Materializing and Querying Learned Knowledge

Volker Tresp[1], Yi Huang[1], Markus Bundschus[2], and Achim Rettinger[3]

[1] Siemens AG, Corporate Technology, Munich, Germany
[2] Ludwig-Maximilians University Munich, Germany
[3] Technical University of Munich, Germany

Abstract. In many Semantic Web domains a tremendous number of statements (expressed as triples) can potentially be true but, in a given domain, only a small number of statements is known to be true or can be inferred to be true. It thus makes sense to attempt to estimate the truth values of statements by exploring regularities in the Semantic Web data via machine learning. Our goal is a "push-button" learning approach that requires a minimum of user intervention. The learned knowledge is materialized off-line (at loading time) such that querying is fast. We define an extension of SPARQL for the integration of the learned probabilistic statements into querying. The proposed approach deals well with typical properties of Semantic Web data. i.e., with the sparsity of the data and with missing data. 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.

1 Introduction

In many Semantic Web (SW) domains a tremendous amount of statements (expressed as triples) might be true but, in a given domain, only a small number of statements is known to be true or can be inferred to be true. It thus makes sense to attempt to estimate the truth values of statements by exploring regularities in the SW data with machine learning, which is the topic of this contribution. The presented work is an integral part of the LarKC project [1] for the development of large-scale reasoning and learning for the SW. In LarKC a number of requirements have been stated. First, machine learning should be "push-button" requiring a minimum of user intervention. Second, learning time should scale well with the size of the SW. Third, the statements and their probabilities, which are predicted from machine learning, should easily be integrated into SPARQL-type querying. Finally, machine learning should be suitable to the data situation on the SW with sparse data (e.g., only a small number persons are friends) and missing information (e.g., some people don't reveal private information).

A number of algorithms have been proposed in the past for learning in the SW, many of which are based on recent work in statistical relational learning (see [2] for a recent overview). One family of approaches formulates a global probabilistic model for a segment of a Semantic Web knowledge-base (SW-KB) and is able to predict the probability of statements in the domain (examples are [3–6]). In these approaches, the states of probabilistic nodes in a graphical model represent the truth values of statements. Although these approaches are quite attractive, we fear that the sheer size of the SW and the huge number of potentially true statements make these approaches inappropriate in many large-scale applications. The second family of approaches consists of conditional models. Here, a classification problem is defined and one attempts to derive appropriate relational features that can be used for predicting the target class. These approaches include Inductive Logic Programming (ILP) [7, 8] and propositionalized ILP approaches [9, 10]. Since the sample size be controlled, scalability is easily achieved but conditional models have problems with missing data.

In this paper we pursue a compromise between global probabilistic models and the conditional models. As in some of the global probabilistic models, we introduce probabilistic nodes whose states reflect the truth value of the corresponding statements. We derive a data matrix for model training. This data matrix is typically high-dimensional and sparse and we apply recently developed matrix completion approaches for estimating the missing information. Since the data matrix is typically independent or only weakly dependent on the overall size of the SW, training time is essentially independent of the overall size of the SW.

The paper is organizes as follows. In the next section we discuss related work and in Section 3 we review relevant facts about the SW and discuss reasoning via inferred closure. In Section 4 we discuss how machine learning can be applied to derive probabilistic weights for statements whose truth values are unknown and introduce our approach. In Section 5 we discuss extensions to SPARQL that would lead to sensible queries and include the probabilistic values derived from machine learning. In Section 6 we present experimental results using friend-of-a-friend (FOAF) data. Finally, Section 7 contains conclusions and outlines further work.

2 Related Work

The work on inductive databases [11] pursues similar goals but is focussed on the less-problematic data situation in relational databases. In [12] the authors describe SPARQL-ML, a framework for adding data mining support to SPARQL. SPARQL-ML was inspired by Microsoft's Data Mining Extension (DMX). A particular ontology for specifying the machine learning experiment is developed. The SRL methods in [12] are ILP-type approaches based on a closed-world assumption (relational Bayes Classifier (RBC) and Relational Probabilistic Trees (RPT)). This is in difference to the work presented here, which maintain more of an open-world assumption that is more appropriate in the context of the SW. Also, the matrix completion approaches used in our approach have been demonstrated to provide superior performance in many high-dimensional relational prediction tasks [13]. Another difference is that in the work presented here, both model training and statement prediction is performed off-line (at loading time). As a result, in the presented approach, querying can be very fast.

3 The Semantic Web Data Model

3.1 RDF: A Data Model for the SW

The recommended data model for the SW is the Resource Description Framework (RDF). It has been developed to represent information about resources on the WWW (e.g., meta data/annotations) where a resource stands for a thing that can be uniquely identified via a uniform resource identifier, URI. The basic statement is a triple of the form *(subject, property, property value)* or, equivalently, *(subject, predicate, object)*. A triple can graphically be described as a directed arc, labeled by the property (predicate) and pointing from the subject node to the property value node. A complete database (triple store) can then be displayed as a directed graph.

RDF Schema (RDFS) and various dialects of OWL (ontology web language) can be used to encode semantic constraints. Concepts and simple relationships between concepts are defined in RDFS, while OWL ontologies build on RDF/RDFS and add expressiveness. More details on SW standards can be found in [14, 15].

3.2 The Query Language SPARQL

SPARQL is a new standard for querying RDF-specific information and for displaying querying results. In its basic function a SPARQL query searches for graph patterns but it also contains the ability to formulate more expressive query patterns, to apply filters and to format the output. A small SPARQL query might contain a `PREFIX` statement for specifying the name space, a `SELECT` statement that determines the output pattern (typically a table of variable bindings) and a `WHERE` statement that specifies the searchable graph pattern and might contain variables. More complex queries are possible via grouping, optional patterns and alternative patterns. Filters can be used to further restrict the search pattern. Filters might include numerical comparisons ($<, >, =$), special operators, boolean operators, and arithmetic operations. The output format can be modified via `CONSTRUCT`, `DESCRIBE` and `ASK`. With `CONSTRUCT` the output can be formatted as an RDF document. `MODIFY` can be used to manipulate the output pattern. The keywords `ORDER BY`, `DISTINCT` can be used to reduce redundancy in the result set.

3.3 Inferred Closure

One way of making querying more powerful is to include in SPARQL not only statements explicitly stated in the data base but also statements that can be derived via reasoning and a number of tools provide that option. Inferred closure is defined as follows: it consists of the extension of a SW-KB with all the implicit statements, that could be inferred from it, using the enforced semantics [16]. In a strategy called materialization, after each update to the SW-KB made, the repository assures that the inferred closure is computed or updated and made available for query evaluation or retrieval. As a reasoning strategy, total materialization is adapted in a number of the popular SW repositories, including some of the standard configurations of Sesame and Jena (http://jena.sourceforge.net/). In the next section, we describe probabilistic materialization, i.e., the materialization of statements weighted by their estimated probabilities. In the following we will assume that *logical* materialization has been performed prior to learning such that the statements that can logically be inferred are available for learning.

4 Machine Learning for the SW

There have been a number of publications on learning with SW-data, e.g., [17–21]. The focus here is on machine learning approaches that permit the derivation of probabilistic statements.

4.1 Global Probabilistic Models

There are a number of approaches for learning in relational domains, in which a global probabilistic model in form of a probabilistic graphical model is learned, e.g., [3–6]. The state of a probabilistic node in these models corresponds to the truth value of the corresponding atomic statement.[4] Formally, let $X^{(s,p,o)} = 1$ stand for the fact that the statement (s, p, o) is true and let $X^{(s,p,o)} = 0$, otherwise. The most natural quantity that could be defined as a statement probability is

$$P(X^{(s,p,o)} = 1|\text{SW-KB}),$$

which is the marginal probability of $X^{(s,p,o)}$ given the information in the SW-KB. This can be decomposed as

$$P(X^{(s,p,o)}|\text{SW-KB}) = \sum_{\{X^U\}} P(X^{(s,p,o)}, \{X^U\}|\text{SW-KB})$$

where $\{X^U\}$ stands for the set of all statements whose truth value are unknown. Certainly, simplifications can be applied such that this sum can (approximately) be calculated for relatively large networks(e.g., [4]) but it needs to be shown that web-size scalability is feasible. A great advantage here is that this approach has no problems with missing information, i.e., can handle arbitrary patterns of missing information.

[4] A probabilistic node is simply the graphical representation of a random variable, representing in our case the truth value of a basic statement or triple. Not to be confused with a node in an RDF-graph.

4.2 Conditional Models

A second family of approaches includes the traditional approaches from ILP [7, 8, 22, 9, 10] but also a number of related statistical approaches [2, 23]. Typically a classification problem is stated. For example, the task might be to assign an entity to an ontological class or the task might be to predict a particular property of an entity (high income). Here we assume that the target class corresponds to a node $X^{(s,p,o)}$. Learning consists of the generation of relational features that are good predictors for the target class. For example, one might be able to predict income from the number of rooms in a person's house or from the income of the person's friends. The features are calculated from nodes in a neighborhood of the entity of interest. The nodes, that render the target class independent of the remaining probabilistic nodes form the Markov blanket $\mathrm{MB}^{(s,p,o)}$ such that

$$P(X^{(s,p,o)})|\text{SW-KB}) \approx P(X^{(s,p,o)}|\mathrm{MB}^{(s,p,o)}).$$

In the situations we are considering, this approach is difficult to apply due to the large number of statements with unknown truth values. ILP solves the problem of unknown truth values by simply making a closed-world assumption (thus there are no missing truth values in the Markov blanket), which is not appropriate in the context of the SW.[5] Due to the closed-world assumption, data points derived from the Markov blanket models are independent and the number of instances in the training set is under the control of the user, thus scalability is guaranteed.

4.3 Learning with Statistical Units Node Sets

In conclusion, a global model can more easily deal with missing information but might not scale well and conditional models scale better but have problems with missing information. We thus propose a model that attempts to combine the advantages of both approaches by being able to handle missing data and by being scalable. In addition, the approach can deal well with sparse data. In the next section we discuss suitable algorithms. Here we discuss how the appropriate data matrix is generated.

To define an appropriate statistical setting, we require the user to define statistical units and a population. Statistical units are the entities (e.g., persons) that are the source of the variables or features of interest. A population is the set of statistical units, 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.

Based on the definition of a statistical unit and a population, the statistical unit node set (SUNS) is defined. Let $U = \{u\}$ be the set of statistical units in the sample under consideration. In a first definition, we define a statistical node set

[5] A discussion on open-world and closed-world reasoning for the SW can be found in [24].

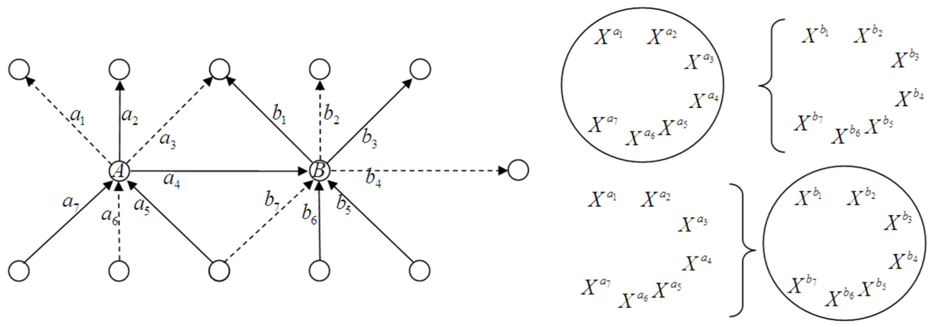
SUNS$_u$ for statistical unit u to include all probabilistic nodes that correspond to all actual and potential statements, in which u is either subject or object. We apply the restriction we have to apply is that if there are triples between the statistical units of the form (u_i, p, u_j) with $u_i, u_j \in U$ then $X^{(u_i,p,u_j)}$ is a member of SUNS$_{u_i}$ but not of SUNS$_{u_j}$. Otherwise the same probabilistic node would appear in two different SUNS, which would make the two SUNS highly dependent.

1. Let $U = \{u\}$ be the set of statistical units in the sample under consideration. The data matrix contains one row per statistical unit. Let (p, o) be a pair, such that a triple of the form triple (u, p, o) is in the SW-KB, for at least one $u \in U$. For each distinct (p, o), we generate a column in the data matrix. The entry in the data matrix for statistical unit u and pair (p, o) is equal to one, if the triple (u, p, o) is in the SW-KB and is zero otherwise.
2. In addition, we generate a column for each distinct p. The entry in the data matrix for statistical unit u and property p is equal to one if the triple (u, p, o) exists for at least one o in the SW-KB and is zero otherwise.
3. Let (s, p) be a pair, such that a triple of the form triple (s, p, u) is in the SW-KB, for at least one $u \in U$. For each distinct (s, p), we generate a column in the data matrix. The entry in the data matrix for statistical unit u_i and pair (s, p) is equal to one, if the triple (s, p, u) is in the SW-KB and is zero otherwise.
4. In addition, we generate a column for each distinct p. The entry in the data matrix for statistical unit u and property p is equal to one if the triple (s, p, u) exists for at least one s in the SW-KB and is zero otherwise.

As a postprocessing step we remove columns for which the number of ones is smaller than a threshold t. If there are triples between the statistical units of the form (u_i, p, u_j) with $u_i, u_j \in U$, we remove the columns for u_j where a statistical unit u_j acts as object. Thus a particular statement only appears once in the data matrix.[6] The approach can be used to estimate statements for the SUNS in the data matrix (transduction), but can also be applied to statistical units and their SUNS in the population (induction). The learned probabilistic statements can be stored in the SW-KB as weighted triples using a number of approaches, e.g., using reification.

In many cases it is desirable to include information outside a SUNS. For example, the wealth of a person can often be predicted by the wealth of a person's friends. This information can easily be added to the data matrix (in form of additional columns). But in the learning process, this information is treated as fixed input (covariate) information, i.e., the SW outside of a SUNS is treated as closed world (see Figure 1). Note, that the generation of the data matrix does not require explicit knowledge about the ontology since the matrix entries are calculated directly based on the statements in the SW-KB.

[6] This is not the only possible way to generate a data matrix, but an important feature is that only probabilistic nodes within a SUNS are evaluated.



a_1
a_2
a_3
a_4
a_5
a_6
a_7
A
B
b_1
b_2
b_3
b_4
b_5
b_6
b_7
X^{a_1} X^{a_2} X^{a_3} X^{a_4} X^{a_5} X^{a_6} X^{a_7}
X^{b_1} X^{b_2} X^{b_3} X^{b_4} X^{b_5} X^{b_6} X^{b_7}

Fig. 1. Left: An RDF-graph fragment with two statistical units A and B. $\{a_i\}$ are triples assigned to A and $\{b_i\}$ are triples assigned to B. Dashed lines indicate triples that are not in the SW-KB. Right top: The probabilistic nodes in the circle form the SUNS for statistical unit A and are modeled jointly. Information from triples not in a SUNS (here, the states of the probabilistic nodes $\{X^{b_i}\}$) are considered as inputs. Right bottom: The probabilistic nodes in the circle form the SUNS for statistical unit B and are modeled jointly. Information from triples not in a SUNS (here, the states of the probabilistic nodes $\{X^{a_i}\}$) are considered as inputs.

4.4 Algorithms for Learning with Statistical Units Node Sets

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 (EV), e.g., [13], matrix completion based on non-negative matrix factorization (NNMF) [25] and matrix completion using latent Dirichlet allocation (LDA) [26]. All three approaches estimate unknown matrix entries via a low-rank matrix approximation. EV 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 statistical units in the population outside the sample.

5 Workflow

The anticipated workflow is that, first, the statistical units and the population are defined via a configuration file. Then the probabilistic nodes in the SUNS are selected automatically and the data matrix is generated (see Section 4) based on a sample of the population of appropriate size. The sample size will be dependent

```
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
```

Fig. 2. Left: A SPARQL query. Right: A SPARQL query that includes probabilistic information.

on the training time that can be tolerated in the application. In an off-line learning process (as in materialization) the probabilities for the existence of SUNS statements, which are not known to be true, are estimated based on a matrix completion procedure. After training, the model is applied to all SUNS in the population.

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 [1] consortium meeting. First consider a regular SPARQL query that *finds all actors that act in movies that are filmed in an Italian city* (Figure 2, Left). 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 2, Right). Note that we have added the construct `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.

A query can also be based on several probability values. Note that we can answer the query: *select a patient who has a high probability of diabetes and has a high probability of hepatitis*, but not: *select a patient with high probability of diabetes and hepatitis*, since the latter would require joint probabilistic information, which is not stored.

6 Experiments

6.1 Data Set and Set Up

Data Set: The experiments are based on friend-of-a-friend (FOAF) data. The purpose of the FOAF project [27] is to create a web of machine-readable pages describing people, their relationships, and people's activities and interests, using W3C's RDF technology. The FOAF ontology is based on RDFS/OWL and is formally specified in the FOAF Vocabulary Specification.

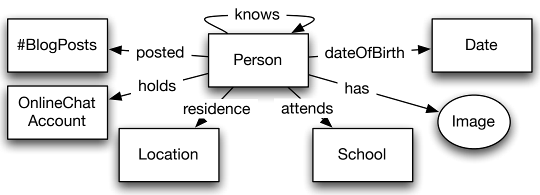


Fig. 3. Relational model of the LJ-FOAF domain

The FOAF dataset was generated from user profiles of the community website LiveJournal.com[7]. Figure 3 shows a summary of the triple-statements we are using in the experiments. 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. 462 of the 491 columns (friendship attributes) refer to the property *knows* (see Figure 3). 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) [28] to evaluate a predicted ranking, which is calculated by summing over all the gains along the rank list R with a log discount factor as $\text{NDCG}(R) = Z \sum_k (2^{r(k)} - 1/\log(1+k)$, where $r(k)$ denote the target label for the k-th ranked item in R, and Z is chosen such that a perfect ranking obtains value 1. 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. The better an algorithm, the higher would the friendship test statement be ranked.

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 SVM out off the LibSVM[29] package. In training, the one-class SVM only needs positive examples, which is quite appropriate for an open-world assumption. We tried three different kernels: a linear kernel, the gaussian RBF-kernel and a polynomial kernel. Two different input feature sets were examined: one 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*). For each friendship attribute, a separate SVM was trained.

[7] http://www.livejournal.com/bots/

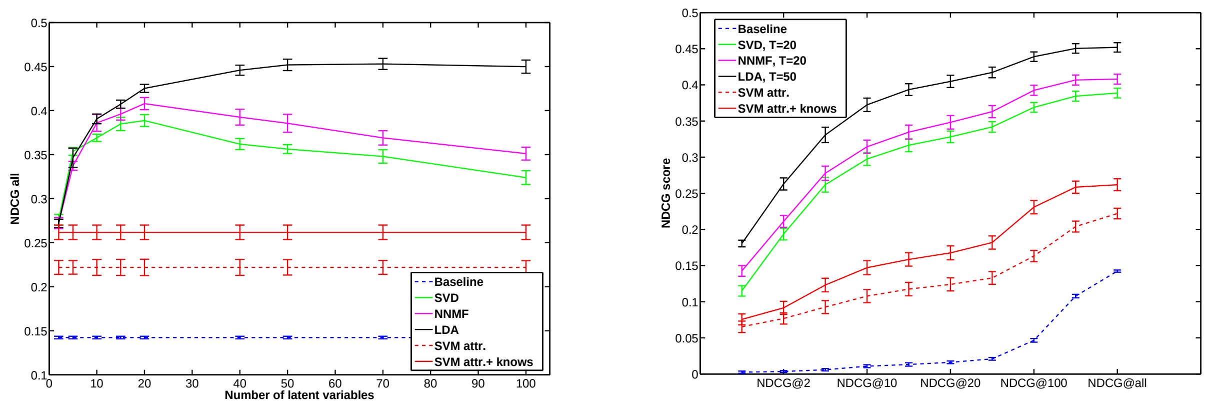


Fig. 4. NDCG comparison between different algorithms. Left: *NDCG all* is plotted against the number of latent variables. Right: *NDCG@n* score for different thresholds.

6.2 Results

Figure 4 shows the results for our FOAF data set. Figure 4 (left) plots the *NDCG all* score of all algorithms against the number of latent variables. Note that for the SVM, the best results were obtained with an RBF kernel ($\nu, \gamma = 0.01$). The error bars show the 95% confidence intervals based on the standard error of the mean. 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 4 (right) 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.

7 Conclusions 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 approach is suitable for a typical SW data situation with sparse data and missing data. The learning process is based on the concept of a statistical unit node set (SUNS) and is to a large degree autonomous. Only the statistical unit and the population need to be defined by a user. Since the size of a SUNS is rather independent of the overall size of the SW and since the sample size can be controlled, SUNS training time 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. The approach is most suitable when all statistical units of interest are in the training data set (transduction) or if the number of statistical units outside of the training data set is comparable to the size of the training data set. If the size of the population becomes very large, some relational information cannot be explored (e.g., random people in the world do not have friends in common). Future work will address this issue.

In our experiments based on the FOAF data set, LDA showed best performance, which we attribute to the fact that LDA, in contrast to NNMF and EV, uses a Bayesian approach, which has a smaller tendency to overfitting. Thus LDA can be a default method being insensible to exact parameter tuning. As confirmed by our experiments, support vector machines did not exhibit competitive performance in learning high-dimensional relational data. We demonstrated how probabilistic statements can be integrated into extended SPARQL queries. As example, based on the learning results for the FOAF data, one could answer queries such as: *Who would likely want to be Jack's friend; which female persons in the north-east US, would likely want to be Jack's friends.*

The approach can be extended in many ways. One might want to allow the user to specify additional parameters in the learning process, if desired, along the line of the extensions described in [12]. Another extension concerns ontological background knowledge. So far, ontological background knowledge was considered by including logically inferred statements into learning. A great advantage of the approach is that ontological knowledge is not required for the generation of the data matrix since the latter is generated based on observed SW triples. Ongoing work explores additional ways of exploiting ontological background information, e.g., for structuring the learning matrix. Similarly, we did not yet address the problem of ontology mapping and of having identical entities represented on the SW under different identifiers.

Acknowledgements: We acknowledge funding by the German Federal Ministry of Economy and Technology (BMWi) under the THESEUS project and by the EU FP 7 Large-Scale Integrating Project LarKC.

References

1. LarKC: The large Knowledge Collider. EU FP 7 Large-Scale Integrating Project, http://www.larkc.eu/. (2008)
2. Tresp, V., Bundschus, M., Rettinger, A., Huang, Y.: Towards Machine Learning on the Semantic Web. In: Uncertainty Reasoning for the Semantic Web I. Lecture Notes in AI, Springer (2008)
3. Getoor, L., Friedman, N., Koller, D., Pferrer, A., Taskar, B.: Probabilistic relational models. In Getoor, L., Taskar, B., eds.: Introduction to Statistical Relational Learning. MIT Press (2007)
4. Domingos, P., Richardson, M.: Markov logic: A unifying framework for statistical relational learning. In Getoor, L., Taskar, B., eds.: Introduction to Statistical Relational Learning. MIT Press (2007)

5. Xu, Z., Tresp, V., Yu, K., Kriegel, H.P.: Infinite hidden relational models. In: Uncertainty in Artificial Intelligence (UAI). (2006)
6. Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: AAAI. (2006)
7. Quinlan, J.R.: Learning logical definitions from relations. Machine Learning **5**(3) (1990)
8. Muggleton, S., Feng, C.: Efficient induction of logic programs. In: Proceedings of the 1st Conference on Algorithmic Learning Theory, Ohmsma, Tokyo (1990)
9. De Raedt, L.: Attribute-value learning versus inductive logic programming: The missing links (extended abstract). In: ILP '98: Proceedings of the 8th International Workshop on Inductive Logic Programming, Springer-Verlag (1998)
10. Lavrač, N., Džeroski, S., Grobelnik, M.: Learning nonrecursive definitions of relations with LINUS. In: EWSL-91: Proceedings of the European working session on learning on Machine learning. (1991)
11. Raedt, L.D., Jaeger, M., Lee, S.D., Mannila, H.: A theory of inductive query answering. In: ICDM. (2002)
12. Kiefer, C., Bernstein, A., Locher, A.: Adding data mining support to sparql via statistical relational learning methods. In: ESWC 2008, Springer-Verlag (2008)
13. Lippert, C., Weber, S.H., Huang, Y., Tresp, V., Schubert, M., Kriegel, H.P.: Relation-prediction in multi-relational domains using matrix-factorization. In: NIPS Workshop: Structured Input - Structured Output. (2008)
14. Antoniou, G., van Harmelen, F.: A Semantic Web Primer. The MIT Press (2004)
15. Hitzler, P., Krötzsch, M., Rudolph, S., Sure, Y.: Semantic Web. Springer (2008)
16. Kiryakov, A.: Measurable targets for scalable reasoning. Ontotext Technology White Paper (2007)
17. Berendt, B., Hotho, A., Stumme, G.: Towards semantic web mining. In: ISWC. (2002)
18. Grobelnik, M., Mladenic, D.: Automated knowledge discovery in advanced knowledge management. Library Hi Tech News: Online and CD Notes **9**(5) (2005)
19. Staab, S., Hotho, A.: Machine learning and the semantic web. In: ICML 2005 tutorial. (2005)
20. d'Amato, C.: Similarity-based Learning Methods for the Semantic Web. PhD thesis, University of Bari (2007)
21. Lisi, F.A.: The challenges of the semantic web to machine learning and data mining. In: Tutorial at ECML 2006. (2006)
22. Kramer, S., Lavrac, N., Flach, P.: From propositional to relational data mining. In Džeroski, S., Lavrac, L., eds.: Relational Data Mining. Springer-Verlag (2001)
23. Popescul, A., Ungar, L.H.: Feature generation and selection in multi-relational statistical learning. In Getoor, L., Taskar, B., eds.: Introduction to Statistical Relational Learning. MIT Press (2007)
24. Grimm, S., Motik, B.: Closed world reasoning in the semantic web through epistemic operators. In: OWLED. (2005)
25. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature (1999)
26. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3** (2003)
27. Brickley, D., Miller, L.: The Friend of a Friend (FOAF) project, http://www.foaf-project.org/
28. Jarvelin, K., Kekalainen, J.: IR evaluation methods for retrieving highly relevant documents. In: SIGIR'00. (2000)
29. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. (2001)