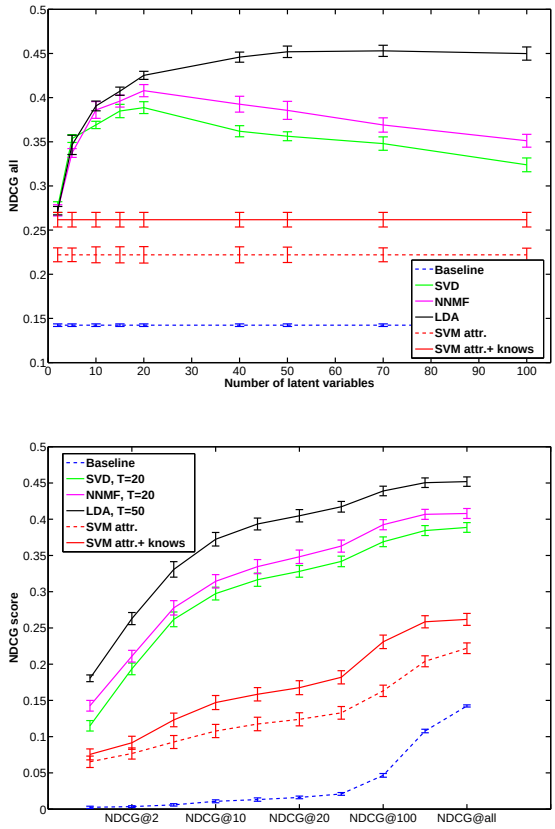

Figure 2. NDCG comparison between different algorithms. Top: *NDCG all* is plotted against the number of latent variables. Bottom: *NDCG@n* score for different thresholds.

contains only general attributes of persons (such as age, location and number of blog posts) (*SVM attr.*) and the second one contains, in addition, the friendship information to all persons (*SVM attr.+knows*).

4.2. Results

Figure 2 (top) plots the *NDCG all* score of all algorithms against the number of latent variables. All three matrix completion methods clearly outperform the benchmark algorithms, while LDA outperforms the two other matrix completion algorithms NNMF and SVD. In addition, LDA is not very sensitive to the predefined number of latent variables as long as the number is reasonably high. LDA reaches it maximum *NDCG all* score with $T = 50$ latent variables and the performance does not deteriorate when the number of latent factors is increased. In contrast, the two other matrix completion methods are sensitive with respect to the predefined number of latent variables. They both reach the maximum with $T = 20$. Figure 2 (bottom) plots the *NDCG@n* score against thresholds n. Again, LDA performs best at every threshold n and the two SVM settings are inferior to all matrix completion methods.

5. Conclusion and Outlook

We have presented a generic learning approach for deriving probabilistic SW statements and have demonstrated how these can be integrated into an extended SPARQL query. The learning process is to a large degree autonomous. Only the key entity and the population need to be defined by a user. Since the number of columns in the learning matrix is rather independent of the overall size of the SW and since the sample size can be controlled, learning is essentially independent of the overall size of the SW. The generalization from the sample to the population is linear in the size of the population. In our experiment based on the FOAF data set, LDA showed best performance, which we attribute to the fact that LDA, in contrast to NNMF and SVD, uses a Bayesian approach, which has a smaller tendency to overfitting. As confirmed by our experiments, support vector machines do not exhibit competitive performance in learning high-dimensional relational data.

References

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, *3*.

Imielinski, T., & Mannila, H. (1996). A database perspective on knowledge discovery. *Comm. Of The Acm*, *39*, 58–64.

Jarvelin, K., & Kekalainen, J. (2000). IR evaluation methods for retrieving highly relevant documents. *SIGIR'00*.

Kiefer, C., Bernstein, A., & Locher, A. (2008). Adding data mining support to sparql via statistical relational learning methods. *ESWC 2008*. Springer-Verlag.

Kramer, S., Aufschild, V., Hapfelmeier, A., Jarasch, A., Kessler, K., Reckow, S., Wicker, J., & Richter, L. (2006). Inductive databases in the relational model: The data as the bridge. *KDID*.

LarKC (2008). *The large knowledge collider*. EU FP 7 Large-Scale Integrating Project, http://www.larkc.eu/.

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*.

Raedt, L. D., Jaeger, M., Lee, S. D., & Mannila, H. (2002). A theory of inductive query answering. *ICDM*.