Towards Machine Learning on the Semantic Web

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

[1] Siemens AG, Corporate Technology, Information and Communications
Learning Systems, Munich, Germany
[2] Ludwig-Maximilian University Munich, Germany
[3] Technical University of Munich, Germany

Abstract. In this paper we explore some of the opportunities and challenges for machine learning on the Semantic Web. The Semantic Web provides standardized formats for the representation of both data and ontological background knowledge. Semantic Web standards are used to describe meta data but also have great potential as a general data format for data communication and data integration. Within a broad range of possible applications machine learning will play an increasingly important role: Machine learning solutions have been developed to support the management of ontologies, for the semi-automatic annotation of unstructured data, and to integrate semantic information into web mining. Machine learning will increasingly be employed to analyze distributed data sources described in Semantic Web formats and to support approximate Semantic Web reasoning and querying. In this paper we discuss existing and future applications of machine learning on the Semantic Web with a strong focus on learning algorithms that are suitable for the relational character of the Semantic Web's data structure. We discuss some of the particular aspects of learning that we expect will be of relevance for the Semantic Web such as scalability, missing and contradicting data, and the potential to integrate ontological background knowledge. In addition we review some of the work on the learning of ontologies and on the population of ontologies, mostly in the context of textual data.

1 Introduction

The world wide web (WWW) represents an ever increasing source of information. Until now the WWW is mostly accessible to humans via search engines and browsers whereas computers only have a very rudimentary understanding of web content. The vision behind the Semantic Web (SW) is that computers should also be able to understand and exploit information offered on the web [1]. In the near future, a web representation might contain human-readable parts and sections made available in SW-formats to be accessible for automated processing. The SW is based on two concepts. First, a formal ontology provides domain specific background information that is shared by several parties: It provides a common vocabulary for a given domain and describes object classes, predicate classes and their interdependencies, as well as additional background information formalized in logical statements. Second, web information is annotated by statements readable and interpretable by machines via the common ontological background knowledge.

One of the prime SW applications will be context/user sensitive information retrieval where the result will still be in textual or multimedia format, to be interpreted by a human. But this information will be much more specific to the user's needs, since data can be integrated from multiple sites and smart information filters can be applied. Thus a search engine becomes more of an agent who knows the user, who has a deep understanding of the information request, who knows what to find where on the web and who presents the requested information in an appropriate user-friendly form. An immediate benefit from semantic annotation will be that annotated web pages might obtain a higher search rank since the match between query and page content can be evaluated with high confidence. In a second group of applications, the items to be searched for are not human readable texts or multimedia data but are machine readable information about an item or a web service. Semantic web services are of great interest both for academia and industry [2,3]. Service requests and service offerings can be formulated precisely based on SW standards and can be understood as precisely by semantic search engines and web applications. In the third family of applications the SW becomes the *web of data*. SW technologies will form the infrastructure for a standardized representation of information and for information exchange. Biomedicine is a forerunner here with almost 1000 databases publicly available today. If the data were published under a common SW ontological format, all this information would be accessible for querying and for analysis. As the WWW brought the knowledge of the world to our finger tips, the SW will bring the data of the world to our applications. Finally, in a fourth family of applications, SW technologies are being used in advanced expert systems to model complex industrial domains [4].

Reasoning plays an important role on the SW: Based on ontological background knowledge and the set of asserted statements, logical reasoning can derive new statements. But logical reasoning has its limitations. First, logical reasoning does not easily scale up to the size of the web as required by many applications; projects like the EU FP 7 Large-Scale Integrating Project LarKC are under way to address this issue [5]. Second, uncertain information is not suitable for logical reasoning. The representation of uncertain information on the SW and reasoning with uncertainty on the SW have only recently been addressed [6]. Third, logical reasoning is completely based on axiomatic prior knowledge and does not exploit regularities in the data that have not been formulated as ontological background knowledge. In contrast, and as it has been demonstrated in many application areas, successful solutions can often be achieved by induction, i.e., by learning from data. The analysis of the potential of machine learning for the SW is the topic of this contribution.

The most immediate application of machine learning is SW mining, enhancing traditional web mining applications. Web content mining, web structure mining, web usage mining and the learning of ranking functions for retrieval will all benefit from the additional information available on the SW [7]. In another group of applications, machine learning serves the SW by supporting ontology construction and management, ontology evaluation, ontology refinement, ontology evolution, as well as the mapping, merging and alignment of ontologies [8,9,10,11,12]. Mostly these tasks are addressed on the basis of unstructured or semi-structured textual data. After all, most current web pages contain textual information; but other types of input data will become increasingly important, as well [13]. Alternatively, researchers are concerned with learning of data already in SW formats. As already mentioned, the current trend is that an increasing amount of information is made available in SW formats and machine learning and data mining will be the basis for the analysis of the combined data sources. A particular aspect here is the learning of logical constraints that can then be formulated in the language of the employed ontology [14,15,16,17]. One can also contemplate that future ontologies should be extended to be able to represent learned information that cannot easily be formulated with current standards, e.g., represent the input-output mapping represented in probabilistic classifiers. The trained statistical models can then be used to estimate the probability that statements are true, which are neither explicitly asserted in the database nor can be proven to be true (or false) based on logical reasoning. Since the conclusions drawn from machine learning are typically probabilistic, this uncertainty needs to be represented [6,18,5]. Consequently, querying can include learned statements, e.g., : *Find all female persons that live in the southeastern US, are older than 21 years, own a house and are likely to own a sailboat* where the last information, i.e., the likelihood of owning a sailboat, was learned from data. Finally, in applications where the raw data is unstructured, machine learning can support the population of ontologies, i.e., the mapping of unstructured data to SW statements. Most work here concerns the population from textual data although the annotation of semi-structured data and multimedia data. e.g. images and video, is of great relevance as well. A goal here is to describe multimedia content semantically for fast content-based reasoning and retrieval.

In this paper we analyze algorithms from machine learning that are suitable for SW applications. First and foremost, SW data describe relationships between objects. Consequently, suitable learning approaches should be able to handel the relational character of the data. By far the majority of machine learning deals with non-relational feature-based representations (also referred to as propositional representation or attribute-value representation). Only recently statistical relational learning (SRL) is finding increasing interest in the ML community [19]. In Section 3 we present a novel discussion on feature-based learning in the SW and in Section 4 we relate this discussion to learning algorithms from inductive logic programming (ILP). In Section 5 we discuss matrix decomposition approaches and in Section 6 we present relational graphical models that are based on a joint probabilistic model of a relational domain. We discuss the machine learning approaches with respect to their applicability in a SW context, i.e., their scalability to the expected large size of the SW, their ability to integrate ontological background knowledge, their ability to handle the varying quality and reliability of data[1] and finally, their ability to deal with missing and contradictory data. In Section 7 we add a discussion on ontology learning and ontology population based on textual data. Ontology learning and ontology population are the most developed aspects of machine learning on the SW. In Section 8 we report first experiments based on the FOAF data set and in Section 9 we present conclusions. We will start the remaining part of the paper with an introduction into the SW as proposed by the W3C.

2 Components of the SW Languages

The World Wide Web Consortium (W3C) [21] is the main international standards organization for the WWW and develops recommendations for the SW. We will discuss here the main SW standards, i.e., RDF, RDFS and OWL [22,23]. RDF is useful for making statements about instances, RDFS defines schema and subclass hierarchies, and OWL can be used to formulate additional background knowledge. Very elegantly, the statements in RDF, RDFS and OWL can all be represented as one combined directed graph (Figure 1). A common semantics is based on the fact that some of the language components of RDFS and OWL have predefined domain-independent interpretations.

2.1 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), but might as well be used to describe other structured data, e.g., data from legacy systems. A resource stands for an object that can be uniquely identified via a uniform resource identifier, URI, which is sometimes referred to as a bar code for objects on the SW. The basic statement is a triple of the form *(subject, property, property value)* or, equivalently, *(subject, predicate, object)*. For example *(Eric, type, Person)*, *(Eric, fullName, Eric Miller)* indicates that Eric is of the concept (or class) *Person* and that Eric's full name is Eric Miller. 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. The subject of a statement is always a URI, the property value is either also a URI or a literal (e.g., String, Boolean, Float). In the first case, one denotes the property as object property and a statement as an object-to-object statement. In the latter case one speaks of a datatype property and of an object-to-literal statement. A complete database (triple store) can then be displayed as a directed graph, a semantic net (Figure 1). One might think of a triple as a tuple of a binary relation *property(subject, property values)*. A triple can only encode a binary relation involving the subject and the property value. Higher order relations are encoded using blank nodes. Consider the originally ternary relation *transaction(User, Item, Rating)*. The blank node might be

[1] Trust learning is an emerging field [20].

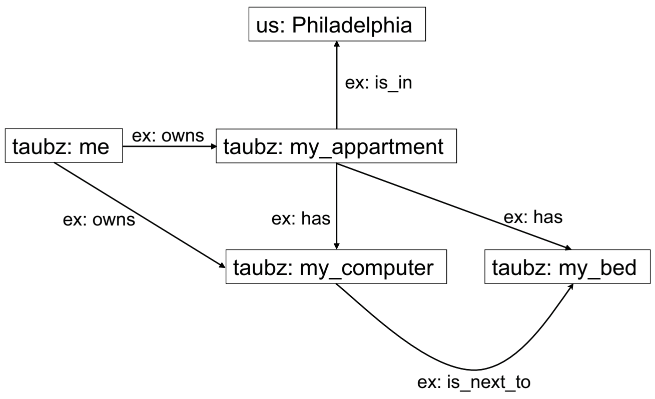

Fig. 1. An RDF-graph fragment. Redrawn from [24].

TransactionId with triples (binary relations): *(TransactionId, userRole, User)*, *(TransactionId, transactionObject, Item)* and *(TransactionId, evaluation, Rating)*. A blank node is treated as a regular resource with an identifier, only that it might be invisible from outside the file. Blank nodes are also helpful for defining containers such as bags (unordered container), sequences (ordered container) and collections (lists).

Each resource is associated with one or several concepts (i.e., classes) via the type-property. A concept can be interpreted as a property value in a type-of-statement. Conversely, one can think of a concept as representing all instances belonging to that concept. Concepts are defined in the RDF Vocabulary Description Language, also called RDF-Schema or RDFS. Both RDF and RDFS form a joint RDF/RDFS graph. In addition to defining all concepts, the RDFS also contains certain properties that have a predefined meaning, implementing specific constraints and entailment rules. First, there is the subclass property. If an instance is of type *Concept1* and *Concept1* is a subclass of *Concept2*, then the instance can be inferred to be also of type *Concept2*. Subclass relations are essential for generalization in reasoning and learning. Each property has a representation (node) in RDFS as well. A property can be a subproperty of another property. For example, the property *brotherOf* might be a subproperty of *relatedTo*. Thus if *A* is a brother of *B* one can infer that *A* is *relatedTo B*.

A property can have a domain respectively range constraint: *(marry, domain, Person)* and *(marry, range, Person)* states that if two resources are married then they must belong to the concept *Person*. Interestingly, RDF/RDFS statements cannot lead to contradictions in RDF/RDFS, one reason being that negation is missing. The same remains true for some less expressive ontologies.

2.2 Ontologies

Ontologies build on RDF/RDFS and add expressiveness. W3C developed standards for the web ontology language OWL, which comes in three dialects or profiles: the most expressive is OWL Full, which is a true superset of RDFS. A full inference procedure for OWL Full is not implementable with simple rule engines [23]. Some applications requiring OWL Full might build an application-specific reasoner instead of using a general one. OWL DL (description language) is included in OWL Full and OWL Lite is included in OWL DL. Both OWL DL and OWL Lite are decidable but are not true supersets of RDFS.

In OWL one can state that classes are equivalent or disjoint and that properties respectively instances are identical or different. The behavior of properties can be classified as being symmetric, transitive, functional or inverse functional, ... (e.g., teaches is the inverse of isTaughtby). In RDFS concepts are simply named. OWL allows the user to construct classes by enumerating their content (explicitly stating its members), through forming intersections, unions and complements of classes. Also classes can be defined via property restrictions. For example, the constraints that (1) first-year courses must be taught by professors, (2) mathematics courses are taught by David Billington, (3) all academic staff members must at least teach one undergraduate course, can all be expressed in OWL using the constructs allValuesFrom ($\forall$), hasValue, and someValuesfrom ($\exists$). Furthermore, cardinality constraints can be formulated (e.g., a course must be taught by someone, a department must have at least ten and at most 30 members) (Examples from [22]). Very attractive is that both instances and ontologies can be joined by simply joining the graphs: in fact the only real thing is the graph [23].

In some data rich applications ontologies will have no relevance beyond the definition of classes and properties. Conversely, in some domains, such as bioinformatics, medical informatics and some industrial applications [4], sophisticated ontologies have already been developed [23].

2.3 Reasoning

An ontology formulates logical statements, which can be used for analyzing data consistency and for deriving new implicit statements concerning instances and concepts. Total materialization denotes the calculation of all implicit triples at loading time, which might be preferred if query response time is critical [25]. Note, that total materialization is only feasible in some restricted ontologies.

2.4 Rules

RuleML (Rule Markup Language) is a rule language formulated in XML and is based on datalog, a function-free fragment of Horn clausal logic. RuleML allows the formulation of if-then-type rules. Both RuleML and OWL DL are different subsets of first-order logic (FOL). SWRL (Semantic Web Rule Language) is a proposal for a Semantic Web rules-language, combining sublanguages of OWL (OWL DL and Lite) with those of the Rule Markup Language (Unary/Binary Datalog). Datalog clauses are important for modeling background knowledge in cases where DL might be inappropriate, for example in many industrial applications.

2.5 Querying

The recommended RDF-query language for the SW is SPARQL (SPARQL Protocol and RDF Query Language). The SPARQL syntax is similar to SQL. A search pattern is a directed graph with variable nodes (i.e., a graph pattern). The result is is either in the form of a list of variable bindings or in the form of an RDF-graph.

3 Feature-Based Statistical Learning on the SW

3.1 Feature-Based Statistical Learning

Based on a long tradition, statistical learning has developed a large number of powerful analytical tools and it is highly desirable to make these tools available for the SW. Figure 2 (top) shows the main steps that are performed in statistical learning, analyzing, as example, students in a university. First, a *statistical unit* is defined, which is the entity that is the source of the variables or features of interest [26,27,28]. The goal is to generalize from observations on a few units to a statistical assembly of units. Typically a statistical unit is an object of a given type, here a student. In general one is not interested in all statistical units but only in a particular subset, i.e., the *population.* The population might be defined in various ways, for example it might concern all students in a particular country or, alternatively, all female students at a particular university.

In a statistical analysis only a subset of the population is available for investigation, i.e. a *sample.* Statistical inference is dependent on the details of the sampling process; the sampling process essentially defines the random experiment and, as a stationary process, allows the generalization from the sample to the population. In a *simple random sample* each unit is selected independently. Naturally, sometimes more complex sampling schemes are used, such as stratified random sampling, cluster sampling, and systematic sampling.

The quantities of interest of the statistical investigation are the *features* (or variables) that are derived from the statistical units. In the example, features are a student's IQ and a student's age. In the next step the *data matrix* is formed where each row corresponds to a statistical unit and each column corresponds to a feature. Finally, an appropriate statistical model is employed for modeling the data, i.e., the analysis of the features and the relationships between the features, and the final result is analyzed by the user. Naturally, all of this is typically an iterative process, e.g., based on a first analysis new features might be added and the statistical model might be modified.

In a supervised statistical analysis one partitions the features in explanatory variables (a.k.a. independent variables, predictor variables, regressors, controlled variables, input variables) and dependent variables (a.k.a response variables, the regressands, the responding variables, the explained variables, or the outcome/output variables). Note that it is often a design choice if one either defines a population based on the state of a variable or if one uses that variable as an independent variable. Consider a binary variable male/female. One choice

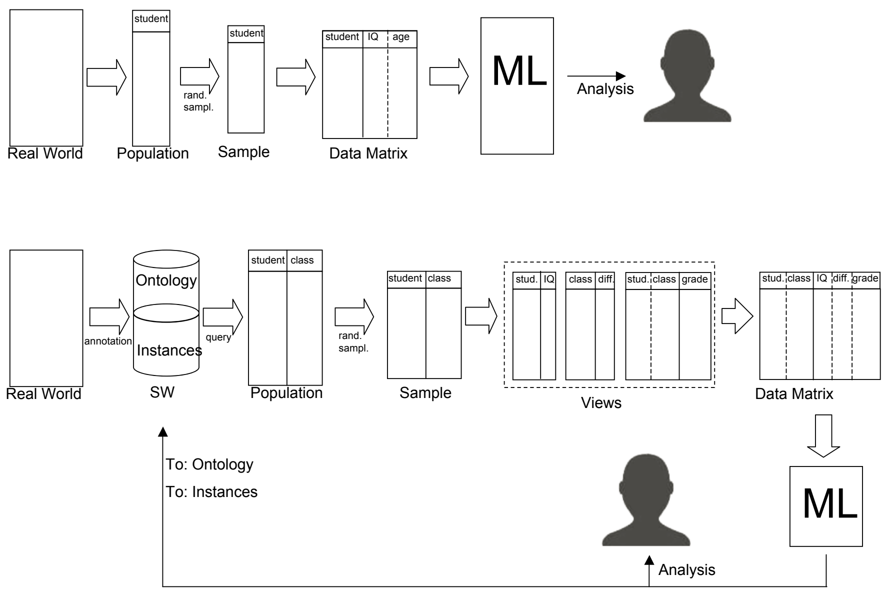

Fig. 2. Top: Standard machine learning. Bottom: Machine learning applied to the SW.

might be to partition the population into males and females and learn separate models for each population. Another option is to simply use gender as an independent variable and consider a joint population of males and females. The second choice is for example more appropriate if the sample is small. Hierarchical Bayesian modeling is a compromise in which statistical inference in different populations is coupled.

3.2 Feature-Based Statistical Learning on the SW

The main steps for statistical learning on the SW are displayed in Figure 2 (bottom). The first new aspect is that the statistical analysis is based on the world as it is represented on the SW and that all quantities of interest, i.e., statistical unit, population, sample and features, are defined in context of the SW.[2] As before, a *statistical unit* might be defined to be an object of a given type, e.g., a student. More generally a statistical unit might be composed of several objects that have a particular relationship to each other. In Figure 2, as an example, a statistical unit might be a composed entity consisting of a student and a class that the student attends, i.e., a registration.

A *population* might now be defined by a SW query that produces a table whose tuples (i.e., variable bindings) correspond to the objects that identify a

[2] Technically one needs to be aware that the generation of a sample with the help of a search engine or a crawler might introduce a bias, for example, if snowball sampling is employed.

statistical unit. In the example in Figure 2 we might define a query to generate a population table with objects student and class; a tuple then stands for the statement that a student registered in a particular class. Sampling, as before, selects a proper random subset of the population. A particular aspect of SW data is the dominance of relationships between objects. Thus, features that are calculated for a statistical unit might reflect this relationship structure.

Technically, one first generates a data matrix. The number of rows in the data matrix is identical to the number of tuples in the sample table, i.e., the number of statistical units in the sample. A statistical unit is a primary key for the table. The data matrix has a fixed number of columns corresponding to the number of features, which are derived for each unit. All matrix entries are initialized to be N/A (not available or missing) and will (partially) be replaced by feature values as described in the following two steps.

Next, database views[3] are generated that contain as attributes the objects in a statistical unit (respectively a subset of those objects) plus additional attributes. In Figure 2, the first view contains the student's ID and the student's IQ, the second view contains the class ID and the class difficulty and the third view contains the student ID, the class ID and the grade the student obtained in a class. Note that views can be generated from rather complex queries.

In the next step, relational features are calculated based on these views. In the simplest case each statistical unit is represented exactly in one tuple in each view and features are calculated based on the tuple attributes. The situation becomes more complex if a statistical unit is not represented in a view or if it is represented more than once. In the first case, i.e., a statistical unit is not represented in a view, one either enters zero or another default entry (e.g., the number of a person's children is zero) or one does not overwrite the corresponding N/A entry in the data matrix (e.g., when a student's IQ is unknown). In the second case, i.e., a statistical unit is represented in more than one tuple in a particular view —in the example if a student attended a class twice and got two grades— some form of aggregation can be applied (number-of, average, max, min, etc.). In domains like the SW, many-to-many relations often play a significant role and can lead to a large number of sparse features: The number of items a customer has acquired is typically still very small if compared to the total number of items. In the case that object IDs are used as features, the learning algorithm needs to be able to handle the potentially high-dimensional sparse data. Technically, it might be possible to execute the described steps, i.e., the generation of the sample, the views and the data matrix, in one SQL/SPARQL operation.

Finally, the statistical model can be applied beyond the sample to the population. It is important to note that we have a well-defined statistical problem as long as we restrict the analysis to the world in as much it is represented in the SW. Of course the SW can grow (and shrink) such that online learning and transfer learning might become applicable. To what degree the statistical model

[3] A view is a stored query accessible as virtual table composed of the result set of a query. Alternatively, one could also work with a temporary or persistent table.

can be generalized to the real world needs to be analyzed carefully since sometimes the SW data are generated by multiple parties for their own reasons and not for the purpose of a statistical analysis.

3.3 Search for the Best Features

So far it was assumed that the user would be able to define the features of interest. In particular in supervised learning one is often interested to automate the selection of the best input features. Popescul and Ungar [29] describe a relational learning approach based on a greedy search for optimal relational features derived from SQL queries (see also [30]). Features are dynamically generated by a refinement style search over SQL queries including aggregation, statistical operators, groupings, richer join conditions and argmax based queries. The features are used to predict the target relation using logistic regression. Additional features are generated by clustering, which leads to new "invented" attributes. The authors obtain good results on citation prediction and document classification. It is straightforward to implement a similar search procedure on SW data. Note that the automatic generation of candidate features is certainly attractive; on the other hand the computational burden is quite large; feature definition based on the experimenters insight and some pruning might be adequate in many applications.

3.4 Discussion

Statistical learning on the SW, as presented, is highly scalable since the determining factor is the number of statistical units in the sample, which basically is independent of the size of the SW. One needs to be aware that sampling with the help of a search engine or a crawler might introduce a bias. The queries, which need to be executed for the calculation of the features, can be executed efficiently with current technology [25]. Ontological background knowledge can be integrated in different ways. First, one might perform complete or partial materialization, which would derive statements from reasoning prior to training. Recall that total materialization is only feasible with less expressive ontologies. Second, since the ontology is part of the RDF-graph, features can be defined including ontological concepts of a statistical units, respecting the subclass restrictions. This has effectively been employed in [31]. It is conceivable that the trained statistical models could be added to an extended "probabilistic" ontology, indicated by the arrow at the bottom of Figure 2. In addition, the statistical models derive probabilistic statements about the truth values of triples. For example, if —based on a trained model— it can be derived that a person has a high IQ, this information could be added to the SW [6]. An option is a weighted RDF-triple, the weight reflecting the likelihood that the statement is true. Moreover, if it was found that particular features generated during learning are valuable, one could define corresponding statements and add those to the SW as "invented predicates". The same is true for the latent variables introduced in a cluster analysis or in a principle component analysis (PCA). We should emphasize again that statistical inference strictly speaking is only applicable within the experimental setting of a particular statistical unit, population and sampling approach. Thus if a statistical model allows the conclusion that *statement X is true with 90% probability*, this is only valid in a particular statistical context. Experiments have shown, for example, that predictive performance can depend to some degree on the object selected as statistical unit. An interesting aspect is that the results from a number of statistical models could be combined in a committee machine [32].

Feature generation is nontrivial and might exploit prior knowledge that is partially available in the domain ontology. For example it is relevant that a person only has exactly one age, exactly one mother, but zero or more children. In fact it would be desirable that the ontological information could be exploited in a way such that the statistical framework is automatically constructed requiring a minimum of additional domain background knowledge from the user. A problem with less expressive ontologies might be that one cannot express negation. Consider the example of gene-gene interactions where the literature primarily reports positive results, i.e., positive evidence for gene-gene interactions. Evidence that two genes do not interact would be important to report but might be difficult to represent in less expressive ontologies.

Maybe the most important issue in SW learning concerns missing or incomplete data. We can make a closed-world assumption and postulate that the world only exists in as much as it is represented in the SW: besides the statements that are known to be true or can be derived to be true, all statements are assumed false. Naturally, in many cases we are really interested to perform inference in the real world and it is more appropriate to assume that the truth values of some statements are unknown. Here we should distinguish, first, the case that statistical units are missing and, second, the case that due to missing information features cannot be calculated or features are biased. The first case is not a problem if statistical units are missing at random, e.g., if some of the students at a university are unknown. The situation is more complex if the fact that a statistical unit is missing is dependent on features of interest, e.g., if only smart students are in the data base. Then the missing data mechanism should be included in the statistical model. For the second case consider that the age of a person's father is an important feature that is not available: Either the age of a person's father might be unknown or a person father's ID might be unknown. Another example is that if the number of transactions is an important feature, the feature might be biased if not all transactions are recorded. If a closed-world assumption is not appropriate, one could deal with missing features using the appropriate procedures known from statistics [33]. Again, the missing data mechanism should be included in the statistical model. Also note that ontological information can be quite relevant for dealing with missing data. For example if we know that a person has brown eye color we know that all other statements about eye color must be false, since a person has only one eye color. Note that there are statistical models that can easily deal with missing data such as naive Bayes, many nearest neighbor methods, or kernel smoothers.

Naturally there are cases where simple missing data models are not appropriate, since missing data can render the independent sample assumption invalid. Consider objects of type *Person* and the properties *friendOf* and *income* and *age*. Furthermore assume that from the age of a person and from the income of a person's friends we can predict the income of a person with some certainty. If all features are available, then training an appropriate classifier is straightforward. If in training and testing the income of a person and of a person's friends are partially unknown, we have the situation that the income prediction for one person depends on the income prediction of the person's friends. The situation, where for the prediction of features of a statistical unit (here a person's income) the same features of linked statistical units are required, is typical for data defined on networks. In the analysis of social networks, this situation is referred to as a collective classification problem and a mechanism is added to propagate information using, e.g., Gibbs sampling, relaxation labeling, iterative classification or loopy belief propagation. Recent overviews are presented in [34,35]. One of the first papers demonstrating the benefits of collective classification in social networks is [36] and some important contributions are described in [37,38,39,40]. It is likely that collective classification will also concern SW applications. Interestingly, many social networks have been shown to exhibit homophily, which means that objects with similar attributes (e.g., persons with similar income) are linked (e.g., are friends). In networks exhibiting homophily, simple propagation models, for example based on Gaussian random field models employed in semi-supervised learning [41], give very competitive results. Collective classification is highly related to the relational graphical model approaches described in Section 6, in particular dependency networks [42,43]. Note, that in collective classification, features for different statistical units are not independent and a statistical analysis becomes more involved. Also recall, that we assumed previously that statistical units were selected randomly from the population. In contrast, in collective classification problems the statistical units (for both training and test) would typically be defined by the complete RDF-graph or a connected RDF-subgraph (compare Section 6).

4 Inductive Logic Programming

Inductive logic programming (ILP) encompasses a number of approaches that attempt to learn logical clauses[4] In the view of the discussion in the last section, ILP uses logical (binary) features derived from logical expressions, typically

[4] A (logical) literal is either an atomic sentence or a negated atomic sentence. A clause is a disjunction of literals: $l_1 \vee l_2 \ldots \vee l_n$. In a definite clause exactly one literal is positive. A definite clause can be written as an implication (if-then rule): $(\neg l_1 \wedge \neg l_2 \wedge \ldots \wedge \neg l_{n-1}) \Rightarrow l_n$ where l_n was assumed to be the positive literal. To the left of the implication sign is the rule body and l_n is the rule head. A Horn clause has at most one positive literal. A function-free definite clause is a datalog clause. A program clause can contain negative literals in the body.

conjunctions of (negated) atoms. Recent extensions on probabilistic ILP have also address uncertain domains.

4.1 ILP Overview

This section is on “strong” ILP, which covers the majority of ILP approaches and is concerned with the classification of statistical units and on predicate definition[5]. Strong ILP performs modeling in relational domains that is somewhat related to the approach discussed in the previous section. Let’s consider FOIL (First Order Inductive Learner) as a typical representative [44]. The outcome of FOIL is a set of definite clauses (a particular if-then rule) with the same head (then-part).

Here is an example (modified from [45]). Let the statistical unit be a customer with ID *CID*. $VC = 1$, indicates that someone is a valuable customer, $GC = 1$ indicates that someone owns a golden credit card and $SB = 1$ indicates that someone would buy a sailboat. The first rule that FOIL might have learned is that a person is interested in buying a sailboat if this person owns a gold card. The second rule indicates that a person would buy a sailboat if this person is older than 30 and has at least once made a credit card purchase of more than 100 EURO:

$$\begin{aligned} &\mathit{sailBoat(CID,\ SB} = 1) \leftarrow \mathit{customer(CID,GC} = 1) \qquad (1)\\ &\mathit{sailBoat(CID,\ SB} = 1) \leftarrow \mathit{customer(CID,\ Age)}\\ &\qquad\qquad \wedge\ \mathit{purchase(CID,\ PID,\ Value,\ PM)}\\ &\qquad\qquad \wedge\ \mathit{PM} = \mathit{credit\text{-}card} \wedge\ \mathit{Value} > 100 \wedge \mathit{Age} > 30. \end{aligned}$$

In rule learning FOIL uses a covering paradigm. Thus the first rule is derived to correctly predict as many positive examples as possible (covering) with a minimum number of false positives. Subsequent rules then try to cover the remaining positive examples. The head of a rule (then-part) is a predicate and the body (the if-part) is a product of (negated) atoms containing constants and variables.[6] Naturally, there are many variants of FOIL. FOIL uses a top down search strategy for refining the rule bodies, PROGOL [46] a bottom up strategy and GOLEM [47] a combined strategy. Furthermore, FOIL uses a conjunction of atoms and negated atoms in the body, whereas other approaches use PROLOG constructs. The community typically discusses the different approaches in terms of language bias (which rules can the language express), search bias (which rules can be found) and validation bias (when does validation tell me to stop refining a rule). An advantage of ILP is that also non-grounded background knowledge can be taken into account (typically in form of a set of definite clauses that might be part of an ontology).

[5] A predicate definition is a set of program clauses with the same predicate symbol in their heads.

[6] FOIL learning is called learning from entailment in ILP terminology.

In view of the discussion in the last section, the statistical unit corresponds to a customer, and FOIL introduces a binary target feature (1) for the target predicate *sailBoat(CID, SB)*. The second feature (2) is one if the customer owns a golden credit card and zero otherwise. Then a view is generated with attribute CID. A CID is entered in that view each time the person has made a credit card purchase of more then 100 EURO, but only if that person is older than 30 years. The third feature (3) is binary and is equal to one if the CID is present in the view at least once and zero otherwise. FOIL then applies a very simple combination rule: if feature (2) or feature (3) is equal to one for a customer, then the target feature (1) is true.

4.2 Propositionalization, Upgrading and Lifting

ILP approaches like FOIL can be decomposed into the generation of binary features (based on the rule bodies) and a logical combination, which in case of FOIL is quite simple. As stated before, ILP approaches contain a complex search strategy for defining optimal rule bodies. If, in contrast, the generation of the rule bodies is performed as a preprocessing step, the process is referred to as propositionalization [48]. Instead of using the simple FOIL combination rule, other feature-based learners are often used. It has been proven that in some special cases, propositionalization is inefficient [49]. Still, propositionalization has produced excellent results. The binary features are often collected through simple joins of all possible attributes. An early approach to propositionalization is LINUS [50].

The inverse process to propositionalization is called *upgrading* (or lifting) [51] and turns a propositional feature-based learner into an ILP learner. The main differences to propositionalization is that the optimization of the features is guided by the improvement of the performance of the overall system. It turns out that many strong ILP systems can be interpreted as upgraded propositional learners: FOIL is an upgrade of the propositional rule-induction program CN2 and PROGOL can be viewed as upgrading the AQ approach to rule induction. Additional upgraded systems are Inductive Classification Logic (ICL [52]) that uses classification rules, TILDE [53] and S-CART that use classification trees, and RIBL [54] that uses nearest neighbor classifiers. nFOIL [55] combines FOIL with a naive Bayes (NB) classifier by changing the scoring function and by introducing probabilistic covering. nFoil was able to outperform FOIL and propositionalized NB on standard ILP problems. kFoil [56] is another variant that derives kernels from FOIL-based features.

4.3 Discussion

ILP algorithms can easily be applied to the SW if we identify atoms with basic statements. ILP fits well into the basically deterministic framework of the SW. In many ways, statistical SW learning as presented in Section 3 is related to ILP's propositionalization; the main difference is the principled statistical framework of the former. Thus most of the discussion on scalability in Section 3 carries over to ILP's propositionalization. When ILP's complex search strategy for defining optimal rule bodies is applied, training time increases but is still proportional to the number of samples. An interesting new aspect is that ILP produces definite clauses that can be integrated, maybe with some restrictions, into the Semantic Web Rule Language. ILP approaches that consider learning with description logic (and clauses) are described, for example, in [57,58,14,15,16,17]. An empirical study can be found in [59].

5 Learning with Relational Matrices

Another representation of a basic statement (RDF-triple) is a matrix entry. Consider the triple *(User, buys, Item)*. Recall that a standard relational representation would be the table *buys* with attributes *User* and *Item*. A relational adjacency matrix on the other hand has as many rows as there are users and as many columns as there are items and as many matrix entries as there are possibly true statements. A matrix entry is equal to one if the item was actually bought by a user and is equal to zero otherwise. Thus SW data can be represented as a set of matrices where the name of the matrix is the property of the relation under consideration. Matrix decomposition/reconstruction methods, e.g., the principle component analysis (PCA) and other more scalable approaches have been very successful in the prediction of unknown matrix entries [60]. Lippert *et al.* [61] have shown how several matrices can be decomposed/reconstructed jointly and have shown that this increases predictive performance if compared to single matrix decompositions. By filling in the unknown entries via matrix decomposition/reconstruction, the approach has an inherent way of dealing with data that is missing at random. Care must be taken if missing at random is not justified. In [61], one type of statement concerns gene-gene interactions where only positive statements are known. Reconstructed matrix entries can, as before, be entered into the SW, e.g., as weighted triples. Scalability of this approach has not been studied in depth but the decomposition scales approximately proportional to the number of known matrix entries. Note that the approach performs a prediction for all unknown statements in one global decomposition/reconstruction step. In contrast, the previous approaches would learn separate models for each statistical unit under consideration. Other approaches, which learn with the relational adjacency matrix, are described in [62] and [63].

6 Relational Graphical Models

The approaches described in Sections 3 and 4 aim at describing the statistical, respectively logical, dependencies between features derived from SW data. In contrast the matrix decomposition approach in the last section and the relational graphical models (RGMs) in this section predict the truth values of all basis statements (RDF-triples) in the SW. Unlike the matrix decomposition techniques in the last section, the RGMs are probabilistic models and statements are represented by random variables. RGMs can be thought of as upgraded versions of regular graphical models, e.g., Bayesian networks, Markov networks, dependency networks and latent variable models. RGMs have been developed in the context of frame-based logical representations, relational data models, plate models, entity-relationship models and first-order logic. Here, we attempt to relate the basic ideas of the different approaches to the SW framework.

6.1 Possible World Models on the SW

Consider all constants in the SW (i.e., all objects and literal values) and all statements that can possibly be true [7]. Now one introduces a binary random variable U for each possibly true statement (grounded atom), where $U = 1$ if the corresponding statement is true and $U = 0$ otherwise. In a graphical model, U would be identified with a node. These nodes should not be confused with the nodes in the RFD-graph, which represent URIs; rather U stands for a potential link in the RDF-graph. We can reduce the number of random variables if type constraints are available and if the truth value of some statements are assumed known in each world under consideration (e.g., if object-to-object statements are all assumed known, as in the basic PRM model in Subsection 6.2). If statements are mutually exclusive, e.g., the different blood types of a person, one might integrate several statements into one random variable using, e.g., multi-state multinomial variables or continuous variables (to encode, e.g., a person's height). An assignment of truth values to all random variables defines a possible world[8]. RGMs assign a probability distribution to each world in the form $P(\boldsymbol{U} = \boldsymbol{u})$.[9] The approaches differ in how these probabilities are defined and mapped to random variables, and how they are learned.

6.2 Directed RGMs

The probability distribution in a directed RGM, i.e., relational Bayesian model, can be written as

$$P(\boldsymbol{U} = \boldsymbol{u}) = \prod_{U \in \boldsymbol{U}} P(U|par(U)).$$

U is represented as a node in a Bayesian network and arcs are pointing from all parent nodes $par(U)$ to the node U. One now partitions all elements of $\boldsymbol{U}$ into node-classes. Each U belongs to exactly one node-class. The key property of all U in the same node-class is that their local distributions are identical, which means that $P(U|par(U))$ is the same for all nodes within a node-class and can be described by a truth-table or more complex representations such as decision trees. For example, all nodes representing the IQ-values of students

[7] We only consider a function-free case.

[8] RGM modeling would be termed learning from interpretation in ILP terminology.

[9] Our discussion includes the case that we are only interested in a conditional distribution of the form $P(\boldsymbol{U} = \boldsymbol{u}|\boldsymbol{V} = \boldsymbol{v})$, as in conditional random fields [64].

in a university might form a node class, all nodes representing the difficulties of university courses might form a node class, and the nodes representing the grades of students in courses might form a node-class. Care must be taken, that no directed loops are introduced in the Bayesian network in modeling or structural learning.

Probabilistic Relational Models (PRMs): PRMs were one of the first published RGMs and found great interest in the statistical machine learning community [65,19]. PRMs combine a frame-based logical representation with probabilistic semantics based on directed graphical models. The nodes in a PRM model the probability distribution of object attributes whereas the relationships between objects are assumed known. Naturally, this assumption simplifies the model greatly. In context of the SW object attributes would primarily correspond to object-to-literal statements. In subsequent papers PRMs have been extended to also consider the case that relationships between objects (in context of the SW these would roughly be the object-to-object statements) are unknown, which is called *structural uncertainty* in the PRM framework [19]. The simpler case, where one of the objects in a statement is known, but the partner object is unknown, is referred to as *reference uncertainty.* In reference uncertainty the number of potentially true statements is assumed known, which means that only as many random nodes need to be introduced. The second form of structural uncertainty is referred to as *existence uncertainty*, where binary random variables are introduced representing the truth values of relationships between objects.

For some PRMs, regularities in the PRM structure can be exploited (encapsulation) and exact inference is possible. Large PRMs require approximate inference; commonly, loopy belief propagation is being used. Learning in PRMs is likelihood based or based on empirical Bayesian learning. Structural learning typically uses a greedy search strategy, where one needs to guarantee that the ground Bayesian network does not contain directed loops.

More Directed RGMs: A *Bayesian logic program* is defined as a set of Bayesian clauses [66]. A Bayesian clause specifies the conditional probability distribution of a random variable given its parents on a template level, i.e. in a node-class. A special feature is that, for a given random variable, *several* such conditional probability distributions might be given. As an example, *bt(X)* | *mc(X)* and *bt(X)* | *pc(X)* specify the probability distribution for blood type given the two different dispositions *mc(X)* and *pc(X).* The truth value for *bt(X)* | *mc(X), pc(X)* can then be calculated based on various combination rules (e.g., noisy-or). In a Bayesian logic program, for each clause there is one conditional probability distribution and for each Bayesian predicate (i.e., node-class) there is one combination rule. *Relational Bayesian networks* [67] are related to Bayesian logic programs and use probability formulae for specifying conditional probabilities. *Relational dependency networks* [42] also belong to the family of directed RGMs and learn the dependency of a node given its Markov blanket using decision trees.

6.3 Undirected RGMs

The probability distribution of an undirected graphical model or Markov network can be written as

$$P(\boldsymbol{U} = \boldsymbol{u}) = \frac{1}{Z} \prod_k g_k(u_k)$$

where $g_k(.)$ is a potential function, u_k is the state of the k-th clique and Z is the partition function normalizing the distribution. One often prefers a more convenient log-linear representation of the form

$$P(\boldsymbol{U} = \boldsymbol{u}) = \frac{1}{Z} \exp \sum_k w_k f_k(u_k)$$

where the feature functions f_k can be any real-valued function and where $w_i \in \mathbb{R}$.

We will discuss two major approaches that use this representation: Markov logic networks and relational Markov models.

Markov Logic Networks (MLN): Let F_i be a formula of first-order and let $w_i \in \mathbb{R}$ be a weight attached to each formula. Then a MLN L is defined as a set of pairs (F_i, w_i) [68] [69]. One introduces a binary node for each possible grounding of each predicate appearing in L (i.e., in context of the SW we would introduce a node for each possible statement), given a set of constants $c_1, \ldots, c_{|C|}$. The state of the node is equal to 1 if the ground atom/statement is true, and 0 otherwise (for an N-ary predicate there are $|C|^N$ such nodes). A grounding of a formula is an assignment of constants to the variables in the formula (considering formulas that are universally quantified). If a formula contains N variables, then there are $|C|^N$ such assignments. The nodes in the Markov network $M_{L,C}$ are the grounded predicates. In addition the MLN contains one feature for each possible grounding of each formula F_i in L. The value of this feature is 1 if the ground formula is true, and 0 otherwise. w_i is the weight associated with F_i in L. A Markov network $M_{L,C}$ is a grounded Markov logic network of L with

$$P(\boldsymbol{U} = \boldsymbol{u}) = \frac{1}{Z} \exp \left(\sum_i w_i n_i(\boldsymbol{u}) \right)$$

where $n_i(\boldsymbol{u})$ is the number of formula groundings that are true for F_i. MLN makes the unique names assumption, the domain closure assumption and the known function assumption, but all these assumptions can be relaxed.

A MLN puts weights on formulas: the larger the weight, the higher is the confidence that a formula is true. When all weights are equal and become infinite, one strictly enforces the formulas and all worlds that agree with the formulas have the same probability.

The simplest form of inference concerns the prediction of the truth value of a grounded predicate given the truth values of other grounded predicates (conjunction of predicates) for which the authors present an efficient algorithm. In the first phase, the minimal subset of the ground Markov network is returned that is required to calculate the conditional probability. It is essential that this subset is small since in the worst case, inference could involve alle nodes. In the second phase Gibbs sampling in this reduced network is used.

Learning consists of estimating the w_i. In learning, MLN makes a closed-world assumption and employs a pseudo-likelihood cost function, which is the product of the probabilities of each node given its Markov blanket. Optimization is performed using a limited memory BFGS algorithm.

Finally, there is the issue of structural learning, which, in this context, defines the employed first order formulae. Some formulae are typically defined by a domain expert *a priori*. Additional formulae can be learned by directly optimizing the pseudo-likelihood cost function or by using ILP algorithms. For the latter, the authors use CLAUDIAN [70], which can learn arbitrary first-order clauses (not just Horn clauses, as many other ILP approaches).

Relational Markov Networks (RMNs): RMNs generalize many concepts of PRMs to undirected RGMs [40]. RMNs use conjunctive database queries as clique templates. By default, RMNs define a feature function for each possible state of a clique, making them exponential in clique size. RMNs are mostly trained discriminately. In contrast to MLN, RMNs, as PRMs, do not make a closed-world assumption during learning.

6.4 Latent Class RGMs

The infinite hidden relational model (IHRM) [71] presented here is a directed RGM (i.e., a relational Bayesian model) with latent variables.[10] The IHRM is formed as follows. First, we partition all objects into classes $K_1, ...K_{|K|}$, using, for example, ontological class information. For each object in each class, we introduce a statement (*Object, hasHiddenState, H*). If *Object* belongs to class K_i, then $H \in \{1, \ldots, N_{K_i}\}$, i.e., the number of states of H is class dependent. As before, we introduce a random variable or node U for each grounded atom, respectively potentially true basic statement. Let Z_{Object} denote the random variables that involve *Object* and H. Z_{Object} is a latent variable or latent node since the true state of H is unknown. $Z_{Object} = j$ stand for the statement that (*Object, hasHiddenState*, j).

We now define a Bayesian network where the nodes Z_{Object} have no parents and the parents of the nodes for all other statement are the latent variables of the objects appearing in the statement. In other words, if U stands for the fact that ($Object_1$, *property*, $Object_2$) is true, then there are arcs from Z_{Object_1} and Z_{Object_2} to U. The object-classes of the objects in a statement together with the property define a node-class for U. If the property value is a literal, then the only parent of U is Z_{Object_1}.

In the IHRM we let the number of states in each latent node to be infinite and use the formalism of Dirichlet process mixture models. In inference, only a small number of the infinite states are occupied, leading to a clustering solution where

[10] Kemp et al. [72] presented an almost identical model independently.

the number of states in the latent variables N_{C_i} is automatically determined during inference.

Since the dependency structure in the ground Bayesian network is local, one might get the impression that only local information influences prediction. This is not true, since in the ground Bayesian network, common children U with evidence lead to interactions between the parent latent variables. Thus information can propagate in the network of latent variables. Training is based on various forms of Gibbs sampling (e.g., the Chinese restaurant process) or mean field approximations. Training only needs to consider random variables U corresponding to statements that received evidence, e.g., statements that are either known to be true or known not to be true; random variables that correspond to statements with an unknown truth value (i.e., without evidence) can completely be ignored.

The IHRM has a number of key advantages. First, no structural learning is required, since the directed arcs in the ground Bayesian network are directly given by the structure of the SW graph. Second, the IHRM model can be thought of as an infinite relational mixture model, realizing hierarchical Bayesian modeling. Third, the mixture model allows a cluster analysis providing insight into the relational domain.

The IHRM has been applied to recommender systems, for gene function prediction and to develop medical recommender systems. The IHRM was the first relational model applied to trust learning [20]. In [31] it was shown how ontological class information can be integrated into the IHRM.

6.5 Discussion

RGMs have been developed in the context of frame-based logical representations, relational data models, plate models, entity-relationship models and first-order logic but the main ideas can easily be adapted to the SW data model. One can distinguish two cases. In the first case, an RGM learns a joint probabilistic model over the complete SW or a segment of the SW. This might be the most elegant approach since there is only one (SW-) world and the dependencies between the variables are truthfully modeled, as discussed in Subsection 3.4. The draw back is that the computational requirements scale with the number of statements whose truth value is known or even the number of all potentially true statements. More appropriate for large-scale applications might be the second case where one applies the sampling approach as described in Section 3. As an example consider that the statistical unit is a student. A data point would then not correspond to a set of features but to a local subgraph that is anchored at the statistical unit, e.g., the student. As before sampling would make the training time essentially independent of SW-size. Ontological background knowledge can be integrated as discussed in Section 3. First, one can employ complete or partial materialization, which would derive statements from reasoning prior to training. Second, an ontological subgraph can be included in the subgraph of a statistical unit [31]. Also note that the MLN might be particularly suitable to exploit ontological background information: ontologies can formulate some of the first-order formulas that are the basis for the features in the MLN. PRMs have been extended to learn class hierarchies (PRM-CH), which can be a basis for ontology learning.

The RGM approaches typically make an open world assumption.[11] The corresponding random variables are assumed missing at random such that the approaches have an inherent mechanism to deal with missing data. If missing at random is not justified, then more complex missing data models need to be applied. As before, based on the estimated probabilities, weighted RDF-triples can be generated and added to the SW.

7 Unstructured Data and the SW

The realization of the SW heavily depends on (1) available ontologies and (2) the annotation of unstructured data with ontology-based meta data. Manual ontology development and manual annotation are two well known SW bottlenecks. Thus learning-based approaches for both tasks are finding increasing interest [2,9]. In this section, we will concentrate on two important tasks, namely ontology learning and semantic annotation (for a compilation of current work on ontology learning and population see, e.g., [73]). A particulary important source of information for these tasks is unstructured or semi-structured textual data. Note that there is a close relationship between textual data and SW data. Textual data describes, first, ontological concepts and relationships between concepts (e.g., a text might contain the sentence: *We all know that cats are mammals*) and, second, instances and relationships between instances (e.g., a document might inform us that: *Marry is married to Jack*). However, the input data for ontology learning and semantic annotation will not be limited to textual data; especially once the SW will be realized to a greater extent, other types of input data will become increasingly important. Learning ontologies from e.g., XML-DTDs, UML diagrams, database schemata or even raw RDF-graphs is also of great interest [74], but is out of scope here. The outline of this section is as follows: first, we consider the case, where a text corpus of interest is given and the task is to infer a prototype ontology. Second, given a text corpus and an ontology, we want to infer instances of the concepts and their relations.

7.1 Learning Ontologies from Text

Ontology learning, in general, consists of several subtasks. This includes the identification of terms, synonyms, polysems, concepts, concept hierarchies, properties, property hierarchies, domain and range constraints and class definitions. These tasks can be illustrated as the so-called ontology learning layer cake [74]. Different approaches differ mainly in the way a concept is defined and one distinguishes between formal ontologies, terminological ontologies and prototype-based ontologies [75]. In prototype-based ontologies, concepts are represented by collections of prototypical instances, which are arranged hierarchically in subclusters. An example would be the concept disease, which is defined by a set

[11] There are some exceptions, e.g., MLN make a closed-world assumption in training.

of diseases. Since prototype-based ontologies are defined by instances, they lack definitions and axiomatic grounding. In contrast, typical examples for terminological ontologies are WordNet and the Medical Subject Headings (MeSH[12]). Terminological ontologies are described by concept labels and both nouns and verbs are organized into hierarchies, defined by hypernym or subclass relationships. For example a disease is defined in WordNet as an impairment of health or a condition of abnormal functioning. Terminological ontologies typically also lack axiomatic grounding. A formal ontology such as OWL, in contrast, is seen as a conceptualization, whose categories are distinguished by axioms and definitions [76]. Most of the state-of-the-art approaches focus on learning prototype-based ontologies. Work on learning terminological or formal ontologies is still quite rare. Here, the big challenge is to deal with uncertain and often even contradicting extracted knowledge, introduced during the ontology learning process. This is addressed in [77], which presents a system that is able to transform a terminological ontology to a consistent formal OWL-DL ontology.

Prototype ontologies are often learned based on some type of hierarchical clustering techniques such as single-link, complete-link or average-link clustering. According to Harris' distributional hypothesis [78], semantic similarity between words can be assessed via the syntactic context, which they are sharing in a corpus. Thus most approaches base the semantic relatedness between words on some distributional similarity between the words. Usually, a vector-space model is used as input and the linguistic context of a term is described by, e.g., syntactic dependencies, which the term establishes in a corpus [79] The input vector for a term to be clustered can be, e.g., composed of syntactic expressions such as prepositional phrases following a verb or adjective modifiers. See [80] for an illustrative example for assessing the semantic similarity of terms. Hierarchical clustering, in its classical form, distinguishes between agglomerative (bottom-up) and divisive (top-down) clustering, whereas the agglomerative form is most commonly used due to its computational efficiency. Somewhat different from hierarchical clustering is the divisive bi-section-Kmeans algorithm, which yielded competitive results for document clustering [81] and has been applied to the task of learning concept hierarchies as well [82,83]. Another variant is the the Formal Concept Analysis (FCA) [84]. FCA is closely related to bi-clustering and tries to build a lattice of so-called formal concepts from a vector space model. FCA thereby makes use of order theory and analyzes the covariance between objects and their features. The reader is referred to [84] for more information.

Recently, [74] set up a benchmark to compare the above mentioned clustering techniques for learning concept hierarchies. While each of the methods had its own benefits, FCA performed better in terms of recall and precision. All the methods just mentioned, face the problem of not being able to appropriately label the resulting clusters, i.e., to determine the name of the concept. To overcome this limitation and to guide the clustering process, [85] either use hyponyms extracted from WordNet or use Hearst patterns [86] derived either from the corpus under investigation or from the WWW.

[12] http://www.nlm.nih.gov/mesh/

Another type of technique for learning prototype ontologies, comes from the topic modeling community, an active research area of machine learning [87,9]. Topic models are generative models based upon the idea that a document is made of a mixture of topics, where a topic is represented by a distribution over words. Powerful techniques such as Latent Semantic Analysis (LSA) [88], Probabilistic Latent Semantic Analysis (PLSA) [89] or Latent Dirichlet Allocation (LDA) [90] have been proposed for the automated extraction of useful information from large document collections. Applications include document annotation, query answering, document summarization, automatic topic extraction as well as trend analysis. Generative statistical models such as the ones mentioned, have been proven effective in addressing these problems. In general, the following advantages of topic models are highlighted in the context of document modeling: First, topics can be extracted in a complete unsupervised fashion, requiring no initial labeling of the topics. Second, the resulting representation of topics for a document collection is interpretable and last but not least, each document is usually expressed by a mixture of topics, thus capturing the topic combinations that arise in documents [89,90,91]. When applying topic modeling techniques in an ontology learning setting, a topic is referred to as concept. To satisfy the hierarchical structure of prototype ontologies, [87] extends the PLSA method to an hierarchical version, where super concepts are introduced. While yielding already impressive results with this kind of techniques, [87] concentrates on learning prototype ontologies, where no labeling of the concept is needed. Furthermore, the hierarchy of the ontology is assumed to be known a priori. Learning the hierarchical order in topic models is an area of growing interest. Here, [92] introduced hierarchical LDA, which models the setup of the tree-structure of the topics as a Chinese Restaurant Process (CRP). As a consequence, the hierarchy is not fixed a priori, instead it is a part of the learning process. To overcome the limitation of unlabeled topics or concepts, [93] tries to automatically infer an appropriate label for multinomial topic models. [9] discusses ontology learning based on topic models in context of the SW.

Ontology Merging, Alignment and Evolution: In many cases no dominant ontology will exist, which leads to the problem that several ontologies need to be merged and aligned. In [11] these tasks have been addressed with the support of machine learning. Another aspect is that an ontology is not a rigid and fixed construct — ontologies will evolve with time. Thus, the structure of an ontology will change and new concepts will be needed to be inserted into an existing ontology. This leads to another task, where machine learning can play a role in ontology engineering: ontology refinement and ontology evolution. This task is usually treated as classification task [76]. The reader is referred to [76,10] for more information.

7.2 Semantic Annotation

Besides ontological support, a second prerequisite to put the SW into practice, is the availability of machine-readable meta data. Producing human readable text from SW data is simple since an RDF triple can easily be formulated as a textual statement. However, even though the statement won't be powerfully eloquent, it will still serve its purpose. The inverse is much more difficult, i.e., the generation of triples from textual data. This process is called semantic annotation, knowledge markup or meta data generation [94]. Hereby, we are following the notion of semantic annotation as linguistic annotations (such as named entities, semantic classes, etc.) as well as user annotations like tags (see the ECIR 2008 workshop on 'Exploiting Semantic Annotations in Information Retrieval'[13]).

The Information Extraction (IE) community provides a number of approaches for these tasks. IE is traditionally defined as the process of filling the fields and records of a database from unstructured text and is seen as precursor to data mining [95]. Usually, the fields are filled with named entities (i.e., Named Entity Recognition (NER)), such as persons, locations or organizations. IE first populates a database from unstructured text and data mining then aims to find patterns. IE is, dependent on the task, made up of five subtasks: segmentation, classification, finding associations and last but not least normalization and deduplication [95]. Segmentation refers to the identification of text phrases, which describe entities of interest. Classification is the assignment to predefined types of entities, while finding associations is the identification of relations between the entities (i.e., relation extraction). Normalization and deduplication describe the task of merging different text descriptions with the same meaning (e.g., mapping entities to URIs).

NER is an active field of research and several evaluation conferences such as the Message Understanding Conference (MUC-6)[96], the Conference on Computational Natural Language Learning (CoNLL-2003) [97] and in the biomedical domain, the Critical Assessments of Information Extraction systems in Biology (BioCreAtIvE I+II[14]) [98] have attracted a lot of interest. While in MUC-6 the focus was NER for persons, locations, organizations in an English newswire domain, CoNLL-2003 focused on language-independent NER. BioCreAtIvE focused on the recognition of biomedical entities, in this case gene and protein mentions. The methods proposed for NER vary, in general, in their degree of reliance on dictionaries, and their different emphasis on statistical or rule-based approaches. Numerous machine learning techniques have been applied to NER tasks such as Support Vector Machines [99], Hidden Markov Models [100], Maximum Entropy Markov Models [101] and Conditional Random Fields [64].

An F-measure in the mid-90s can now be achieved for extracting persons, organizations and locations in the newswire domain [95]. For extracting gene and protein mentions, however, the F-measure lies currently in the mid- to high 80s (see the BioCreAtIvE II conference for details). So NER can provide high accuracy solutions for the SW, but typically only for a small number of classes, mostly because of a limited amount of labeled training data. However, when populating an existing ontology, there will often be the need to be able to extract hundreds of classes of entities. Thus, systems which are able to scale to a large

[13] http://www.yr-bcn.es/dokuwiki/doku.php?id=ecir08_entity_workshop_proposal

[14] http://biocreative.sourceforge.net/biocreative_2.html

number of classes on a large amount of unlabeled data are needed. Also flexible and domain-independent recognition of entities is an important and active field of research. State-of-the-art approaches try to extract hundreds of entity classes in an unsupervised fashion [102], but so far with a fairly low accuracy. Promising areas, which could help to overcome current limitations of supervised IE systems, are semi-supervised learning [103,104] as well as active learning [105].

The same entities can have different textual representation (e.g., 'Clark Kent', 'Kent Clark' and 'Mr. Clark' refer to the same person). Normalization is the process of standardizing the textual expressions. This task is usually also referred to as entity resolution, co-reference resolution or normalization and deduplication. The Stanford Entity Resolution Framework (SERF), e.g., has the goal to provide a framework for generic entity resolution [106]. Other techniques for entity resolution employ relational clustering [107] as well as probabilistic topic models [108].

Another important task is the identification of relations between instances of concepts (i.e., the association finding stage in the traditional IE workflow). Up to now, most of research on text information extraction has focused on tagging named entities. The Automatic Content Extraction (ACE) program provides annotation benchmark sets for the challenging task of relation extraction. At ACE, this task is called Relation Detection and Characterization (RDC). A representative system using an SVM with a rich set of features, reports results for Relation Detection (74.7% F-measure) and 68.0% F-measure for the RDC task [109]. Co-occurrence based relation extraction is a simple, effective and popular method [110], but usually suffers of a lower recall, since entities can co-occur for many other reasons. Other methods are kernel-based [111] or rule-based [112]. Recently, [113] propose a new method that treats relation extraction as sequential labeling task. They extend Conditional Random Fields (CRFs) towards the extraction of semantic relations. Hereby, they focus on the extraction of relations between genes and diseases (five types of relations) as well as between disease and treatment entities (eight types of relations). The work applies the authors' method to a biomedical textual database and provides the resulting network of genes and diseases in a machine-readable RDF graph. Thereby, gene and disease entities are normalized to Bio2RDF[15] URIs.

8 First Experiments in the Analysis of FOAF-Data

The purpose of the FOAF (Friend of a Friend) project [114] is to create a web of machine-readable pages describing people, the relationships between people and people's activities and interests, using W3C's RDF technology. The FOAF ontology is defined using RDFS/OWL and is formally specified in the FOAF Vocabulary Specification 0.91 [115]. In our study we employed the IHRM model as described in Section 6. The trained IHRM can, for instance, recommend new friendships, the affiliations of persons, and their interests and projects. Furthermore one might want to predict attributes of certain persons, like their gender

[15] http://bio2rdf.org/

or age. Finally, by interpreting the clustering results of the IHRM one can answer typical questions from social network analysis concerning the relationships between members of the FOAF social system.

In general FOAF data is either uploaded by each person individually or generated automatically from user profiles of community websites like Tribe.net, LiveJournal.com or my.opera.com. The resulting network of linked FOAF-files can be gathered using a FOAF harvester, a so called "scutter". Some scutter dumps are readily available for download, e.g., in one large rdf/xml-file or stored in a relational database.

Even though this use case only covers a very basic statistical inference problem on the SW, there still are major challenges to meet. First, there are characteristics of the FOAF-data that need special consideration: For instance, the actual data is extremely sparse. With more than 100000 users, there are far more potential links as actual links between persons.

Another typical characteristic of friendship data is that the topology of the *knows*-RDF-graph consists of a few barely connected star graphs, corresponding to a few active network users with a long list of friends as the "center" of the stars and the mass of users that don't specify their friends. Second, there are prevalent challenges of SW data in general that can also be observed in a FOAF analysis. For instance, there is a variety of additional untested and potentially conflicting ontologies specified by users. If this information is ignored by only considering data consistent with the FOAF ontology, most of the information specified by users is ignored. This also applies to the almost arbitrary use of literals by users. For instance the relation *interest* with range *Document* defined in the FOAF-schema is in reality mostly used with a literal instead. Consequently, this results in a loss of semantic information. To still make use of this information one would, e.g., need to use automated semantic annotation as described in Section 7. Another preprocessing step that needs to be considered in practice is the materialization of triples, which can be inferred deductively. For example there might be an instance of the relation *holdsAccount* with domain *Person* in the data, which is not given in the schema. However, from the ontology it can be inferred that *Person* is a *subClassOf Agent* which in turn has a property *holdsAccount*. As stated before, total materialization is only feasible in less expressive ontologies.

Considering these issues, it becomes clear that there are not only theoretical but also a large number of interesting practical challenges for learning on the SW.

9 Conclusions

Data in Semantic Web formats will bring many new opportunities and challenges to machine learning. Machine learning complements ontological background knowledge by exploiting regularities in the data while being robust against some of the inherent problems with Semantic Web data such as contradicting information and non-stationarity. A general issue with machine learning is that the problem of missing information needs to be carefully addressed in learning, in particular if either the selection of statistical units or the probability that a feature is missing depend on the features of interest, which is common in many-to-many relations.

We began with a section on feature-based statistical learning on the Semantic Web. This procedure is widely applicable, scales well with the size of the Semantic Web and provides a promising general purpose learning approach. The greatest challenge here is that most feature-based statistical learning approaches have no inherent way of dealing with missing data requiring additional missing data models. A common situation in social network data is that features in linked objects are mutually dependent and need to be modeled jointly. One can expect that this will also often occur in SW data and SW learning will benefit from ongoing research in social network modeling.

We then presented the main approaches in inductive logic programming. Inductive logic programming has the potential to learn deterministic constraints that can be integrated into the employed ontology. We presented a discussion on learning with relational matrices, which is quite attractive if multiple many-to-many relations are of interest, as in recommendation systems. We then studied relational graphical models. Although these approaches were originally defined in various frameworks, e.g., frame-based logical representation, relational data models, plate models, entity-relationship models and first-order logic, they can easily be modified to be applicable in context of the Semantic Web. Relational graphical models are capable of learning a global probabilistic Semantic Web model and inherently can deal with missing data. Scalability to the size of the Semantic Web might be a problem for RGMs and we discussed subgraph sampling as a possible solution. All approaches have means to include ontological background knowledge by complete or partial materialization. In addition, the ontological RDF-graph can be incorporated in learning and ontological features can be derived and exploited. Ontologically supported machine learning is an active area of research. It is conceivable that in future ontological standards, the developed statistical models could become in integral part of the ontology. Also, we have discussed that most presented approaches can be used to produce statements that are weighted by their probability value derived from machine learning, complementing statements that are derived form logical reasoning. An interesting opportunity is to include weighted triples in Semantic Web queries.

We reported about initial work on learning ontologies from textual data and on the semantic annotation of unstructured data. So far, this concerns the most advanced work in Semantic Web learning covering ontology construction and management, ontology evaluation, ontology refinement, ontology evolution, as well as the mapping, merging and alignment of ontologies. In addition there is growing work on Semantic Web mining extending the capabilities of standard web mining, although most of this work needs to wait for the Semantic Web to be realized on a large scale.

In summary, machine learning has the potential to realize a number of exciting applications on the Semantic Web and can complement axiomatic inference by exploiting regularities in the data.

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. We thank Frank Harmelen, and two anonymous reviewers for their valuable comments.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (2001)
2. van Harmelen, F.: Semantische Techniken stehen kurz vor dem Durchbruch. C'T Magazine (2007)
3. Fensel, D., Hendler, J.A., Lieberman, H., Wahlster, W.: Spinning the Semantic Web: Bringing the World Wide Web to its Full Potential. MIT Press, Cambridge (2003)
4. Ontoprise: Neue Version des von Ontoprise entwickelten Ratgebersystems beschleunigt die Roboterwartung. Ontoprise Pressemitteilung (2007)
5. LarKC: The large Knowledge Collider. EU FP 7 Large-Scale Integrating Project (2008), `http://www.larkc.eu/`
6. Incubator Group: Uncertainty Reasoning for the World Wide Web. W3C (2005), `http://www.w3.org/2005/Incubator/urw3/`
7. Berendt, B., Hotho, A., Stumme, G.: Towards semantic web mining. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342. Springer, Heidelberg (2002)
8. Grobelnik, M., Mladenic, D.: Automated knowledge discovery in advanced knowledge management. Library Hi Tech News incorporating Online and CD Notes 9(5) (2005)
9. Grobelnik, M., Mladenic, D.: Knowledge discovery for ontology construction. In: Davies, J., Studer, R., Warren, P. (eds.) Semantic Web Technologies. Wiley, Chichester (2006)
10. Bloehdorn, S., Haase, P., Sure, Y., Voelker, J.: Ontology evolution. In: Davies, J., Studer, R., Warren, P. (eds.) Semantic Web Technologies. Wiley, Chichester (2006)
11. Bruijn, J.d., Ehrig, M., Feier, C., Martin-Recuerda, F., Scharffe, F., Weiten, M.: Ontology mediation, merging, and aligning. In: Davies, J., Studer, R., Warren, P. (eds.) Semantic Web Technologies. Wiley, Chichester (2006)
12. Fortuna, B., Grobelnik, M., Mladenic, D.: Ontogen: Semi-automatic ontology editor. In: HCI (9) (2007)
13. Mladenic, D., Grobelnik, M., Foruna, B., Grcar, M.: Knowledge discovery for the semantic web (submitted, 2008)
14. Lisi, F.A.: Principles of inductive reasoning on the semantic web: A framework for learning in AL-Log. In: Fages, F., Soliman, S. (eds.) PPSWR 2005. LNCS, vol. 3703. Springer, Heidelberg (2005)
15. Lisi, F.A.: A methodology for building semantic web mining systems. In: The 16th International Symposium on Methodologies for Intelligent Systems (2006)
16. Lisi, F.A.: Practice of inductive reasoning on the semantic web: A system for semantic web mining. In: Alferes, J.J., Bailey, J., May, W., Schwertel, U. (eds.) PPSWR 2006. LNCS, vol. 4187. Springer, Heidelberg (2006)
17. Lisi, F.A.: The challenges of the semantic web to machine learning and data mining. In: Tutorial at ECML 2006 (2006)
18. Fukushige, Y.: Representing probabilistic relations in rdf. In: ISWC-URSW (2005)

19. 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, Cambridge (2007)
20. Rettinger, A., Nickles, M., Tresp, V.: A statistical relational model for trust learning. In: Proceeding of 7th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2008 (2008)
21. W3C: World Wide Web Consortium, `http://www.w3.org/`
22. Antoniou, G., van Harmelen, F.: A Semantic Web Primer. MIT Press, Cambridge (2004)
23. Herman, I.: Tutorial on the Semantic Web. W3C, `http://www.w3.org/People/Ivan/CorePresentations/SWTutorial/Slides.pdf`
24. Tauberer, J.: Resource Description Framework, `http://rdfabout.com/`
25. Kiryakov, A.: Measurable targets for scalable reasoning. Ontotext Technology White Paper (2007)
26. Fahrmeir, L., Künstler, R., Pigeot, I., Tutz, G., Caputo, A., Lang, S.: Arbeitsbuch Statistik, 4th edn. Springer, Heidelberg (2004)
27. Casella, G., Berger, R.L.: Statistical Inference. Duxbury Press (1990)
28. Trochim, W.: The Research Methods Knowledge Base, 2nd edn. Atomic Dog Publishing (2000)
29. 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, Cambridge (2007)
30. Karalič, A., Bratko, I.: First order regression. Machine Learning 26(2-3) (1997)
31. Reckow, S., Tresp, V.: Integrating ontological prior knowledge into relational learning. Technical report, Siemens (2007)
32. Tresp, V.: Committee machines. In: Hu, Y.H., Hwang, J.N. (eds.) Handbook for Neural Network Signal Processing. CRC Press, Boca Raton (2001)
33. Little, R.J.A., Rubin, D.B.: Statistical Analysis with Missing Data, 2nd edn. Wiley, Chichester (2002)
34. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. AI Magazine (Special Issue on AI and Networks), forthcoming (forthcoming, 2008)
35. Macskassy, S., Provost, F.: Classification in networked data: a toolkit and a univariate case study. Machine Learning (2007)
36. Chakrabarti, S., Dom, B., Indyk, P.: Enhanced hypertext categorization using hyperlinks. In: SIGMOD (1998)
37. Neville, J., Jensen, D.: Iterative classification in relational data. In: AAAI (2000)
38. Lu, Q., Getoor, L.: Link-based classification. In: ICML (2003)
39. Getoor, L., Friedman, N., Koller, D., Taskar, B.: Learning probabilistic models of link structure. journal of machine learning research. Journal of Machine Learning Research (2002)
40. Taskar, B., Abbeel, P., Koller, D.: Discriminative probabilistic models for relational data. In: Uncertainty in Artificial Intelligence, UAI (2002)
41. Zhu, X.: Semi-supervised learning literature survey. Technical report, Technical Report 1530, Department of Computer Sciences, University of Wisconsin (2005)
42. Neville, J., Jensen, D.: Dependency networks for relational data. In: ICDM 2004: Proceedings of the Fourth IEEE International Conference on Data Mining (2004)
43. Neville, J., Jensen, D.: Relational dependency networks. Journal of Machine Learning Research (2007)
44. Quinlan, J.R.: Learning logical definitions from relations. Machine Learning 5(3) (1990)

45. Džeroski, S.: Inductive logic programming in a nutshell. In: Getoor, L., Taskar, B. (eds.) Introduction to Statistical Relational Learning. MIT Press, Cambridge (2007)
46. Muggleton, S.: Inverse entailment and Progol. New Generation Computing, Special issue on Inductive Logic Programming 13(3-4) (1995)
47. Muggleton, S., Feng, C.: Efficient induction of logic programs. In: Proceedings of the 1st Conference on Algorithmic Learning Theory, Ohmsma, Tokyo, Japan (1990)
48. Kramer, S., Lavrac, N., Flach, P.: From propositional to relational data mining. In: Džeroski, S., Lavrac, L. (eds.) Relational Data Mining. Springer, Heidelberg (2001)
49. De Raedt, L.: Attribute-value learning versus inductive logic programming: The missing links (extended abstract). In: Page, D.L. (ed.) ILP 1998. LNCS, vol. 1446. Springer, Heidelberg (1998)
50. Lavrač, N., Džeroski, S., Grobelnik, M.: Learning nonrecursive definitions of relations with LINUS. In: Kodratoff, Y. (ed.) EWSL 1991. LNCS, vol. 482. Springer, Heidelberg (1991)
51. Van Laer, W., De Raedt, L.: How to upgrade propositional learners to first order logic: A case study. In: Machine Learning and Its Applications, Advanced Lectures (2001)
52. De Raedt, L., Van Laer, W.: Inductive constraint logic. In: Zeugmann, T., Shinohara, T., Jantke, K.P. (eds.) ALT 1995. LNCS, vol. 997. Springer, Heidelberg (1995)
53. Blockeel, H., De Raedt, L.: Top-down induction of first-order logical decision trees. Artificial Intelligence 101(1-2) (1998)
54. Emde, W., Wettschereck, D.: Relational instance based learning. In: Saitta, L. (ed.) Machine Learning - Proceedings 13th International Conference on Machine Learning (1996)
55. Landwehr, N., Kersting, K., De Raedt, L.: nFOIL: Integrating naïve bayes and FOIL. In: Veloso, M., Kambhampati, S. (eds.) Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI 2005) (2005)
56. Landwehr, N., Passerini, A., De Raedt, L., Frasconi: kFOIL: Learning simple relational kernels. In: National Conference on Artificial Intelligence (AAAI) (2006)
57. Cohen, W.W., Hirsh, H.: Learning the CLASSIC description logic: Theoretical and experimental results. In: Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR 1994) (1994)
58. Rouveirol, C., Ventos, V.: Towards learning in CARIN-ALN. In: International Workshop on Inductive Logic Programming (2000)
59. Edwards, P., Grimnes, G., Preece, A.: An empirical investigation of learning from the semantic web. In: ECML/PKDD, Semantic Web Mining Workshop (2002)
60. Takacs, G., Pilaszy, I., Nemeth, B., Tikk, D.: On the gravity recommendation system. In: Proceedings of KDD Cup and Workshop 2007 (2007)
61. 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)
62. Yu, K., Chu, W., Yu, S., Tresp, V., Xu, Z.: Stochastic relational models for discriminative link prediction. In: Advances in Neural Information Processing Systems 19 (2006)
63. Yu, S., Yu, K., Tresp, V.: Soft clustering on graphs. In: Advances in Neural Information Processing Systems 18 (2005)

64. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of 18th International Conference on Machine Learning (2001)
65. Koller, D., Pfeffer, A.: Probabilistic frame-based systems. In: Proceedings of the National Conference on Artificial Intelligence (AAAI) (1998)
66. Kersting, K., De Raedt, L.: Bayesian logic programs. Technical report, Albert-Ludwigs University at Freiburg (2001)
67. Jaeger, M.: Relational bayesian networks. In: Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI) (1997)
68. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning 62(1-2) (2006)
69. 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, Cambridge (2007)
70. De Raedt, L., Dehaspe, L.: Clausal discovery. Machine Learning 26 (1997)
71. Xu, Z., Tresp, V., Yu, K., Kriegel, H.P.: Infinite hidden relational models. In: Uncertainty in Artificial Intelligence (UAI) (2006)
72. Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: Poceedings of the National Conference on Artificial Intelligence (AAAI) (2006)
73. Buitelaar, P., Cimiano, P.: Ontology Learning and Population: Bridging the Gap between Text and Knowledge. IOS Press, Amsterdam (2008)
74. Cimiano, P.: Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. Springer, Heidelberg (2006)
75. Sowa, J.F.: Ontology, metadata, and semiotics. In: International Conference on Computational Science (2000)
76. Biemann, C.: Ontology learning from text: A survey of methods. LDV Forum 20(2) (2005)
77. Völker, J., Haase, P., Hitzler, P.: Learning expressive ontologies. In: Buitelaar, P., Cimiano, P. (eds.) Ontology Learning and Population: Bridging the Gap between Text and Knowledge. IOS Press, Amsterdam (2008)
78. Harris, Z.S.: Mathematical Structures of Language. Wiley, Chichester (1968)
79. Hindle, D.: Noun classification from predicate-argument structures. In: Meeting of the Association for Computational Linguistics (1990)
80. Cimiano, P., Staab, S.: Learning concept hierarchies from text with a guided agglomerative clustering algorithm. In: Proceedings of the ICML 2005 Workshop on Learning and Extending Lexical Ontologies with Machine Learning Methods (2005)
81. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering techniques. In: KDD Workshop on Text Mining (2000)
82. Maedche, A., Staab, S.: Semi-automatic engineering of ontologies from text. In: Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering (2000)
83. Cimiano, P., Hotho, A., Staab, S.: Comparing conceptual, divise and agglomerative clustering for learning taxonomies from text. In: Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI 2004 (2004)
84. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Heidelberg (1997)

85. Cimiano, P., Staab, S.: Learning concept hierarchies from text with a guided agglomerative clustering algorithm. In: Biemann, C., Paas, G. (eds.) Proceedings of the ICML 2005 Workshop on Learning and Extending Lexical Ontologies with Machine Learning Methods (2005)
86. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th conference on Computational linguistics (1992)
87. Paaß, G., Kindermann, J., Leopold, E.: Learning prototype ontologies by hierachical latent semantic analysis. In: Knowledge Discovery and Ontologies (2004)
88. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science 41 (1990)
89. Hofmann, T.: Probabilistic latent semantic analysis. In: Uncertainty in Artificial Intelligence (UAI) (1999)
90. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. 3 (2003)
91. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proc. Natl. Acad. Sci. USA (2004)
92. Blei, D.M., Griffiths, T.L., Jordan, M.I., Tenenbaum, J.B.: Hierarchical topic models and the nested chinese restaurant process. In: Advances in Neural Information Processing Systems (2003)
93. Mei, Q., Shen, X., Zhai, C.: Automatic labeling of multinomial topic models. In: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (2007)
94. Bontcheva, K., Cunningham, H., Kiryakov, A., Tablan, V.: Semantic annotation and human language technology. In: Davies, J., Studer, R., Warren, P. (eds.) Semantic Web Technologies. Wiley, Chichester (2006)
95. McCallum, A.: Information extraction: distilling structured data from unstructured text. Queue 3(9) (2005)
96. Grishman, R., Sundheim, B.: Design of the MUC-6 evaluation. In: MUC6 1995: Proceedings of the 6th conference on Message understanding (1995)
97. Erik, F., Sang, T., De Meulder, F.: Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In: Daelemans, W., Osborne, M. (eds.) Proceedings of CoNLL 2003 (2003)
98. Yeh, A., Morgan, A., Colosimo, M., Hirschman, L.: Biocreative task 1a: gene mention finding evaluation. BMC Bioinformatics 6 (2005)
99. Mayfield, J., McNamee, P., Piatko, C.: Named entity recognition using hundreds of thousands of features. In: Proceedings of the seventh conference on natural language learning (2003)
100. Ray, S., Craven, M.: Representing sentence structure in hidden markov models for information extraction. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (2001)
101. Lin, Y.F., Tsai, T.H., Chou, W.C., Wu, K.P., Sung, T.Y., Hsu, W.L.: A maximum entropy approach to biomedical named entity recognition. In: Proceedings of 4th ACM SIGKDD Workshop on Data Mining in Bioinformatics (BioKDD) (2004)
102. Cimiano, P., Völker, J.: Towards large-scale, open-domain and ontology-based named entity classification. In: Angelova, G., Bontcheva, K., Mitkov, R., Nicolov, N. (eds.) Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP (2005)
103. Chapelle, O., Schölkopf, B., Zien, A.: Semi-Supervised Learning. MIT Press, Cambridge (2006)

104. Ando, R.K., Zhang, T.: A high-performance semi-supervised learning method for text chunking. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (2005)
105. Kristjansson, T.T., Culotta, A., Viola, P.A., McCallum, A.: Interactive information extraction with constrained conditional random fields. In: Nineteenth National Conference on Artificial Intelligence (AAAI) (2004)
106. Benjelloun, O., Garcia-Molina, H., Menestrina, D., Su, Q., Whang, S.E., Widom, J.: Swoosh: A generic approach to entity resolution. VLDB Journal (2008)
107. Bhattacharya, I., Getoor, L.: Collective entity resolution in relational data. ACM Transactions on Knowledge Discovery from Data 1(1) (2007)
108. Bhattacharya, I., Getoor, L.: A latent dirichlet model for unsupervised entity resolution. In: SIAM SDM 2006 (2006)
109. Zhou, G., Su, J., Zhang, J., Zhang, M.: Exploring various knowledge in relation extraction. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (2005)
110. Ramani, A.K., Bunescu, R.C., Mooney, R.J., Marcotte, E.M.: Consolidating the set of known human protein-protein interactions in preparation for large-scale mapping of the human interactome. Genome Biol. 6(5) (2005)
111. Bunescu, R.C., Mooney, R.J.: Subsequence kernels for relation extraction. In: Advances in Neural Information Processing Systems (2005)
112. Ono, T., Hishigaki, H., Tanigami, A., Takagi, T.: Automated extraction of information on protein-protein interactions from the biological literature. Bioinformatics 17(2) (2001)
113. Bundschus, M., Dejori, M., Stetter, M., Tresp, V., Kriegel, H.: Extraction of semantic biomedical relations from text using conditional random fields. BMC Bioinformatics 9 (2008)
114. Brickley, D., Miller, L.: The Friend of a Friend (FOAF) project, `http://www.foaf-project.org/`
115. Brickley, D., Miller, L.: FOAF Vocabulary Specification, `http://xmlns.com/foaf/spec/`