



LarKC

*The Large Knowledge Collider
a platform for large scale integrated reasoning and Web-search*

FP7 – 215535

D3.2

Description of Strategy and Design for Machine Learning Approaches

Volker Tresp, Yi Huang

| | |
|-----------------------------|--------------------------|
| Document Identifier: | LarKC/2008/D3.2 |
| Class Deliverable: | LarKC EU-IST-2008-215535 |
| Version: | version 1.0 |
| Date: | December 31, 2009 |
| State: | final |
| Distribution: | public |



EXECUTIVE SUMMARY

In the deliverable we describe strategy and design for the machine learning approach in LarKC. Similar to some logical reasoners that materialize triple stores by performing deductive inference, the proposed machine learning approach attempts to estimate the truth value (likelihood) of statements by exploring regularities in the data and by performing inductive inference. Therefore, the approach can be viewed as the transformation of a data set of facts to a probabilistic data set of both certain facts and probabilistic facts. The approach deals well with the challenging properties of the Semantic Web, i.e., its heterogeneity, the sparsity of relationships and missing information. In addition, we define an extension of the SPARQL query language for querying the probabilistic data base. The proposed approach is capable to handle very large data sets and is to a great extent autonomous. The goal is that the machine learning module can be used by people who are not machine learning experts.



DOCUMENT INFORMATION

| | | | |
|---------------------------|---|----------------|-------|
| IST Project Number | FP7 – 215535 | Acronym | LarKC |
| Full Title | Large Knowledge Collider | | |
| Project URL | http://www.larkc.eu/ | | |
| Document URL | | | |
| EU Project Officer | Stefano Bertolo | | |

| | | | | |
|---------------------|---------------|-----|--------------|--|
| Deliverable | Number | 3.2 | Title | Description of Strategy and Design for Machine Learning Approaches |
| Work Package | Number | 3 | Title | Abstraction and Learning |

| | | | | |
|----------------------------|---|-----|---|----------------|
| Date of Delivery | Contractual | M21 | Actual | 31-December-09 |
| Status | version 1.0 | | final <input checked="" type="checkbox"/> | |
| Nature | prototype <input type="checkbox"/> report <input type="checkbox"/> dissemination <input type="checkbox"/> | | | |
| Dissemination Level | public <input checked="" type="checkbox"/> consortium <input type="checkbox"/> | | | |

| | | | | |
|--------------------------|------------------------|---------|---------------|--------------------------|
| Authors (Partner) | Volker Tresp, Yi Huang | | | |
| Resp. Author | Volker Tresp | | E-mail | volker.tresp@siemens.com |
| | Partner | Siemens | Phone | +49 (89) 636-49408 |

| | |
|-------------------------------------|--|
| Abstract (for dissemination) | In the deliverable we describe strategy and design for the machine learning approach in LarKC. Similar to some logical reasoners that materialize triple stores by performing deductive inference, the proposed machine learning approach attempts to estimate the truth value (likelihood) of statements by exploring regularities in the data and by performing inductive inference. Therefore, the approach can be viewed as the transformation of a data set of facts to a probabilistic data set of both certain facts and probabilistic facts. The approach deals well with the challenging properties of the Semantic Web, i.e., its heterogeneity, the sparsity of relationships and missing information. In addition, we define an extension of the SPARQL query language for querying the probabilistic data base. The proposed approach is capable to handle very large data sets and is to a great extent autonomous. The goal is that the machine learning module can be used by people who are not machine learning experts. |
| Keywords | Statistical Machine Learning, Inductive Inference, Multivariate Prediction, Linked Data, Relational Graph Model |

PROJECT CONSORTIUM INFORMATION













| Acronym | Partner | Contact |
|--|---|---|
| Semantic Technology Institute Innsbruck http://www.sti-innsbruck.at |  | Prof. Dr. Dieter Fensel Semantic Technology Institute (STI) Innsbruck, Austria E-mail: dieter.fensel@sti-innsbruck.at |
| AstraZeneca AB http://www.astrazeneca.com/ |  | Bosse Andersson AstraZeneca Lund, Sweden E-mail: bo.h.andersson@astrazeneca.com |
| CEFRIEL SCRL. http://www.cefriel.it/ |  | Emanuele Della Valle CEFRIEL SCRL. Milano, Italy E-mail: emanuele.dellavalle@cefriel.it |
| CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O. http://cyceurope.com/ |  | Dr. Michael Witbrock CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O., Ljubljana, Slovenia E-mail: witbrock@cyc.com |
| Hchstleistungsrechenzentrum, Universitaet Stuttgart http://www.hlrs.de/ |  | Georgina Gallizo Hchstleistungsrechenzentrum, Universitaet Stuttgart Stuttgart, Germany E-mail : gallizo@hlrs.de |
| Max-Planck-Institut fur Bildungsforschung http://www.mpib-berlin.mpg.de/index_js.en.htm |  | Dr. Lael Schooler, Max-Planck-Institut fr Bildungsforschung Berlin, Germany E-mail: schooler@mpib-berlin.mpg.de |
| Ontotext Lab, Sirma Group Corp. http://www.ontotext.com/ |  | Atanas Kiryakov, Ontotext Lab, Sirma Group Corp. Sofia, Bulgaria E-mail: atanas.kiryakov@sirma.bg |
| SALT LUX INC. http://www.saltlux.com/EN/main.asp |  | Kono Kim SALT LUX INC Seoul, Korea E-mail: kono@saltlux.com |
| SIEMENS AKTIENGESELLSCHAFT http://www.siemens.de/ |  | Dr. Volker Tresp SIEMENS AKTIENGESELLSCHAFT Muenchen, Germany E-mail: volker.tresp@siemens.com |
| THE UNIVERSITY OF SHEFFIELD http://www.shef.ac.uk/ |  | Prof. Dr. Hamish Cunningham THE UNIVERSITY OF SHEFFIELD Sheffield, UK E-mail: h.cunningham@dcs.shef.ac.uk |
| VRIJE UNIVERSITEIT AMSTERDAM http://www.vu.nl/ |  | Prof. Dr. Frank van Harmelen VRIJE UNIVERSITEIT AMSTERDAM Amsterdam, Netherlands E-mail: Frank.van.Harmelen@cs.vu.nl |
| THE INTERNATIONAL WIC INSTITUTE, BEIJING UNIVERSITY OF TECHNOLOGY http://www.iwici.org/ |  | Prof. Dr. Ning Zhong THE INTERNATIONAL WIC INSTITUTE Mabeshi, Japan E-mail: zhong@maebashi-it.ac.jp |
| INTERNATIONAL AGENCY FOR RESEARCH ON CANCER http://www.iarc.fr/ |  | Dr. Paul Brennan INTERNATIONAL AGENCY FOR RESEARCH ON CANCER Lyon, France E-mail: brennan@iarc.fr |



TABLE OF CONTENTS

| | | |
|-----|---|---|
| 1 | THE DELIVERABLE REPORT CONSISTING OF TWO PAPERS | 1 |
| 2 | DISCUSSION OF THE APPROACH IN CONTEXT OF THE WP3 TASKS | 2 |
| 2.1 | Why the SUNS Approach Is Preferred | 2 |
| 2.2 | Discussion in Terms of Task 3.1: Training Set Retrieval | 3 |
| 2.3 | Discussion in Terms of Task 3.2: Active Learning and Dealing with Incomplete Data | 3 |
| 2.4 | Discussion in Terms of Task 3.3 Abstraction / Predicate Invention / Feature Generation | 4 |
| 2.5 | Discussion in Terms of Task 3.4 Optimize Learner and Ontology Integration | 5 |
| 3 | DESIGN OF THE LEARNING APPROACH | 6 |
| 3.1 | Architecture and Components | 6 |
| 3.2 | Transform Plug-ins | 6 |

1 THE DELIVERABLE REPORT CONSISTING OF TWO PAPERS

The aim of the LarKC project is to go beyond the limited storage, querying and inference technology currently available for semantic computing, taking into account, at the same time, the scalability challenge of the Semantic Web. Whereas logical reasoning can exploit deterministic constraints in a domain, machine learning is able to exploit statistical patterns in a domain. In accordance with the general goals of LarKC, machine learning needs to be scalable and the predicted information should be accessible by a query language. This is exactly the goal of the effort in WP 3.

In the deliverable we describe the design and strategy of Machine Learning approaches in LarKC. The deliverable consists of two papers:

- The paper “*Materializing and Querying Learned Knowledge*” was presented at the *ESWC Workshop on Inductive Reasoning and Machine Learning on the Semantic Web 2009*.

The paper proposes a learning approach which achieves inductive inference by exploring regularities in the data. The approach is suitable for the challenging properties of the data on the Semantic Web and can easily be used by people who are not Machine Learning experts. The learnt knowledge can be materialized and jointly queried together with facts and logically inferred statements. The paper discusses problems with some current approaches to learning on the Semantic Web. The proposed approach is based on probabilistic nodes whose states reflect the truth values of statements. The probabilistic nodes are partitioned into *statistical unit node sets* (SUNS). Within a statistical unit node set, joint probabilistic models are estimated, while information outside an SUNS can be exploited as covariates. We study learning algorithms from multivariate prediction, which we utilize to estimate the truth values of statements not yet present in the data.

- The paper “*Multivariate Prediction for Learning in Relational Graphs*” was presented at the NIPS 2009 Workshop *on Analyzing Networks and Learning with Graphs*.

In the paper we extend the SUNS approach to be able to deal with data in relational formats, since currently the majority of data yet resides in relational databases. Inspired by the RDF graph, we define a graphical representation, which can represent relations with arbitrary arity. The discussion in the paper clarifies how blank nodes should be addressed in SUNS learning on RDF-graphs.

In both papers we present our preliminary experimental results using a FOAF data set gathered from a social network site.

2 DISCUSSION OF THE APPROACH IN CONTEXT OF THE WP3 TASKS

2.1 Why the SUNS Approach Is Preferred

In a similar way as logical reasoning can supplement triples not explicitly present in the data, machine learning, viewed as inductive reasoning, can generate probabilistic triples that can be integrated into a triple store or can be used to predict triples of interest online (at runtime / at querying time). Thus a scalable machine learning solution fits well into the overall framework of the LarKC project.

The LarKC project concerns reasoning and machine learning in a relational RDF graph. As described in the proposal, such a graph is typically heterogeneous with different entity types and different relationship types. In general, information might be missing. A learning approach needs to be able to deal with these properties and constraints and should scale with the size of the Web. Moreover, it is required that machine learning should easily be applicable by non-experts. As discussed in the two papers, the SUNS approach has the potential to meet all of these requirements.

Let's consider some alternatives. Initially we experimented with the infinite hidden relation model (IHRM) ([5] in the first paper). Although the model is quite appealing, convergence of the learning algorithm was often too slow to be considered in LarKC. In addition, the IHRM belongs to a family of approaches that attempt to form a joint probabilistic model of a domain, where, in principle, all random variables are coupled. This necessarily leads to limited scalability. This comment also applies, maybe to a lesser degree, to the Markov Logic Networks (MLNs) approaches ([4] in the first paper). Another feature of MLNs is that they rely on a prior definition of rich logical constraints or features, which might often pose a difficulty for the user.

The SUNS approach deals well with high dimensionality, high sparsity and missing information. In a much less challenging case where only attributes (and no relationships) are derived as features and the values of those features are available for all instances, alternative approaches, like neural networks and support vector machines, might have advantages. The SUNS approach would then (with a small modification) behave like a log-linear model, resp. a linear regression model.

In conclusion, one cannot expect that any machine learning approach outperforms all other approaches but the SUNS approach seems to be applicable in more challenging domains. If it will turn out that a large number of applications would require an alternative algorithm this would not pose a serious problem. The most important component of the SUNS approach is the construction of a data matrix that could be the basis for alternative learning algorithms as well.

Currently, at the heart of the SUNS approach is a multivariate modeling problem which we solve via singular value decomposition (SVD) and latent Dirichlet allocation (LDA). It is likely that one of these two approaches will become the default approach. Again, if another multivariate modeling approach turns out to be favorable, we can easily use that approach as well.

As discussed in the papers, scalability is mainly reached via selection, in the simplest case via random sampling. Our experimental results have shown that in most cases, the approach works sufficiently well with a reasonable small sample and

including more data over a certain limit does not further improve the performance. If in some cases a unusual large training sample is required, various forms of parallelization are possible, typically in form of committee machines. Here the sample would be divided into non overlapping chunks, a multivariate model is learned independently for each chunk. The overall prediction is finally achieved by, for instance, an average over the estimation of each individual chunk.

We are in discussion with the Use Cases concerning the application of machine learning. As an orientation and for inspiration here is a list of applications where we feel that our approach could be effective:

- Prediction of user preferences
- Travel planning
- (Web-) Service recommendation for an application or for a user
- Prediction and analysis of social network patterns
- Entity resolution: are two entities in two domains identical?
- Protein function prediction
- Protein interaction prediction
- Recommendation of medical procedures
- Ranking of patient diagnosis
- Rating objects (texts, exams, pictures, ...)

2.2 Discussion in Terms of Task 3.1: Training Set Retrieval

In the SUNS approach, training set retrieval refers to the construction of the data matrix for the training process. The rules for the generation of the data matrix are described in the papers. Given the definition of the statistic unit and the population we sample a subset of statistic units (sampling strategies are described in the next section). The features derived from the triples that the statistical units are involved in form the multivariate response variables. Additionally, aggregated information is calculated from triples further away in the relational graph. The SUNS relies less on an involved feature selection procedure as, for example, inductive logic programming (ILP) approaches. Rather, we believe that all derived features have some predictive power and should be included. This is in accordance with the experience gained in other high dimensional relational domains as in document modeling.

2.3 Discussion in Terms of Task 3.2: Active Learning and Dealing with Incomplete Data

In this section we present several sampling strategies that are being pursued in the SUNS approach. If the size of the population (i.e., the total number of all statistical

units) can be handled by the learning algorithm, no sampling is necessary. In contrast, if the number of the statistical units in the population is too large to be efficiently handled by the learning algorithm (for the training), a subset need to be sampled.

- The random sampling is in many ways the best option since statistical inference is well defined for a random sample.
- We have also implemented link-following sampling, very similar to the standard crawling of web pages. The advantage here is that statistical units do not need to be catalogued a priori for random sampling and only a subgraph of the whole RDF graph of a given domain needs to be accessed. Therefore the sampling is usually more efficient. A possible disadvantage is that statistical inference might be more difficult, when the training data and the test data have different statistical distributions.
- As another option we will implement an active learning strategy. If only a small set of the population can be used for training, for example due to the high cost of labeling, it makes sense to select the training set carefully. We plan to implement the active learning algorithm described in the paper [1], since it can be applied both for labeled and unlabeled data.

Missing data are handled as follows. In the SUNS approach we distinguish two cases. In one case, a triple is known to be true. Such a triple will result in a *one* entry in the data matrix. In the other case, a triple has an unknown truth value or is known to be untrue. Such a triple will produce a *zero* entry in the data matrix. We decided for this pragmatic choice since in many domains it is not clear how a missing triple should be treated (as evidence that the triple is untrue or as missing information). If some application domains require missing data mechanism e.g., if the domain makes a clear distinction between missing information and untrue triples, then a proper missing data mechanism can be included. In fact matrix factorization can be extended to handle missing information ([13] in the first paper).

2.4 Discussion in Terms of Task 3.3 Abstraction / Predicate Invention / Feature Generation

As mentioned in Section 2.2 the SUNS approach dose not aspire to carefully select an optimal subset of best features, since in high dimensional statistical domains, feature selection has not proven to be effective. On the other hand, additional invented features can improve the predictive performance of the learnt models. There are two situations where predicates / features are invented in the SUNS approach. In the first case we register the aggregated features forming the inputs or covariates. In the second case the latent representation formed by SVD and LDA can be considered as invented features. Both the aggregated features and the active dimensions / topics can be assigned to a statistical unit as additional features for use in additional learning tasks and as predicates for logical reasoning.

2.5 Discussion in Terms of Task 3.4 Optimize Learner and Ontology Integration

Ontological knowledge can be integrated in two ways. First, logical reasoning can be performed prior to learning such that triples inferred by logical reasoning are available for the data matrix. Secondly, the subclass hierarchy can easily be integrated as demonstrated in the publication.

The optimization of the learner will be based on the experience gained in application domains. We will explore additional multivariate models, as the one described in reference [13] in the first publication. That approach might lead to better results, if strong attribute information for both objects in an RDF statement is available. Other promising candidates are multinomial mixture model, naïve Bayesian approaches, tree-augmented naïve Bayes, and other decision tree oriented algorithms.

3 DESIGN OF THE LEARNING APPROACH

In this chapter we briefly describe the architecture of the SUNS learning approach, its components and the transform plug-ins under development.

3.1 Architecture and Components

Based on the workflow of the SUNS approach (see Section 5 in the first paper) we designed the architecture of the approach. Figure 3.1 shows an overview of the architecture. The core of the architecture is the learning engine. The learning engine can directly be integrated into the LarKC platform as plug-in or can be provided as a web service so that any other web service and application can utilize its functionalities via HTTP. To facilitate the usability of the learning engine, a default configuration will be available in form of an XML/RDF file. Naturally, the user can overwrite the default setting. We now introduce the main components of the learning engine:

- *Manager*: loads the configuration file and coordinates the communication among the components and the communication between the learning engine and external frameworks, tools and libraries.
- *View Generator*: processes the sampling of the statistical units according to predefined strategies.
- *Feature Creator*: derives features from the triples of the sampled statistical units and performs feature pruning by removing those features that occur very rarely.
- *Query Generator*: builds SPARQL queries used for sampling and feature extraction.
- *Learning Libraries*: are machine learning libraries implemented within the SUNS. They are mainly multivariate prediction approaches, active learning methods and committee machines.
- *Approach Selector*: is responsible for selecting the underlying approach from the learning libraries.
- *Data Convertor*: converts the matrix into a particular data format required by the chosen learning approach, e.g., CSV (comma-separated values) or binary format.

As shown in the Figure 3.1 the learning engine contains also two APIs: One accesses triple stores over a Semantic Web framework, e.g., Jena, and the other calls external (remote) machine learning libraries, if necessary.

3.2 Transform Plug-ins

Currently we are working on the development of two transform plug-ins.

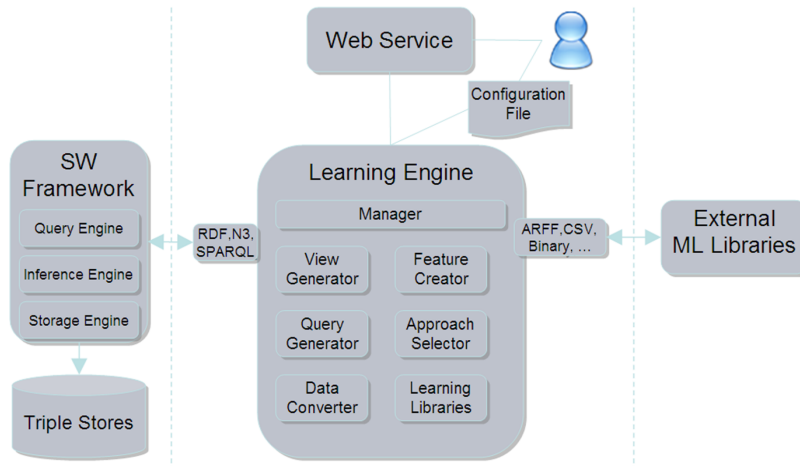


Figure 3.1: Overview of the architecture

- The *RDF2Matrix* plug-in constructs the data matrix by transforming a data set available in RDF format into a matrix. Rows stand for instances of a statistical units and columns represent their features derived from the associated RDF graph. The binary entries *one* and *zero* reflect the truth values of the corresponding triples true and unknown, respectively. For example, if rows are genomes and columns are diseases, a *one* of the (i,j) entry in a genetic-variation-relationship matrix indicates that the i -th gene influences the j -th disease; a *zero* entry indicates that it is unknown whether a relationship exists between the particular gene and the particular disease.
- The *Inductive Materialization* plug-in performs inductive inference by using machine learning and materializes the learnt knowledge. The plug-in estimates probabilities of the truth value of triples not present yet in the triple store and persistently saves those probabilistic triple in some manner, ideally in quads. Thereby they can be then retrieved by (extended) SPARQL queries. The plug-in can be viewed as the transformation of a conventional triple store into a probabilistic triple store.

Obviously the latter is based on the former, while the *RDF2Matrix* plug-in can be included in any LarkC pipeline which requires the data matrix as input.

REFERENCES

- [1] Kai Yu, Jinbo Bi, and Volker Tresp. Active learning via transductive experimental design. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, New York, NY, USA, 2006.

Materializing and Querying Learned Knowledge

Volker Tresp¹, Yi Huang¹, Markus Bundschuh², and Achim Rettinger³

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 organized 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 $MB^{(s,p,o)}$ such that

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

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.

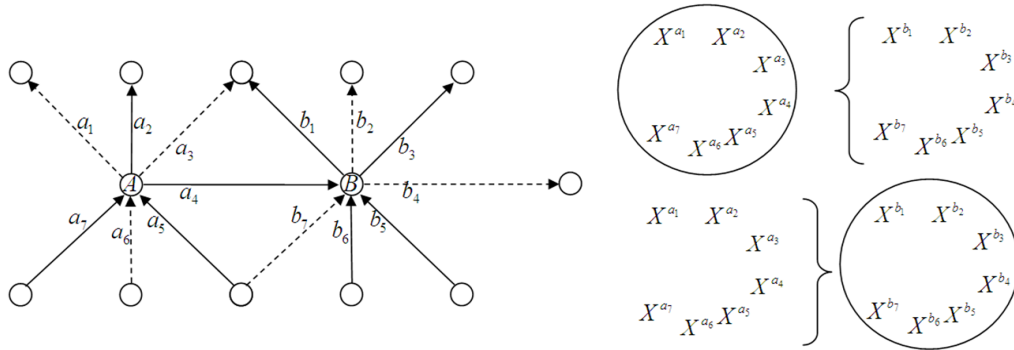


Fig. 1. Left: An RDF-graph fragment with two statistical units A and B . $\{a_i\}$ are triples assigned to A and $\{b_i\}$ are triples assigned to B . Dashed lines indicate triples that are not in the SW-KB. Right top: The probabilistic nodes in the circle form the SUNS for statistical unit A and are 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

| | |
|--|---|
| <pre> 1 PREFIX ex: http://example.org/ 2 SELECT ?actor 3 WHERE 4 { ?movie ex:filmedIn ?city . 5 ?city ex:inCountry ex:Italy . 6 ?actor ex:actIn ?movie 7 }</pre> | <pre> 1 PREFIX ex: http://example.org/ 2 SELECT ?actor 3 WHERE 4 { ?movie ex:filmedIn ?city . 5 ?city ex:inCountry ex:Italy . 6 ?actor ex:actIn ?movie 7 WITH PROB ?prob 8 } 9 ORDER BY ?prob</pre> |
|--|---|

Table 1. 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 1, 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 1, 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.

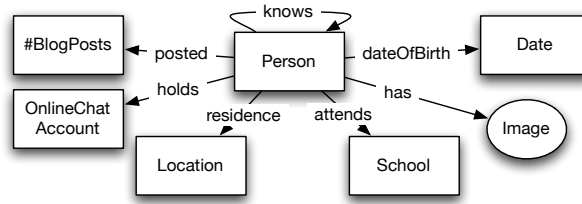


Fig. 2. Relational model of the LJ-FOAF domain

The FOAF dataset was generated from user profiles of the community website LiveJournal.com⁷. Figure 2 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 2). 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 $NDCG(R) = Z \sum_k (2^{r(k)} - 1) / \log(1 + k)$, 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/>

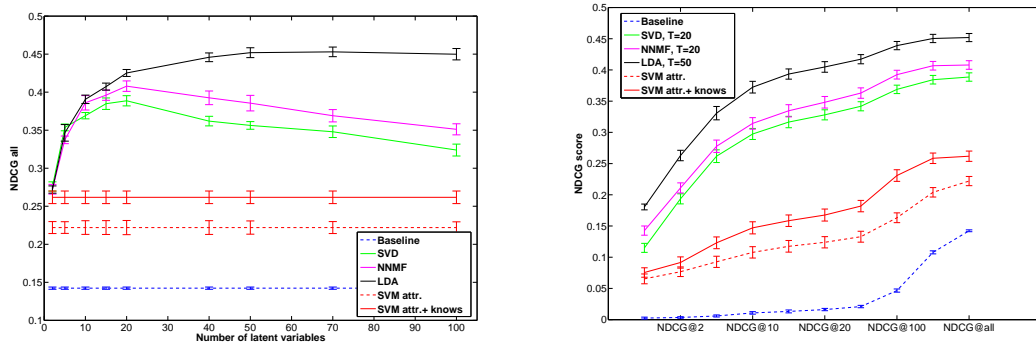


Fig. 3. NDCG comparison between different algorithms. Left: $NDCG_{all}$ is plotted against the number of latent variables. Right: $NDCG@n$ score for different thresholds.

6.2 Results

Figure 3 shows the results for our FOAF data set. Figure 3 (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 NMF 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 its 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 3 (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 insensitive 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 (BMW) 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., Bundschuh, 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., Krieger, 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: Proceedings of the National Conference on Artificial Intelligence (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. De Raedt, L., 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., Huang, Y., Weber, S.H., Tresp, V., Schubert, M., Kriegel, H.P.: Relation prediction in multi-relational domains using matrix factorization. Technical report, Siemens (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, Springer-Verlag (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)

Multivariate Prediction for Learning in Relational Graphs

Yi Huang and Volker Tresp

Siemens AG, Corporate Technology
Otto-Hahn-Ring 6, 81739 München, Germany
YiHuang{Volker.Tresp}@siemens.com

Hans-Peter Kriegel

Institute for Computer Science
Ludwig-Maximilians-Universität München
Oettingenstr. 67, 80538 München, Germany
kriegel@dbs.ifi.lmu.de

Abstract

This paper concerns learning with data in relational formats. In focus are simplicity, scalability and ease of use. Inspired by the graphical representation used in context of the RDF data model of the Semantic Web, we propose a graphical representation for the data in relational data bases. Based on this relational graphical model, we define a statistical learning framework and derive a data matrix on which machine learning is applicable. The data matrix typically contains a large number of coupled random variables (outputs) and a large number of covariates (inputs). We discuss learning algorithms from multivariate prediction, which we utilize to estimate the truth values of tuples not yet present in the data base. Learned tuples and their certainty values can be stored in the data base and can be integrated in querying. We present experimental results using data from a social network side.

1 Introduction

Relational learning is an area of increasing interest mainly due to the growing availability of linked data, e.g., in the form of hyperlinked World Wide Web documents, of citation data, of social network data and of various networks in the life sciences. More specifically, the term Linked Data is used in context of the Semantic Web (SW)¹ as the effort to link and jointly explore several structured data sources on the Web². A prominent example is the LinkedLifeData project³, in which a subset of the more than 1000 publicly available life science data bases can jointly be queried. Linked Data become increasingly interesting as an application area for machine learning. As the name implies, to a great extent Linked Data describe relations between objects and thus relational learning should immediately be applicable. Relational learning must face the challenges raised by the typical data situation of Linked Data. First, data are heterogeneous with many different entity types and relationships, often arranged in a hierarchical ontology. Second, relations are often extremely sparse, e.g., only a tiny subset of all possible persons are someone's friends. Third, e.g., for privacy reasons, information is missing. In addition we require machine learning to be highly scalable, which excludes many theoretically interesting relational learning approaches. We also want machine learning to be easily configurable since it is often applied in an explorative matter and utilized by people who are not experts in machine learning. Within the EU FP7 project LarKC [9], a machine learning approach called SUNS (Statistical Unit Node Set) is being developed that attempts to conquer all those challenges by learning on the SW [17]. In this paper we extend the SUNS approach to be applicable in the context of relational data bases, mainly for two reasons. First, although the SW might provide

¹<http://www.w3.org/2001/sw/>

²<http://linkeddata.org/>

³<http://www.linkedlifedata.com/>

the data models of the future, currently the majority of data resides in relational data bases. Second, the RDF (Resource Description Framework)⁴ directly represents only binary relations; to represent higher order relations, awkward, so called blank nodes need to be introduced. No such restriction exists for data in a relational format.

A great advantage of the RDF data model is the simple semantics transported by its graphical representation: a complete data base can be represented as a directed graph where nodes stand for resources (e.g., objects) and links denote relations between resources. In this paper we introduce a similarly simple and intuitive graphical representation for a relational data base (RDB) containing relations with arbitrary arity. We assume the data base has been transformed into the fifth normal form (5NF). We map the scalable SUNS approach to this representation by defining a statistical setting, in which statistical inference is well defined. We derive a data matrix to which multivariate modeling is applicable and estimate the truth values of tuples not yet present in the data base. Learned tuples and their certainty values can be stored in the data base and can then be integrated in querying.

The paper is organized as follows. In the next section, we discuss related work. In Section 3 we introduce our graphical representation of a relational data base. In Section 4 we define the statistical setting, define the random variables in the domain of interest, and derive the data matrix. In Section 5 we present experimental results. Our conclusions are presented in Section 6.

2 Related Work

The approach is an extension of the work presented in [17] for learning with the RDF data model. Here we use a relational representation, which is able to encode relations with arity larger than two. Inductive logic programming (ILP) has traditionally been applied to relational domains [15, 13]. The difference is that classical ILP is concerned with deterministic or close to deterministic dependencies whereas here we are concerned with statistical learning. Also, ILP requires a complex feature selection step, which we avoid. Propositionalization [3, 10] is an ILP approach, which is closer to our view. We see advantages in our approach due to its simplicity, generality, ease of use and scalability. Also in contrast to typical approaches in propositionalization, we apply multivariate prediction, which allows us to predict jointly a large number of tuples in one step. Multivariate prediction has been shown to give superior performance in predicting relationships between objects (e.g., between users and movies). Learning frameworks in relational data bases are also being pursued in industry. For example, Microsoft's Data Mining Extension (DMX) defines a general approach for learning in relational data bases. A difference is that we are mostly interested in predicting relationships between objects by using multivariate models, whereas most common frameworks focus on standard data mining tasks such as clustering and classification. The work in SRL (Statistical Relation Learning) [5, 4, 19, 7, 14] is certainly also closely related and the main advantage of our approach is that we do not require an involved structural search procedure, as, for example, PRMs (probabilistic relational models) [5]. In contrast to MLNs (Markov Logic Networks) [16], we do not need as a basis rules or other logical constraints defined by a domain expert.

3 A Graphical Representation of a Relational Data Base

The data format of the SW is the RDF-graph, in which resources (i.e., objects) are linked by directed arcs describing simple subject-predicate-object statements. Since the graphical representation in form of the RDF-graph has proven to be very useful for the SW, we introduce now a graphical representation of a relational data base (RDB), i.e., a relational graph. The graphical representation is convenient for the derivation of the data matrix but, naturally, the approach is independent of this graphical representation. Whereas an RDF-graph directly presents only binary relationships, the relational graph can represent relations with arbitrary arity. An advantage of our representation is that random variables are explicitly represented as nodes, which is neither true in the RDF-graph nor in many other relational graphical representations.

⁴<http://www.w3.org/RDF/>

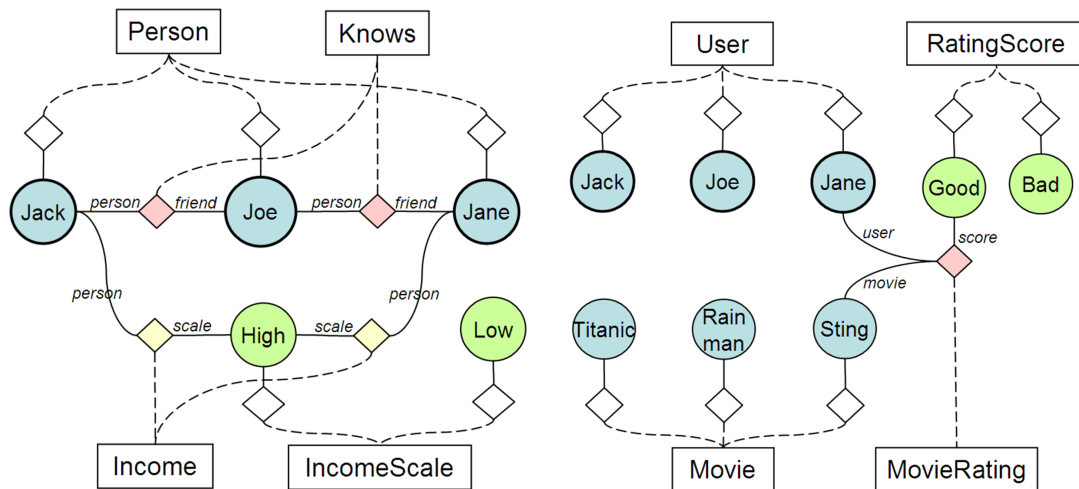


Figure 1: Relational Graphs. Constants are represented by circular nodes, tuples by diamond-shaped nodes and relations by rectangular nodes. Nodes representing statistical units (in both cases: *Persons*) have a darker rim. The dashed lines indicate which tuple belongs to which relation. Constants are linked to the tuples they appear in and the links are labeled by the attribute names. The left graph shows a social friendship network in which the income of a person is an attribute. The right side shows a movie rating domain.

For simplicity of exposure we make the following assumptions, which can partially be relaxed. First, we assume that each object has a UID, i.e., a unique identifier. Secondly, we assume that the underlying RDB is minimal in the sense that the arity of a relation cannot be reduced without changing its semantics. An example: the relation *MovieRating*(*Jane*, *Sting*, *Good*) cannot be reduced. In contrast, *PersonAttributes*(*Jane*, *Blond*, *Tall*) can be reduced to *HairColor*(*Jane*, *Blond*) and *PersonHeight*(*Jane*, *Tall*) without loss of information. The reason is that higher order relationships are more difficult to handle for machine learning and should be avoided if possible. In the theory of data base normalization, this corresponds to the fifth normal form (5NF) [8]. Finally we make a distinction between constants that have object character (resources in SW terms, e.g., persons) and constants that have attribute character (literals in SW terms, e.g., numerical values, Good, Bad, etc.).

We illustrate the graphical relational graph using as an example a social network in Figure 1 (left). The constants in the data base are represented as circular nodes with object nodes *Jack*, *Joe* and *Jane* (blue) and attribute nodes *Low* and *High* (green). Next we introduce a rectangular node for each relation *Person*, *Knows*, *IncomeScale*, and *Income*. For each tuple in the RDB a diamond-shaped node is introduced. A tuple node is connected via a dashed line to the associated relation and is linked to the constants in the tuple via solid lines. We draw unary tuples in white, tuples between object constants in purple and tuples between object constants and attribute constants in yellow. Finally, the links between the tuples and the objects are labeled by the corresponding attribute (column) name of the relation. The tuple nodes for unary relations, i.e., *Person*, *IncomeScale* are somewhat redundant but they are useful later when we introduce random variables. Given objects, relations and attributes, the random quantities are the tuples; thus we consider that tuples, which are not known to be true, exist with some probability.

4 Statistical Modeling

4.1 Defining the Sample

In a relational domain the issue of a probability distribution is somewhat tricky. Consider a set of physicians and patients, let Q be the quality of a physician, and assume that 50% of the physicians are good and 50% of the physicians are bad. If we sample physicians, we might conclude that $P(Q = \text{Good}) = P(Q = \text{Bad}) = 1/2$. On the other side patients might have a preference to visit good physicians. Thus, if we sample patients, we might obtain that $P(Q = \text{Good}) = 0.7$,

i.e., that 70% of the physicians of the patients are good. Thereby one can conclude that in relational domains (as in any other domains), the interpretation of a probability distribution is determined by the sampling process. More precisely, we must be careful in defining the statistical unit, the population, the sampling procedure and the features. A statistical unit is an object of a certain type, e.g., a person. In our framework, a statistical unit might be an object in a unary relation, e.g., in the relation *Person*. The population is the set of statistical units under consideration. In our framework, a population might be defined as the set of persons that attend a particular university. For learning we use a subset of the population, typically by random sampling. Based on the sample, a data matrix is generated where the statistical units in the sample define the rows.

4.2 The Random Variables in the Data Matrix

We now introduce the state of a tuple node. A tuple node is in state *one* (*true*) if the tuple is known to exist and is in state *zero* (*false*) if the tuple is known not to exist. Graphically, one only draws the tuple nodes in state one, i.e., the existing tuples in the RDB. In RDBs one typically makes a closed-world assumption such that the tuples that are not represented in the RDB are assumed false. Here we assume that the truth values of the remaining tuples are unknown. This is in accordance with the view, for example, in MLNs [16].

We now associate some tuples with statistical units. In connection with the random sampling process described earlier tuple nodes become random variables with states *one* and *zero*. The idea is now to assign a tuple to a statistical unit if the statistical unit appears in the tuple. Let's consider the statistical unit *Jane* (Figure 1, left). Based on the tuples she is participating in we obtain the expressions *Person(X)*, *Knows(Joe, X)*, *Income(X, High)*, where *X* is a variable that represents a statistical unit. The expressions form the random variables (outputs) and define columns in the data matrix. By considering the remaining statistical units *Jack* and *Joe* generate the expressions (columns) *Knows(X, Jane)*, *Knows(X, Joe)*, *Knows(Jack, X)*. We will not add *Knows(Jane, X)* since Jane considers no one in the data base to be her friend. We iterate this procedure for all statistical units in the sample and add new expressions (i.e., columns in the data matrix), if necessary. Note, that expressions that are not represented in the sample will not be considered. Also, expressions that are rarely true (i.e., for few statistical units) will be removed since no meaningful statistics can be derived from a small sample.

An example for a ternary relation is shown in Figure 1 (right). Here, based on *Jane's* tuples the expression *MovieRating(X, Sting, Good)* is added; it represents statistical units that like the movie *The Sting*.

In [17] the tuples associated with a statistical unit were denoted as statistical unit node set or SUNS.

4.3 Non-random Covariates in the Data Matrix

The columns we have derived so far represent tuples that belong to the schema. Those tuples are treated as random variables in the analysis. If machine learning predicts that a tuple is very likely, we can enter this tuple in the RDB (see Section 6). We now add columns that provide additional information for machine learning but which we treat as covariates or fixed inputs.

First, we derive simplified relations from the RDB. More precisely, we consider the expressions derived in the last subsection and replace constants by variables. For example, from *Knows(X, Jane)* we derive *Knows(X, Y)* and count how often this expression is true for a statistical unit *X*, i.e. we count the number of friends of a statistical unit *X*. From the ternary tuple *MovieRating(X, Sting, Good)* we obtain *MovieRating(X, Y, Good)*, *MovieRating(X, Sting, Y)*, and *MovieRating(X, Y, Z)*. By counting the occurrences we obtain the number of movies a statistical unit has rated as *Good*, how often a statistical unit has rated *The Sting*, and how often a statistical unit has rated any movie.

Second, we consider two simple types of aggregated features from outside a SUNS. Consider first a binary tuple *Knows(X, Jane)*. If Jane is part of another binary tuple, in the example, *Income(Jane, High)*, then we form the expression *Knows(X, Y) ∧ Income(Y, High)* and count how many rich friends a statistical unit has. Consider a ternary tuple *MovieRating(X, Sting, Good)* with two objects and one attribute constant. Consider an additional binary tuple of the form *Starring(Sting, PaulNewman)*. Then we form *MovieRating(X, Y, Good) ∧ LeadingActor(Y, PaulNewman)* and count how many

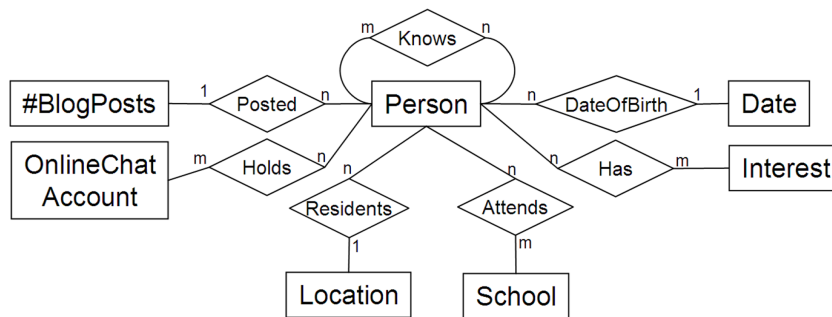


Figure 2: Entity-relationship diagram of the LJ-FOAF domain

Paul Newman movies a statistical unit X likes. A large number of additional aggregated features are possible but so far we restricted ourselves to these two types.

After construction of the data matrix we prune away columns which have ones in fewer than ϵ percent of all rows or in more than $(1 - \epsilon)$ of all rows, where ϵ is usually a very small number. Thus we remove aggregates features that are very rarely true or almost always true, since for those no meaningful statistical analysis is possible. Note that by this pruning procedure we have reduced the exponential number of random variables to typically a much smaller set.

4.4 Algorithms for Learning with Statistical Units Node Sets

A row in the resulting data matrix contains external inputs based on aggregated information (if available) and typically a large number of binary and sparse outputs. A *one* stands for a tuple known to be true and a *zero* for a tuple whose truth value is unknown. In this situation, multivariate structured prediction approaches have been most successful [18]. In multivariate structured prediction all outputs are jointly predicted such that statistical strength can be shared between outputs. The reason is that some or all model parameters are sensitive to all outputs, improving the estimates of those parameters. The approaches we are employing here are based on a matrix completion of the complete data matrix, including inputs and outputs.⁵ We investigate matrix completion based on a singular value decomposition (SVD), e.g., [12], matrix completion based on non-negative matrix factorization (NNMF) [11] and matrix completion using latent Dirichlet allocation (LDA) [1]. 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 of the *zero* entries in the data matrix, the entries are interpreted as certainty values that the corresponding tuples are true. After training, the models can be applied to statistical units in the population outside the sample.

5 Experiments

5.1 Data Set and Experimental Setup

The experiments are based on a friend-of-a-friend (FOAF) data set. The purpose of the FOAF project [2] is to create a web of machine-readable pages describing people, their relationships, and people’s activities and interests.

We gathered our FOAF data set from user profiles of the community website LiveJournal.com⁶. The data set will be made available online. All extracted entities and relations are shown in Figure 2. In total we collected 32,062 persons and all related attributes. We selected 14,425 persons with “dense” friendship relationships. On average, a given person has 27 friends.

⁵ Although the completion is applied to the complete matrix, only *zeros* —representing tuples with unknown truth values— are overwritten.

⁶<http://www.livejournal.com/bots/>

Table 1: Best *NDCG all*: knows relation with threshold 10

| Setting | Method | Threshold of attr. / aggr. attr. | | | |
|--------------|--------|----------------------------------|-----------------|-----------------|-----------------|
| | | 0/0 | 10/50 | 20/200 | inf/inf |
| inductive | SVD | 0.3460 ± 0.0044 | 0.3211 ± 0.0040 | 0.3330 ± 0.0038 | 0.3155 ± 0.0041 |
| | LDA | 0.2804 ± 0.0022 | 0.2807 ± 0.0024 | 0.2967 ± 0.0023 | 0.3137 ± 0.0020 |
| transductive | SVD | 0.3446 ± 0.0092 | 0.3297 ± 0.0084 | 0.3349 ± 0.0094 | 0.3201 ± 0.0085 |
| | LDA | 0.3312 ± 0.0079 | 0.3265 ± 0.0078 | 0.3289 ± 0.0072 | 0.3393 ± 0.0073 |

 Table 2: Best *NDCG all*: knows relation with threshold 20

| Setting | Method | Threshold of attr. / aggr. attr. | | | |
|--------------|--------|----------------------------------|-----------------|-----------------|-----------------|
| | | 0/0 | 10/50 | 20/200 | inf/inf |
| inductive | SVD | 0.4811 ± 0.0060 | 0.4657 ± 0.0058 | 0.4557 ± 0.0059 | 0.4424 ± 0.0061 |
| | LDA | 0.3809 ± 0.0047 | 0.4025 ± 0.0046 | 0.3742 ± 0.0046 | 0.4332 ± 0.0044 |
| transductive | SVD | 0.4703 ± 0.0073 | 0.4460 ± 0.0066 | 0.4453 ± 0.0065 | 0.4201 ± 0.0070 |
| | LDA | 0.4801 ± 0.0055 | 0.4790 ± 0.0056 | 0.4465 ± 0.0053 | 0.4705 ± 0.0052 |

The task is to predict potential friends of a person, i.e., *Knows* tuples. For each person in the data set, we randomly selected one *Knows* friendship tuple and set the corresponding matrix entry to *zero*, to be treated as unknown (test tuple). Also we consider the modification in the aggregated features. In the test phase we then predicted all unknown friendship entries, including the entry for the test tuple. The test tuple should obtain a high likelihood value, if compared to the other unknown friendship entries.

Aggregated features here describe the aggregated properties of persons to which a *Knows* relation exist. For instance, an aggregated property is whether a person knows anybody who attends a certain school.

Here we use the normalized discounted cumulative gain (NDCG) [6] 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 $NDCG(R) = Z \sum_k (2^{r(k)} - 1) / \log(1 + k)$, where $r(k)$ denotes the target label for the k -th ranked item in R , and Z is chosen such that a perfect ranking obtains value 1. The better an algorithm, the higher would the friendship test tuple be ranked.

5.2 Results

In the experiment, we analyzed to what degree friendship variables (i.e., the relation *Knows*) can be predicted from the friendship patterns (the patterns in the friendship columns) alone and to what degree person’s attributes and aggregated attributes improve the predictive performance. Also we were interested to see to what degree the setting of the thresholds ϵ in the pruning steps affected the results. All experiments were done with 2,000 persons in the training set and 2000 persons in the test set. All experimental results were repeated with 5 non-overlapping sets to obtain confidence values.

The Table 1 shows results when the threshold for the *Knows* relation was set to 10, i.e., columns of friends with less than 10 entries were removed. The columns in the table are labeled by the threshold chosen for the attributes (first number) and the aggregated attributes (second number). We chose different numbers for attributes and aggregated attributes since there were many fewer aggregated features than attributes. Table 2 shows results where the threshold for the *Knows* relations was set to 20. We only report results for LDA and for SVD; the results for NNMF were consistently worse than the results for those two algorithms. The number of topics / principle components was chosen to be optimally: about 100 in all cases.

The first thing to observe is that the second experiment resulted in better scores. This can be explained by the fact that there is stronger correlation between friendship patterns of popular friends. As can be seen in the tables, attributes and aggregated attributes improved the score significantly

for SVD. For SVD, a *zero* threshold for both attributes and aggregates features gave overall best results (i.e., no pruning). Interestingly, when all attributes and aggregated attributes were included, the results for the inductive setting were slightly better than the results for the transductive setting. LDA could only beat SVD when no attributes were used (threshold infinity). Interestingly, the performance of LDA was not improved in general when attributes or aggregated attributes were used.

6 Discussion and Conclusions

We have presented a learning approach for relational data. The learning process is based on a large degree of autonomy. Mainly the statistical unit and the population need to be defined by a user. Since the number of columns in the data matrix is, to a first approximation, independent of the overall size of the RDB and since the sample size can be controlled, training time is essentially independent of the overall size of the RDB. The generalization from the sample to the population is linear in the size of the population.

In our experiments based on the FOAF data set, SVD showed most consistent performance. Both LDA and SVD exploited the benefits of multivariate prediction since approaches based on single predictions (not reported here) did not even reach the performance of very simple benchmark approaches.

The tuples with their certainty values can be entered into the RDB. Although the number of estimated tuples is much smaller than the number of possible tuples, in many cases one might want to apply an additional selection process, e.g., only store the N tuples in a relation with highest probability. The derived probabilistic tuples can readily be integrated into SQL queries. For example, based on the learnt results using the FOAF data set, 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 relies on the fact that, in the relational graph, information is local to a statistical unit. Let's assume that any descendant of Rockefeller is rich. If the data base only contains the relation *ChildOf*, the presented approach could not directly learn that rule. On the other hand, the father or mother of a descendant of Rockefeller would also be rich (at least one of them is a descendant of Rockefeller as well). Also a descendant of Rockefeller might typically have rich friends. The presented approach would be able to exploit this information and conclude that a rich parent and rich friends would be an indicator of wealth.

For an analysis of statistical inference one can distinguish three different scenarios. First, we consider a transductive setting, where we only consider statistical units in the sample and predict the truth values of their tuples. This corresponds to inference used in many recommendation systems (e.g., the Netflix competition). Second, we generalize the learnt model to statistical units in the population (i.e., the RDB) that are outside the sample and not observed during the training. Considering the design of the experiment where we generated the sample as a random subset of statistical units in the data base, this type of inference is sound. Third, we consider the generalization to statistical units which are not in the RDB. This type of generalization is of course the most interesting one but also the most difficult one to analyze. This latter type of generalization is an area of active research in network analysis.

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] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3, 2003.
- [2] D. Brickley and L. Miller. *The Friend of a Friend (FOAF) project*. <http://www.foaf-project.org/>.
- [3] L. De Raedt. 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.

-
- [4] P. Domingos and M. Richardson. Markov logic: A unifying framework for statistical relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.
 - [5] L. Getoor, N. Friedman, D. Koller, A. Pferrer, and B. Taskar. Probabilistic relational models. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.
 - [6] K. Jarvelin and J. Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *SIGIR'00*, 2000.
 - [7] C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2006.
 - [8] W. Kent. A simple guide to five normal forms in relational database theory. *Communications of the ACM*, 26, 1983.
 - [9] LarKC. *The large Knowledge Collider*. EU FP 7 Large-Scale Integrating Project, <http://www.larkc.eu/>, 2008.
 - [10] N. Lavrač, S. Džeroski, and M. Grobelnik. Learning nonrecursive definitions of relations with LINUS. In *EWSL-91: Proceedings of the European working session on learning on Machine learning*, 1991.
 - [11] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 1999.
 - [12] C. Lippert, S. H. Weber, Y. Huang, V. Tresp, M. Schubert, and H.-P. Kriegel. Relation-prediction in multi-relational domains using matrix-factorization. In *NIPS 2008 Workshop: Structured Input - Structured Output*, 2008.
 - [13] S. Muggleton and C. Feng. Efficient induction of logic programs. In *Proceedings of the 1st Conference on Algorithmic Learning Theory*. Ohmsma, Tokyo, 1990.
 - [14] A. Popescul and L. H. Ungar. Feature generation and selection in multi-relational statistical learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.
 - [15] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3), 1990.
 - [16] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2), 2006.
 - [17] V. Tresp, Y. Huang, M. Bundschuh, and A. Rettinger. Materializing and querying learned knowledge. In *Proceedings of the First ESWC Workshop on Inductive Reasoning and Machine Learning on the Semantic Web*, 2009.
 - [18] V. Tresp and K. Yu. Learning with dependencies between several response variables. In *Tutorial at ICML 2009*, 2009.
 - [19] Z. Xu, V. Tresp, K. Yu, and H.-P. Kriegel. Infinite hidden relational models. In *Uncertainty in Artificial Intelligence (UAI)*, 2006.