KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering

Statistical Relational Learning of Natural Language

Mathias Verbeke

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

December 2014

Statistical Relational Learning of Natural Language

Mathias VERBEKE

Examination committee:
Prof. dr. Adhemar Bultheel, chair
Prof. dr. Luc De Raedt, supervisor
Prof. dr. Bettina Berendt, co-supervisor
Prof. dr. ir. Maurice Bruynooghe
Prof. dr. Jesse Davis
Prof. dr. Paolo Frasconi
(Università degli Studi di Firenze)
Prof. dr. Walter Daelemans
(University of Antwerp)
Prof. dr. Antal van den Bosch
(Radboud University Nijmegen)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

December 2014

Uitgegeven in eigen beheer, Mathias Verbeke, Celestijnenlaan 200A box 2402, B-3001 Heverlee (Belgium)

ISBN 978-94-6018-939-5

D/2014/7515/162

Abstract

While understanding natural language is easy for humans, it is complex for computers. The main reasons for this can be found in the structural nature and the inherent ambiguity of natural language. Correctly interpreting language therefore requires one to take into account the necessary context. In order to perform natural language understanding by means of machine learning techniques, an appropriate representation is required that takes into account this relational information and integrates the necessary background knowledge into the learning process.

Statistical relational learning is well-suited to represent this structural information, and to incorporate the necessary context and background knowledge for natural language understanding. Furthermore, its inherent probabilistic nature offers opportunities to deal with linguistic ambiguity. This thesis investigates the promise of statistical relational learning for natural language processing and provides evidence for the utility of this approach.

As a first contribution, we demonstrate the expressiveness and interpretability of the relational representation on two natural language learning problems. Furthermore, we explore the importance of the declarative approach for the inclusion of contextual information, and analyze the influence of the relational representation by a comparison of several machine learning techniques. A second contribution is the extension of the graph kernel–based relational learning framework kLog with a natural language processing module, in order to obtain a full relation learning framework for natural language learning. As a third contribution, we introduce relational regularization and feature ranking in order to assess the importance of the relational features. Finally, we extend rule learning to a probabilistic setting and explore its application in the context of machine reading.

Beknopte samenvatting

Het begrijpen van natuurlijke taal is gemakkelijk voor mensen, maar is erg complex voor computers. De belangrijkste redenen hiervoor zijn het structurele karakter en de inherente dubbelzinnigheid van natuurlijke taal. Bijgevolg is voor een correcte interpretatie van taal context vereist. Om het begrijpen van natuurlijke taal uit te voeren door middel van technieken uit machinaal leren is een gepaste voorstelling noodzakelijk die deze relationele informatie in rekening brengt, en de nodige achtergrondkennis opneemt in het leerproces.

Statistisch relationeel leren is bijzonder geschikt voor het uitdrukken van deze structurele informatie, en laat ook toe om de context en achtergrondkennis op te nemen die nodig is voor het begrijpen van natuurlijke taal. Bovendien biedt de probabilistische aard van deze aanpak perspectieven voor het omgaan met linguïstische ambiguïteit. Deze thesis onderzoekt de belofte van statistisch relationeel leren voor natuurlijke taalverwerking en toont het nut van deze aanpak aan.

Als eerste bijdrage tonen we de expressiviteit en interpreteerbaarheid van de relationele representatie voor twee problemen in natuurlijke taalverwerking. We verkennen ook het belang van de declaratieve aanpak voor het opnemen van contextuele informatie, en analyseren de invloed van de relationele representatie door een vergelijking van verschillende technieken voor machinaal leren. Een tweede bijdrage is de uitbreiding van kLog, een framework voor relationeel leren met kernels, met een module voor natuurlijke taalverwerking, om zo een raamwerk voor relationeel leren van natuurlijke taal te bekomen. Als derde bijdrage introduceren we technieken voor relationele regularisatie en feature ordening, die toelaten het belang van relationele eigenschappen te beoordelen. Tenslotte breiden we het leren van regels uit naar een probabilistische setting, en verkennen de toepassing hiervan in het kader van machinaal lezen.

Acknowledgments

Doing a Ph.D. is a challenging, yet exciting and rewarding experience. During its course, you meet many people, who all directly or indirectly influence the ideas and the general outcome. I would like to thank everyone who crossed my path during this journey, and in some way contributed to the result lying in front of you.

In the first place, I am very grateful to Luc De Raedt and Bettina Berendt, my supervisor and co-supervisor. I would like to thank them for awakening my scientific interest in general and introducing me to the exciting field of machine learning and data mining in particular while guiding me during my Master's thesis. I was very happy they offered me the opportunity to pursue this research. I am very thankful for their invaluable support and encouragement during the past years, for setting the example by being outstanding researchers, and, despite their busy schedule, for always making the time to discuss new ideas or give feedback. Thank you, Luc, for each time challenging me with new questions, giving me the freedom to find the appropriate solutions, while still keeping track of the general picture. Basically, thank you for teaching me the what, why and and how of being a researcher. Thank you, Bettina, for your enduring enthusiasm and optimism, and for letting me lose track of this general picture once in a while to explore interesting research ideas outside the thesis context. Besides that, I also very much enjoyed our discussions on life, politics, and language. It was a great pleasure to work with both of you.

Next, I would like to thank the members of the jury, Maurice Bruynooghe, Jesse Davis, Paolo Frasconi, Walter Daelemans and Antal van den Bosch, for carefully reading the text, and their detailed and valuable feedback which helped me to improve it. Special thanks go to Paolo and Walter for the nice collaboration over the past years. They both had a big influence on the course of this thesis, and it was an enriching experience to work with them. I also would like to thank Adhemar Bultheel for chairing the defense.

One of the nicest things about doing research is that it offers one the chance of working together with many people, both inside the research group and beyond. First of all, I would like to thank all the DTAI and former HMDB colleagues for creating a very enjoyable environment to work in. To this end, a lot of credit goes to all the people I had the pleasure of sharing an office with; thank you, Angelika, Bo, Bogdan, Davide, Dimitar, Dries, Francesco, Ilija, Jonas, Joris, Jose, Laura, McElory, Nik, Parisa and Sten, for being great colleagues and friends, for the interesting discussions on research ideas, and for the nice memories on the city lunches, the running trips, the Dutch-Italian language jokes, etc. A special word of thanks goes to Kurt and Anton, for their technical help with kLog and ProbLog respectively. I would also like to thank Roser Morante and Vincent Van Asch from CLiPS at the University of Antwerp for the fruitful collaboration, and their expertise in computational linguistics. While we only started collaborating when he already left Leuven, I am very thankful to Fabrizio Costa, for the interesting discussions, his valuable insights, and for being a great person to work with. In the second half of my Ph.D., I had the pleasure to meet Leen d'Haenens and Michaël Opgenhaffen from the Institute of Media Studies. Their abiding enthusiasm and open mind made it great to explore our interdisciplinary research ideas together. I also enjoyed the many lunches with all former and present members of the "Alma gang".

Furthermore, all of this would not have been possible without the support of the administrative and system group staff. Especially, I would like to thank Karin for her organisational talent in smoothly making all the necessary travel arrangements. Furthermore, I am grateful for the financial support by the Research Foundation-Flanders (FWO project G.0478.10 - Statistical Relational Learning of Natural Language), and the Academische Stichting Leuven, for enabling me to set up an IdeaLab to explore interdisciplinary research ideas in collaboration with researchers from the Faculty of Social Sciences.

Apart from colleagues, I would also like to especially thank my friends. In the first place, a word of thanks goes to OVG, for always being interested in the Ph.D. stories, and for understanding when I disappeared under the radar for a while and could not be present at one of the meetups. Second, to the friends from the Red Cross, for the summer camp memories and all events in between.

Finally, and above all, I would like to thank my parents, for giving me every opportunity to find my own way in life, while being always there when needed. Isabelle, thanks for being such a great sister and friend, and for always being able to make me laugh. Mieke, Erwin, Lore, Lander and Joy, thank you for becoming family during these past few years, and for making me always feel welcome. My final words of thanks go to Hanne, for bringing love, happiness and balance, and whom I am very lucky to have in my life. Thank you, for everything!

Mathias, December 2014

Contents

List of Figures

List of Tables

Part I

Overture

Chapter 1

Introduction

Since the origin of the domain more than sixty years ago, understanding language by means of computers is one of the longstanding goals of *Artificial Intelligence* (AI). While this is an easy task for humans, it is complex to do this automatically. To illustrate this, consider the following example:

Example 1.1. *Alan Turing is one of the most influential scientists of all times. The English mathematician is often considered as the father of modern Computer Science.*

In order to answer the question *Which logician is seen as the founder of Computer Science?*, one needs to know that *mathematician* refers to *Alan Turing* and *father* is figurative language referring to the role of founding the Computer Science field. Furthermore, knowing that logic is a subfield of mathematics is essential to get to the right answer. Language thus is inherently *relational* in nature.

While processing this information is natural for humans, in order for it to be done computationally, it needs to be transformed into a representation that can be understood by computers. This representation needs to encompass both the data and the background knowledge. Traditional approaches often resort to a so-called *propositional representation*, in which the data is represented as a table consisting of columns representing particular properties and rows representing examples with values for each of the properties. A simplified propositional representation of part of the text in Example 1.1 is given in Table 1.1. It represents the words in the text, with for each word its two surrounding words as context and their respective part-of-speech (POS) tags, which indicate the linguistic type of a word in the sentence.

word	POS-tag	prev. word	prev. POS-tag	next word	next POS-tag
			...		
most	adverb	the	determiner	influential	adjective
influential	adjective	most	adverb	scientist	noun
scientist	noun	influential	adjective	of	preposition
			...		

Table 1.1: Propositional representation of a part of the text in Example 1.1.

However, to encode the necessary information in order to be able to *learn from and interpret* language, this representation is often impractical or insufficient. *Statistical relational learning* is a technique in AI which allows one to represent both the necessary data and encode additional background knowledge which might be useful to solve a particular problem. A relational representation, again representing the text from Example 1.1, is shown in Figure 1.1[1]. In addition to the individual words (purple rectangles) and the POS-tags (yellow ovals), now also the dependency relations between words (orange diamonds), indicating the structure of the sentence, are represented. Furthermore, the named entities (pink ovals), indicating which words are proper names, and the coreference relations (green diamonds), which connect words that refer to the same entity in the real world (in this case, all coreferent words refer to *Alan Turing*), are included as well. The *IS_A* relation represents some additional background knowledge, namely that a *mathematician* is a *scientist*. As may be clear, the relational representation is more concise, and offers a more flexible approach to take contextual information into account when compared to the propositional representation. In the first half of this thesis we will study the advantages of statistical relational learning to represent the contextual, relational information and background knowledge which is necessary for a machine to learn from text.

The relational information contained in text is also a valuable source to extract knowledge from. Let us again consider the text in Example 1.1. For a human, it is easy to extract that *Alan Turing* was a logician. In a machine, this information can be expressed as `hasProfession(Alan Turing, logician)`. Furthermore, if one also knows that a *logician* studies *logic*, i.e., `studies(logician, logic)`, this allows one to infer that *Alan Turing*'s research area is *logic*. This knowledge can be represented in the form of the following rule: *if* `hasProfession(Alan Turing, logician)` *and* `studies(logician, logic)` *then* `hasResearchArea(Alan Turing, logic)`. While it is in the human nature to extract and reason about this information, for a machine this is not a trivial operation, and often comes along with uncertainty about the extracted

[1]For the ease of understanding, only a part of the linguistic information is represented. The notation will be introduced and explained in more detail in Chapter 2.

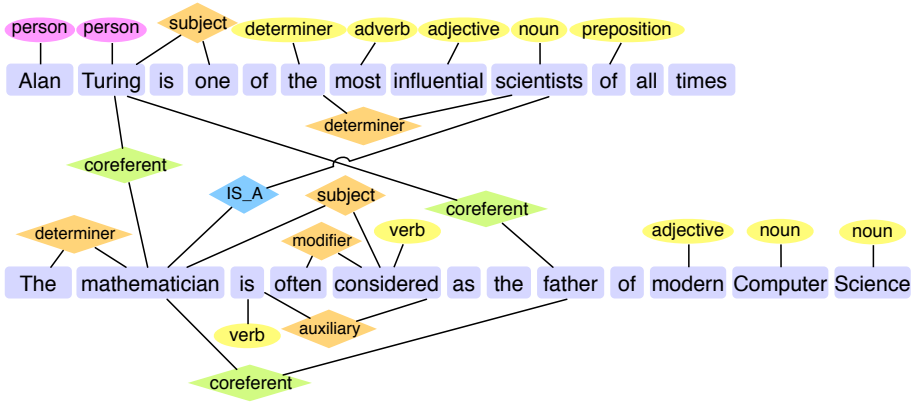


Figure 1.1: Relational representation of the text in Example 1.1.

information. The second half of this thesis will be concerned with automatically extracting rules from text in the presence of uncertainty.

This research is part of computational natural language learning, a subfield of Artificial Intelligence. Both the origin of this subfield and the actual interest in the relational representation stem from the parallel course of the history of AI and linguistics.

John McCarthy was the first to use the term "Artificial Intelligence" to refer to "the science and engineering of making intelligent machines, especially intelligent computer programs" in 1956. However, Alan Turing already lectured on "Intelligent Machines" in 1947, and he may have been the first to propose that AI was best researched by programming computers rather than by building machines (McCarthy, 2007).

At around the same time, modern linguistics emerged. In 1957, Burrhus F. Skinner proposed a behaviorist approach to language learning in his book *Verbal Behaviour* (Skinner, 1957), in which he accounted for language development by means of environmental influence. Not only his book, but also a critical review of it by Noam Chomsky (Chomsky, 1959) attracted a lot of attention. In this review, Chomsky argued that the behaviorist approach did not account for the creativity that is present in human language acquisition. For example, children cannot understand sentences, but still come up with new ones solely based on the input they get from their environment. He defended the theory he presented in *Syntactic Structures* (Chomsky, 1957), namely that some linguistic knowledge is innate, and children already have prior knowledge about language structure, but still need to learn the idiosyncratic features of the language. The theory is

based on syntactic models that could be formally represented. This attracted the interest of the AI community, and led to the development of *computational linguistics* and *natural language processing*. The boundary between these two fields is rather vague. *Computational linguistics* is commonly defined as the discipline that studies linguistic processes from a computational perspective, while *natural language processing* is generally considered as referring to the field studying the automation of textual and linguistic analysis.

In order to test the ability of a machine to exhibit intelligent behavior, Alan Turing proposed the "Imitation Game", which evolved into the well-known Turing Test (Turing, 1950). The test consists of a human interrogator posing written questions. The test is successful if the interrogator cannot tell whether the responses to these questions come from a person or a computer. In order to pass it, a computer needs to possess a number of capabilities: *knowledge representation* to store what it knows and hears, *automated reasoning* to use this knowledge to answer questions and draw conclusions, *machine learning* to detect patterns in this knowledge and adapt to new circumstances, and *natural language processing* to be able to communicate (Russell and Norvig, 2009). Later, a number of variations on the original Turing Test were designed. For example, the Total Turing Test by Harnad (1991) adds two additional requirements to the original test, in order to be able to test the ability of the subject to perceive and manipulate objects. On top of the original capabilities, this also requires *computer vision* and *robotics* respectively. Despite the controversy around the test (Moor, 2003), the six disciplines corresponding with these capabilities represent the current main research areas of artificial intelligence.

The requirement of the original Turing Test to interact with the interrogator in a text-based manner led to the longstanding goal of natural language processing, and one of the ultimate goals of AI in general, namely acquiring human knowledge by automatically understanding texts. Initially, this was expected to be a relatively easy problem, as witnessed by the claim from participants in the Georgetown experiment, an influential demonstration of machine translation. They stated that machine translation would be a solved problem in three to five years (Dostert, 1955), which nourished the presumption that automatic language understanding and knowledge acquisition would be an easily-solved problem. After a decade of high expectations and disillusionment, attention shifted from Chomskyan to corpus linguistics, i.e., the analysis of language by studying large collections of text (*corpora*). Together with the introduction of statistical techniques and machine learning in natural language processing, this revolutionized the domain (the so-called *statistical revolution*). One of the first attempts toward the goal of natural language understanding was SHRDLU (Winograd, 1972). It consisted of a language parser that allowed the user to instruct the program to move various objects around in a so-called "blocks

world"; a setting containing blocks in different forms. Over the past decades, tremendous progress has been made in this field, with IBM's Watson (Ferrucci et al., 2010) as one of the most prominent examples to date. It is an AI system that is able to answer questions posed in natural language, and has won the quiz show *Jeopardy!* against the reigning human champions.

This thesis is situated at the intersection of these two domains; machine learning and natural language processing. Section 1.1 will briefly introduce both, as well as their respective subfields, *statistical relational learning* and *computational natural language learning*, that are in the focus of this work. This will be followed by the motivation and research questions (Section 1.2) and the contributions of this thesis (Section 1.3). Finally, an overview of the outline of the text is provided in Section 1.4.

1.1 Context

In this section, we will briefly introduce machine learning and natural language processing, the two research fields in which this thesis is situated. Subsequently, we will turn to statistical relational learning and computational natural language learning, which are the specific areas of interest in this thesis.

1.1.1 Machine Learning and Natural Language Processing

The term "Machine Learning" (ML) was coined in 1952 by Arthur Samuel, who defined it as "a field of study that gives computers the ability to learn without being explicitly programmed". Later, this definition was formalized by Tom Mitchell as "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" (Mitchell, 1997).

Figure 1.2 illustrates the general process of a machine learning system. In order to learn, machine learning algorithms start from a set of examples originating from a certain domain. In the context of this thesis, the examples will consist of textual data, e.g., sentences or documents. As an example, consider a set of product reviews on smartphones, in which the goal is to distinguish the positive reviews from the negative ones. First, some specific characteristics from these examples are extracted, called *features*. This can be seen as translating the examples into a format the algorithm can understand. For the smartphone review example, this could be a vocabulary of words, where for each review it is known which of the words from the vocabulary appear in a particular review. The *task* is now to devise a mapping from these features to the output. In terms of the example, this means finding out which combination of words leads to a positive (resp. negative) smartphone review. In machine learning, this

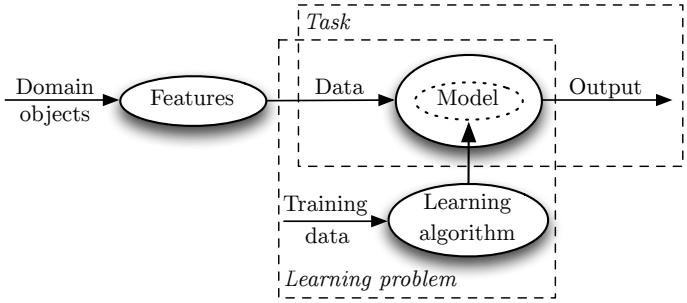


Figure 1.2: General process of a machine learning system (taken from Flach, 2012)

mapping is referred to as a *model*. In order to learn this model, a machine learning algorithm analyses a *training dataset*, which consists of examples for which the output is already known. This is referred to as the *learning problem*. Flach (2012) summarizes this process as follows: "machine learning is concerned with using the right features to build the right models that achieve the right tasks".

A *natural language* is commonly defined as a language that has evolved naturally as a means of communication among people and has developed through use rather than by prescription. It stands in contrast to formal or artificial languages, such as programming languages or mathematical logic. *Natural Language Processing* (NLP) is a subfield of artificial intelligence and linguistics concerned with the computational analysis of natural language. This can be as simple as separating a chunk of continuous text into separate words or as complex as automatically reading texts and answering questions about it.

Although using and learning language is relatively easy for humans, it is very hard for a computer. The goals of NLP are thus not easy to reach, since understanding language means identifying words and phrases that refer to higher-level concepts, interpreting these concepts and linking them together in a meaningful way. This requires one to solve a set of separate subproblems.

1.1.2 Computational Natural Language Learning

In the early days of NLP research, systems were based on symbolic techniques that required manual knowledge engineering, e.g., in the form of handcrafted production rules (Allen, 1988). This approach worked for small amounts of data (as illustrated by the SHRDLU system of Winograd (1972) that was mentioned earlier), but did not generalize to larger amounts of data.

The successful application of statistical methods on large amounts of data in speech recognition (Jelinek, 1976), which eventually caused a paradigm shift in the domain (Jelinek, 1997), combined with the limitations of the manual knowledge engineering approaches, led to the introduction of statistical methods in NLP. More advanced ML methods made their appearance and started to show promising results on tasks such as part-of-speech tagging and parsing, i.e., analyzing the structure of a sentence.

This gave rise to *Computational Natural Language Learning* (CoNLL) (Manning and Schütze, 1999) as the area of research that applies machine learning techniques to natural language understanding. The progress in this new domain was driven and accelerated by the increasing availability of additional (annotated) resources (e.g., the Penn Treebank (Marcus et al., 1993)). Furthermore, new techniques quickly entered the domain, and the increased availability of computational power facilitated their successful deployment.

1.1.3 Statistical Relational Learning

Many machine learning methods traditionally resort to a *propositional* or *attribute-value representation* of the examples they learn from. Intuitively, this can be seen as a table in which the columns represent the attributes, and the rows are descriptions of examples in terms of those attributes. The process of transforming a relational representation of a learning problem into a propositional one is referred to as *propositionalization* (Kramer et al., 2000). However, in many cases this propositionalization step results in information loss, and consequently suboptimal generalization during learning.

The need to deal with increasingly complex data that is structural or relational in nature, led to the consensus within the fields of artificial intelligence, and machine learning in particular, that both logical and relational representations were needed to solve many real-life applications. This insight resulted in an emerging new area of research known as *inductive logic programming* (ILP) (Muggleton and De Raedt, 1994), *multi-relational data mining* (Lavrač and Džeroski, 2001), or *logical and relational learning* (De Raedt, 2008). As the name indicates, ILP combines the inductive machine learning paradigm of learning from examples with first-order logical representations from logic programming, which are used to describe the data. Logical predicates allow one to represent the data in a relational format, as well as specify additional background knowledge declaratively.

In order to deal with uncertainty, which is often inherent in real-world applications (e.g., originating during data collection), the aforementioned logical and relational representations can be extended by probability theory. This gave

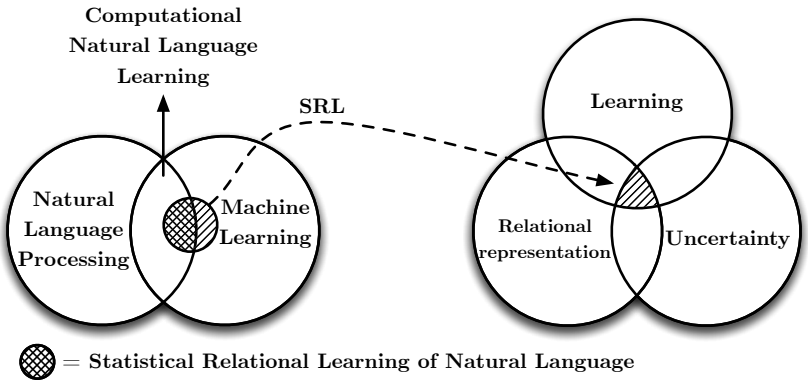


Figure 1.3: Situating statistical relational learning of natural language

rise to the field of *statistical relational learning* (SRL) (Getoor and Taskar, 2007), which is also known as *probabilistic inductive logic programming* (De Raedt and Kersting, 2004; De Raedt et al., 2008) or *probabilistic logic learning* (De Raedt and Kersting, 2003).

Today a large number of different representations and approaches exist for integrating logical, relational and probabilistic representations with machine learning. Prominent examples include Markov Logic (Richardson and Domingos, 2006), which integrates Markov networks with first-order logic, and systems such as PRISM (Sato and Kameya, 2001), ICL (Poole, 2000) and ProbLog (De Raedt et al., 2007), which integrate the logic programming language Prolog with probabilistic annotations. At the same time several applications of statistical relational learning in areas such as natural language processing, bioinformatics and social network analysis are being developed, and they often outperform alternative state-of-the-art methods. Domingos (2006) has argued that one of the reasons for this is that statistical relational learning models act as an *interface layer* for artificial intelligence because they allow one to combine and integrate various sources of information and knowledge.

We now have the necessary building blocks to describe the area of research in which this thesis is situated. This is graphically illustrated in Figure 1.3. Statistical relational learning of natural language is a research area at the intersection of computational natural language learning and statistical relational learning. We will now explain why combining these two domains is particularly interesting, and give an overview of the research questions of this thesis.

1.2 Motivation and Research Questions

Since the emergence of computational natural language learning more than two decades ago, there has been much progress. A number of promising results have been obtained on a wide variety of language learning tasks such as part-of-speech tagging and parsing. These tasks were mostly *syntactic* in nature, i.e., they mainly focused on the analysis of the *structural* relationships between words, with the goal of studying the way that words are arranged together.

Semantic analysis deals with giving words, phrases, sentences or paragraphs *meaning*, i.e., coupling them to elements and concepts in the real world. Examples of well-established semantic processing tasks are semantic role labeling and word sense disambiguation. This type of tasks still remains challenging because the broader context of words at sentence or discourse level has to be considered in order to account for aspects of meaning that are expressed by certain combinations of words. Often these tasks require one to combine syntactic with semantic dependencies, structured with unstructured data, local with global models, and probabilistic with logical information. It is unclear how to realize this using state-of-the-art language learning techniques, since most of these are rooted in the traditional local token-based approaches based on the lexico-syntactic features of individual words.

On the other hand, statistical relational learning, with its integrated approach to probability, logic and learning, may well be able to contribute to solving these problems. Language is full of relational structure, and SRL enables to exploit this knowledge. Furthermore, SRL allows one to compactly incorporate additional background knowledge about the domain. This is summarized by Bunescu and Mooney (2007) as:

> *"Natural language processing problems are particularly suited for statistical relational learning methods that combine the strengths of first-order predicate logic and probabilistic graphical models."*

The goal of this thesis is to *strengthen this observation and provide additional insights into how statistical relational learning techniques can be applied to natural language processing.* There are already a number of promising initial results in applying statistical relational learning to natural language, e.g., for information extraction (Bunescu and Mooney, 2007; Roth and Yih, 2007; Kok and Domingos, 2008) and semantic role labeling (Riedel and Meza-Ruiz, 2008). While these results definitely show the promise of SRL for language learning, we still lack a deep understanding of the possibilities and limitations of statistical relational learning, and we are still far away from the routine application of SRL techniques to language learning.

This overall goal of studying how statistical relational learning can contribute to natural language learning leads to the following specific research questions.

The first research question focuses on the specific characteristics of statistical relational learning, namely the ability to represent structured data in a relational representation and the declarative approach that offers the opportunity to specify additional background knowledge. As indicated before, these properties are particularly interesting in the context of natural language. Current semantic processing tasks require the integration of lower-level features (e.g., the lexico-syntactic features at word level) and higher-level (syntactical) structure and (semantic) relations. This research question can thus be summarized as

Q1 *What are the advantages of the relational representation and the declarative approach, as offered by SRL, for natural language learning?*

As indicated before, we are still far away from a routine application of SRL for natural language learning. In order to achieve the goal of SRL as an interface layer for artificial intelligence, a first step is to come to SRL frameworks that abstract away from the underlying details and offer a fully declarative specification of the learning problem. In natural language learning, a lot of attention needs to be given to the modeling of the task. The process of modeling a task in a relational representation that can be used as input to an SRL framework is often seen as an obstacle to applying SRL methods, with the consequence that one often resorts to the traditional propositional methods. This forms the subject of the following research question.

Q2 *Can an SRL system be tailored for use in natural language learning, in order to achieve a full relational learning framework for NLP?*

The high-level declarative specification of the domain, and the ability to encode additional background knowledge using logical predicates as declarative features, offers an increased expressivity of the model and interpretability of the results, especially when contrasted with traditional propositional approaches. This functionality proves particularly useful in the context of natural language learning, and its increased expressivity and interpretability can lead to new insights. This requires a way to calculate the importance of these high-level declarative features. Whereas this problem is well-studied for the propositional case, it is not yet well understood in the context of statistical relational learning. This leads to the following research question:

Q3 *How can the importance of declarative, relational features be assessed in the case of statistical relational learning?*

Relations not only form the input, but are often also part of the output of natural language learning tasks. One of these tasks is information extraction from structured and unstructured documents. Recently, this task has attracted a lot of attention in the context of Machine Reading, i.e., the automatic extraction of knowledge from text. In this respect, several systems have been developed for extracting relations from words or phrases from text. Furthermore, these relations often have a probability attached that represents the confidence of the method in the extracted relation. Due to its probabilistic reasoning capabilities, statistical relational learning could provide an added value here, which is the subject of the last research question:

Q4 *How can statistical relational learning be leveraged to reason about and extend probabilistic relational information in the context of Machine Reading?*

Answering these research questions resulted in the following contributions.

1.3 Contributions

The **main contribution** of this thesis is a set of new insights into how statistical relational learning can be advantageous for natural language learning. We will now give a detailed overview of the specific contributions according to the research questions outlined above.

The main contributions with respect to **Q1**, *what are the advantages of the relational representation and the declarative approach, as offered by SRL, for natural language learning*, are the following:

- An expressive and interpretable **relational representation** for natural language learning problems, and the use of graph kernel–based relational learning for two (semantic) processing tasks; hedge cue detection as a binary sentence classification task at the sentence level and the identification of evidence-based medicine categories as a multiclass multilabel classification task at the document level.
- An exploration of the importance of the **declarative approach** for the inclusion of contextual information in natural language processing tasks. This will be illustrated by means of declarative feature construction for including context at the sentence level (hedge cue detection) as well as at the document level (evidence-based medicine category identification).
- An **analysis of the influence** of the relational representation and declarative approach on the performance by a comparison of several

machine learning techniques along two dimensions: a propositional versus a relational representation for both lazy and eager learning. To this end we integrated the graph-based relational distance into a memory-based learner in order to attain a **relational memory-based learning** setting.

With regard to **Q2**, *can an SRL system be tailored for use in natural language learning, in order to achieve a full relational learning framework for NLP*, the following contribution was derived:

- The introduction of **kLogNLP**, a natural language processing module for kLog, that enriches it with NLP-specific preprocessors, and in this way enables the use of state-of-the-art libraries and toolkits within an elegant and powerful SRL framework. Furthermore a declarative, relational model that fits most natural language learning tasks is introduced. This results in a **full relational learning framework for NLP**.

This leads to **Q3**, *how can the importance of declarative features be assessed in the case of statistical relational learning*, with the following contributions:

- The introduction of **relational regularization**, which takes into account the relational structure of the domain during regularization. The relations are used to tie the parameters of a predictive linear model. In this way, one can assess the importance of the original elements in the relational model based on the resulting importance scores for the individual features.
- An embedded and a wrapper approach for **relational feature selection and ranking**, which allows one to get deeper insights into the relative importance of the elements in the relational model of the domain, i.e., the declarative features.
- An **empirical evaluation** on the aforementioned binary sentence classification task for hedge cue detection illustrates the use of relational regularization and feature ranking for natural language learning.

Finally, the contributions related to **Q4**, *how can statistical relational learning be leveraged to reason about and extend probabilistic relational information in the context of Machine Reading*, are:

- The exploration of **probabilistic rule learning to expand NELL**, the Never-Ending Language Learner, a knowledge base of probabilistic facts extracted from textual Web data. This takes place in the general

problem setting of inducing probabilistic logic programs from probabilistic examples. It combines the principles of the rule learner FOIL with ProbLog, a probabilistic Prolog.

1.4 Outline of the Dissertation

This thesis is organized into four main parts. **Part I** continues with an introduction of the necessary foundations on statistical relational learning and natural language processing. In addition, a thorough review of related work on statistical relational learning of natural language is provided. **Part II** focuses on graph kernel–based relational learning of natural language, whereas **Part III** turns to probabilistic rule learning for knowledge base expansion. **Part IV** concludes and outlines opportunities for future work. In the following, a brief overview of each of the parts is given.

Part I continues with an overview of relevant concepts and techniques from machine learning and natural language processing, and lays the foundations for the rest of this thesis (**Chapter 2**). Furthermore, **Chapter 3** will review related work in statistical relational learning of natural language, the interdisciplinary domain that is the focus of this research.

Part II focuses on graph kernel–based relational learning of natural language. The work in this part is rooted in kLog, a logical and relational language for graph kernel–based learning. Its graph-based relational representation will prove particularly useful for modeling natural language learning problems. The relational distance between examples is based on graph kernels, which enables one to incorporate the contextual information invaluable for natural language learning tasks. Furthermore, as an SRL framework, it offers a declarative approach, which allows one to specify additional background knowledge about the domain. **Chapter 4** introduces a relational representation for natural language learning problems in the context of kLog and illustrates the approach using hedge cue detection, a binary sentence classification task. In order to show the importance of the relational representation of kLog and its respective distance measure for natural language learning, a propositional and a relational representation are compared, both for lazy and eager learning. To this end, we also integrate the relational distance measure in a memory-based learner. In **Chapter 5**, we extend the relational representation to the document level and the learning problem to a multiclass multilabel sentence classification setting for identifying evidence-based medical categories. On this task, another advantage of the SRL approach will be illustrated, namely the inclusion of additional document context by means of declarative feature construction. In **Chapter 6**, kLogNLP is presented. Based on the models from the previous chapters, a general model that fits most language learning tasks is devised. It is part of a

natural language learning module for kLog, which results in a full relational learning framework for NLP. **Chapter 7** introduces relational regularization and feature ranking. These techniques were developed to gain deeper insights into the elements of the relational model, which is particularly interesting in the case of natural language learning. The chapters in this part are largely based on the following papers:

Mathias Verbeke, Paolo Frasconi, Vincent Van Asch, Roser Morante, Walter Daelemans, and Luc De Raedt. Kernel-based logical and relational learning with kLog for hedge cue detection. In Stephen H. Muggleton, Alireza Tamaddoni-Nezhad, and Francesca A. Lisi, editors, *Proceedings of the 21st International Conference on Inductive Logic Programming, Inductive Logic Programming, Windsor Great Park, UK, 31 July 2011 - 3 August 2011*, pages 347–357. Springer, 2012.

Mathias Verbeke, Vincent Van Asch, Walter Daelemans, and Luc De Raedt. Lazy and eager relational learning using graph-kernels. In Laurent Besacier, Adrian-Horia Dediu, and Carlos Martín-Vide, editors, *Proceedings of the Second International Conference on Statistical Language and Speech Processing, International Conference on Statistical Language and Speech Processing, Grenoble, France, 14-16 October 2014*. Springer, 2014. Accepted.

Mathias Verbeke, Vincent Van Asch, Roser Morante, Paolo Frasconi, Walter Daelemans, and Luc De Raedt. A statistical relational learning approach to identifying evidence based medicine categories. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP - CoNLL), Jeju Island, Korea, 13-14 July 2012*, pages 579–589. Association for Computational Linguistics, 2012.

Mathias Verbeke, Paolo Frasconi, Kurt De Grave, Fabrizio Costa, and Luc De Raedt. kLogNLP: Graph kernel-based relational learning of natural language. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Annual Meeting of the Association for Computational Linguistics (ACL), Baltimore, Maryland, USA, 22-27 June 2014*, pages 85–90. Association for Computational Linguistics, 2014.

Fabrizio Costa, Mathias Verbeke, and Luc De Raedt. Relational regularization and feature ranking. In *Proceedings of the 14th SIAM International Conference on Data Mining, SIAM International Conference*

on Data Mining, Philadephia, Pennsylvania, USA, 24-26 April 2014, pages 650–658. SIAM, 2014.

Part III explores the use of probabilistic rule learning for knowledge base expansion in the context of NELL. This problem is rooted in the induction of logic programs from probabilistic examples, which is the topic of **Chapter 8**. To this end, FOIL, a deterministic rule learner, is combined with ProbLog, a probabilistic Prolog. This part is largely based on the following paper:

Luc De Raedt, Anton Dries, Ingo Thon, Guy Van den Broeck, and Mathias Verbeke. Inducing probabilistic relational rules from probabilistic examples. Submitted.

Part IV contains the concluding chapter, in which the thesis is summarized and opportunities for future work are discussed.

Appendix B discusses work that was carried out during the course of this Ph.D., but which is not included in the main thesis text.

Chapter 2

Foundations

As outlined in the introduction, this thesis is situated at the intersection between two domains: statistical relational learning and natural language processing. In this chapter, we provide the relevant background knowledge on the concepts and techniques which we will build on in the rest of this thesis. In Section 2.2, we review the basic principles of machine learning in general, to continue with an introduction to statistical relational learning. Finally, Section 2.3 discusses the core concepts in natural language processing.

2.1 Machine Learning

In this section we review the basic principles and notation of machine learning that will be used and extended in the rest of this thesis. We first discuss classification, as one of the most common predictive machine learning scenarios, before continuing with the evaluation measures that will be used to evaluate the performance of the methods to be presented. The section concludes with a description of a number of machine learning techniques that were used as part of these methods. For additional details, Flach (2012) offers a thorough introduction to machine learning.

2.1.1 Classification

Machine learning problems can be structured into a taxonomy of which the two major branches distinguish between *supervised* and *unsupervised* learning problems. In supervised learning problems, the goal is to infer a *model* from a set of labeled training data, whereas unsupervised learning problems try to

extract the hidden structure in an unlabeled data set.

Supervised learning algorithms study a set of objects from a certain domain, called *instances*, each originating from an *instance space* $\mathcal{X}$. An instance $x \in \mathcal{X}$ is represented by a d-dimensional *feature vector* $x = (x^{(1)}, \ldots, x^{(d)})$. Each dimension of this vector is called a *feature*, and can either be continuous or discrete. An instance thus represents a point in a d-dimensional *feature space*. It is an abstract representation of an object in the dataset. Recall the smartphone review classification example from the introduction. In that case, the feature space consisted of a vocabulary of all words from all documents, whereas an individual feature represented the number of times a particular word appeared in a particular document.

Classification is one of the most common tasks in supervised predictive machine learning. It focuses on assigning a label or *class* to an example from a predetermined set of categories. The set of all instance labels defines the output space $\mathcal{Y}$. As it is a supervised classification task, the learning algorithm starts from a *training set* $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ where $x_i = (x_i^{(1)}, \ldots, x_i^{(d)}) \in \mathcal{X}$ and $y_i \in \mathcal{Y}$, consisting of n examples for which the labels are known. The assumption is that these instances are sampled independently from an unknown distribution $P(x, y)$. This is denoted by $(x_i, y_i) \overset{i.i.d.}{\sim} P(x, y)$, where *i.i.d.* stands for *independent and identically distributed*. The goal of classification is now to learn a discriminant function $f : \mathcal{X} \to \mathcal{Y}$ from a function family F, such that $f(x)$ predicts the true label y on unseen *test data* x, where $(x, y) \overset{i.i.d.}{\sim} P(x, y)$. This function is often referred to as the *classifier*, whereas the function family of possible classifiers is called the *hypothesis space*.

In order to estimate the quality of the classifier, a *loss function* ℓ is used to calculate the prediction error or *empirical risk* $\mathbb{E}_{(x,y)\sim P}[\ell(x, y, f(x))]$ on the training set. For classification, a frequently used loss function is the 0-1 loss, which calculates the number of misclassified cases: $\frac{1}{n}\sum_{i=1}^{n}(f(x_i) \neq y_i)$.

In sum, the classification problem can now be formalized as follows:

Given: A training set T, a function family of possible classifiers $F = \{f | f : \mathcal{X} \to \mathcal{Y}\}$ and a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$.

Find the classifier $\bar{f} \in F$ with the lowest prediction error $\mathbb{E}_{(x,y)\sim P}[\ell(x, y, \bar{f}(x))]$ on the training set, i.e.,

$$\bar{f} = \arg\min_{f \in F} \mathbb{E}_{(x\,y)\sim P}[l(x, y, f(x))] \tag{2.1}$$

This is often referred to as *empirical risk minimization*.

Regularization When the training set only contains a small number of examples or the hypothesis space is large, a possible risk is that the learning algorithm starts memorizing the training examples without being able to generalize to the unseen examples in the test set. This is referred to as *overfitting*. A way to deal with this problem is by enforcing simplicity constraints on the classifier, i.e., preferring simple classifier functions over more complex ones. This can be done by *structural risk minimization*, i.e., integrating a *regularization penalty* in the optimization function.

One way to enforce these simplicity constraints is to reduce the effective degrees of freedom of the classifier by tying together its parameters in some prescribed way. Consider the case of a linear classifier, with a function of the form

$$f(x) = \sum_{i=1}^{d} w_i x_i \tag{2.2}$$

In order to control the complexity of this function, it is now possible to constrain the values of w using a regularization function Ω. A few popular regularization penalties are the L_0 norm, which counts the number of non-zero w_i's, the L_1 norm, which is defined as $\sum_i |w_i|$ and the L_2 norm, which corresponds to the squared Euclidean norm of the weights, i.e., $\sum_i w_i^2$. In Chapter 7, we will study regularization for relational domains.

We can now define the *regularized risk* as the weighted sum of the empirical risk and the regularization penalty $\mathbb{E}(f) + \lambda\Omega(f)$, where $\lambda \geq 0$. The classification problem can now be reformulated as minimizing the regularized risk, i.e.,

$$\bar{f} = \arg\min_{f \in F} \mathbb{E}(f) + \lambda\Omega(f) \tag{2.3}$$

Classification tasks are often distinguished based on the number of possible labels in the output space $\mathcal{Y}$. When it only contains two possible classes, i.e., $y \in \{-1, +1\}$, the problem is referred to as a *binary classification task*, whereas if the output space contains multiple labels, the classification task becomes a *multiclass classification problem*. Furthermore, if an instance can be assigned more than two labels at the same time, the task is referred to as a *multiclass multilabel classification problem*. Both binary and multiclass multilabel classification will be considered in this thesis.

We will now turn to a number of statistical machine learning algorithms that can be used to solve this classification problem.

2.1.2 Statistical Learners

In order to solve the classification task above (and other machine learning problems in general) a number of different approaches can be taken. Depending on the task at hand and the available input data, an appropriate learning algorithm needs to be chosen. We will now discuss a number of (statistical) learning algorithms that will be used in the context of this thesis.

k-Nearest Neighbors

Instance-based or *memory-based learning* (MBL) is a family of learning algorithms that refrains of performing explicit generalization by learning a model. Instead, it stores a particular representation of the training data in memory, and classifies unseen data points by comparing them with these stored instances and extrapolating from the most similar stored cases. It exists under different variants and is referred to by a variety of names such as memory-based, exemplar-based, case-based and lazy learning, depending on the context (Stanfill and Waltz, 1986; Aha et al., 1991; Cost and Salzberg, 1993; Kolodner, 1992). Due to the lack of performing explicit generalization, these algorithms are categorized as *lazy learning* methods.

The most well-known instantiation of this class of algorithms, which formed the foundation for a number of related techniques is the k-nearest neighbor (kNN) algorithm (Fix and Hodges, 1951). It can be summarized as follows:

Given: A training set $T = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, a distance function δ, the number of neighbors k, and a test instance x^*.

1. **Find** the k training instances $x_{i_1}, \ldots, x_{i_k}$ closest to x^* according to distance δ.
2. **Output** y^* as the majority class of class labels $y_{i_1}, \ldots, y_{i_k}$, for which ties are broken randomly.

When the examples are represented in propositional format, one of the most frequently used distances between two examples is the Euclidean distance. *Distance-weighted kNN* (Dudani, 1976) also takes the distance into account during classification. The closer an example is to the instance to be classified, the more its vote counts. In Chapter 4, the integration of a *relational* distance measure into the kNN algorithm will be studied.

The algorithm is graphically illustrated in Figure 2.1 on a small dataset of 11 training examples from 2 different classes, black and white, for $k = 3$ and $k = 5$. The inner circle with small dashes includes the 3 nearest neighbors of the unseen

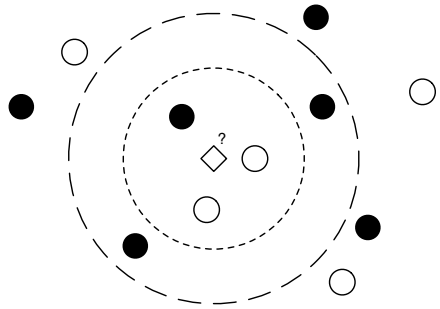


Figure 2.1: Example of kNN classification for $k = 3$ (inner circle, - -) and $k = 5$ (outer circle, — —)

example, represented by a diamond. As two out of the three training examples are white, the unseen example will be classified as white. When the number of nearest neighbors is set to 5, the unseen example will be classified as black, as three out of the five training examples are black (outer circle with big dashes).

Support Vector Machine

Whereas kNN was an example of a lazy learning algorithm, *Support Vector Machines* (SVM) (Cortes and Vapnik, 1995) are one of the most prominent examples of *eager* learning algorithms to date. The simplest case to calculate such a model appears when the data to classify is linearly separable, i.e., it is possible to draw a straight line that separates the examples of each class. The resulting model is a linear function of the form:

$$f(x; w, b) = \sum_{i=1}^{d} w_i x^{(i)} + b = w^\top \cdot x + b \tag{2.4}$$

where $x \in \mathbb{R}^d$ is a vector input in the d-dimensional feature space and $w \in \mathbb{R}^d$ is the associated parameter vector. The model is linearly separable if a w and b can be found such that:

$$sgn(f(x_i; w, b)) = y_i \qquad \forall 1 \geq i \geq n \tag{2.5}$$

However, in most cases, the ranges of w and b allow many possible lines to be drawn. The goal is now to find the parameter settings such that the resulting line intuitively corresponds to a line that runs evenly between the data. This

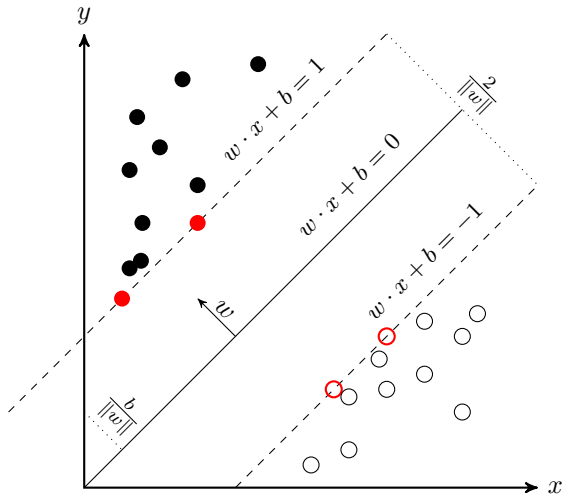


Figure 2.2: Support Vector Machine in a two-dimensional feature space

forms the regularization criterion for SVMs which is known as finding a *maximal margin hyperplane*.

In the case of binary classification, the following set of equations follows from Equation 2.5:

$$y(x) = \begin{cases} +1, & \text{if } w^\top \cdot x + b > 0, \\ -1, & \text{if } w^\top \cdot x + b \leq 0. \end{cases} \tag{2.6}$$

This now allows to derive the geometric width of the margin between the positive and the negative examples, which corresponds to $\frac{2}{||w||}$. Maximizing the margin thus equals to minimizing $\frac{1}{2}||w||^2$, which results in the following optimization problem:

$$\begin{aligned} &\min_{w,b} \frac{1}{2}||w||^2 \\ &\text{s.t. } y_i(w^\top \cdot x_i + b) \geq 1 \quad \forall i \end{aligned} \tag{2.7}$$

Figure 2.2 illustrates a support vector machine for a two-dimensional feature space. The points located on the margin are called the *support vectors*. These are indicated in red in the figure.

Until now, only the case in which the data is perfectly linearly separable has been considered. When this is not the case, *slack variables* ξ can be introduced that measure the misclassification rate of these noisy data points. This allows a number of data points to be inside the margin or at the wrong side of the decision boundary. These points are referred to as the *margin errors*. Equation 2.7 can be updated accordingly to the *soft margin* optimization problem as follows:

$$\begin{aligned} &\min_{w,b,\xi_i} \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\xi_i \\ &\text{s.t. } y_i(w^\top \cdot x_i + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0 \quad \forall i \end{aligned} \tag{2.8}$$

The hyperparameter C represents the *cost of the error term*, i.e., it controls the amount of training error allowed. A large value of C thus results in less training error and smaller margins but an increased generalization error, as the model becomes more complex.

Kernel Trick Until now, we have focused on the optimization problem of finding a maximum margin separating hyperplane for input data drawn from a finite-dimensional feature space $\mathbb{R}^d$. However, in some cases the data in $\mathbb{R}^d$ is not linearly separable, or comes from a discrete space. The latter is a common problem in NLP, where the feature space often consists of the space of all words, or in case the data is structured, the space of all (parse) trees or graphs.

The *kernel trick* (Mercer, 1909; Boser et al., 1992) offers a solution to this problem. To illustrate its principle, consider the feature space in Figure 2.3 (left) in which the data is not linearly separable. By using a feature mapping Φ, the feature space can be mapped to a higher-dimensional (and possibly infinite) one in which the data is linearly separable, as shown in the right part of the figure. One way to solve the optimization problem in this new feature space is to explicitly calculate the feature mapping $\Phi(x)$. However, to reduce the computational complexity of this explicit calculation, one can also leave this computation implicit by means of a *kernel function* $\kappa(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$. It implicitly computes the dot product between x_i and x_j in the new feature space without explicitly transforming them to points in this feature space. This enables efficient computation of nonlinear decision boundaries for SVMs. To this end, we will now turn to a brief explanation of the mathematical intuition behind the kernel trick.

The goal is to convert the optimization problem of Equation 2.8 into an equivalent optimization problem that does not need the explicit feature mapping $\Phi(x)$, and consequently avoids the computation of a (possibly infinite) weight vector w in this space. This can be solved using *Lagrange multipliers* α, a method

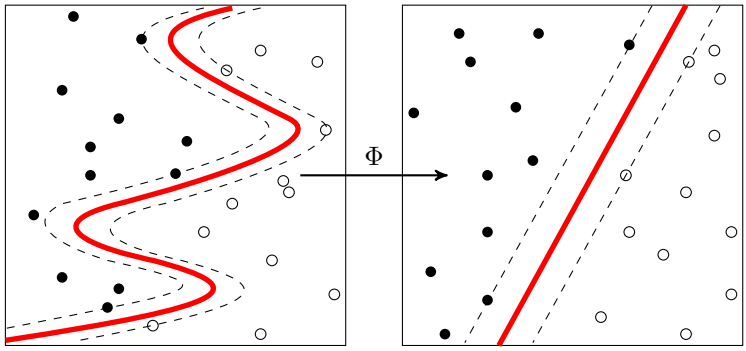


Figure 2.3: The kernel trick: by means of feature map Φ, the feature space can be transformed such that the data becomes linearly separable (inspired by Flach, 2012).

from optimization theory to find local minima or maxima of a function that is subject to equality constraints, as follows:

$$\alpha_1^*, \ldots, \alpha_n^* = \arg\max_{\alpha_1,\ldots,\alpha_n} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i \cdot x_j + \sum_{i=1}^{n}\alpha_i \quad \text{s.t. } \alpha_i \geq 0 \text{ and } \sum_{i=1}^{n}\alpha_i y_i = 0 \tag{2.9}$$

Whereas Equation 2.8 is called the *primal* optimization problem, this is called the *dual* optimization problem. As this is defined in terms of a dot product between the instances and given that $\kappa(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$, this leads to the following optimization problem in the high-dimensional feature space $\Phi(x)$:

$$\alpha_1^*, \ldots, \alpha_n^* = \arg\max_{\alpha_1,\ldots,\alpha_n} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \kappa(x_i, x_j) + \sum_{i=1}^{n}\alpha_i \quad \text{s.t. } \alpha_i \geq 0 \text{ and } \sum_{i=1}^{n}\alpha_i y_i = 0 \tag{2.10}$$

Kernels Several possible kernels exist. We will discuss three of the most common ones that were used in the context of this thesis.

The *linear kernel* is the simplest one, as the feature space is exactly the same as the input space. It is defined as:

$$\kappa(x_i, x_j) = x_i \cdot x_j \tag{2.11}$$

Polynomial kernels represent a feature space over polynomials of the original instances. A polynomial kernel of degree ρ is defined as

$$\kappa(x_i, x_j) = (\nu + \psi x_i \cdot x_j)^\rho \tag{2.12}$$

with constants ν and ψ as hyperparameters.

Finally, the *Gaussian* or *radial basis function* (RBF) kernel is defined as

$$\kappa(x_i, x_j) = exp\left(\frac{-||x_i - x_j||^2}{2\sigma^2}\right) \tag{2.13}$$

Intuitively this kernel draws circles around the instances, where the hyperparameter σ corresponds to the size of these circles.

These were all kernels over $\mathbb{R}^d$. There also exist kernels over discrete spaces. In Section 2.2.3, we will discuss the Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) (Costa and De Grave, 2010), which is the graph kernel used in the context of the SRL framework kLog.

Hidden Markov Support Vector Machine

The SVMs introduced in the previous section can now be generalized to *structured SVMs* (Tsochantaridis et al., 2004, 2005), for which not only the input space, but also the output space can be multivariate or structured. It enables to predict complex objects such as sets, sequences, or graphs.

One instantiation is the *Hidden Markov Support Vector Machine* (HM-SVM or SVM-HMM) (Altun et al., 2003), a structured SVM for sequence tagging. In contrast to regular SVMs, where for each instance x_i a single output value y_i is predicted, SVM-HMM allows to predict a label sequence $\mathbf{y} = (y_1, \ldots, y_m)$ for a given input sequence of feature vectors $\mathbf{x} = (x_1, \ldots, x_m)$. Each label is taken from a label set Υ, i.e., $y_i \in \Upsilon$. The output space $\mathcal{Y}$ thus consists of all possible label sequences.

Instead of learning a direct mapping from $\mathcal{X}$ to $\mathcal{Y}$, a discriminant function in structured output learning assigns a score to each input-output pair, which

leads to the following function family F of possible discriminant functions: $F : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. The optimal w-parametrized discriminant function $\bar{f}$ can then be found by maximizing this score over all possible output structures:

$$\bar{f}(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}, \mathbf{y}; w) \tag{2.14}$$

As F represents a set of functions in some *combined feature representation* of inputs and outputs, this requires a mapping $\Phi(\mathbf{x}, \mathbf{y})$ to extract features from pairs of sequences of input vectors and label sequences $(\mathbf{x}, \mathbf{y})$. The goal is to find an F that is linear in this combined feature representation, i.e.,

$$F(\mathbf{x}, \mathbf{y}; w) = w \cdot \Phi(\mathbf{x}, \mathbf{y}) \tag{2.15}$$

An efficient computation is possible by avoiding the explicit mapping Φ, which is possible when a kernel κ exists such that

$$\kappa((\mathbf{x}_i, \mathbf{y}_i), (\mathbf{x}_j, \mathbf{y}_j)) = \Phi(\mathbf{x}_i, \mathbf{y}_i) \cdot \Phi(\mathbf{x}_j, \mathbf{y}_j) \tag{2.16}$$

When the output consists of label sequences, inspiration for this combined feature representation can be found with Hidden Markov Models (HMM), which suggest defining two kinds of features. A first kind are the *emissions*, i.e., the dependencies between the input vectors and a specific label, whereas a second are the *transitions*, i.e., the dependencies between neighboring labels in the Markov chain.

Given a training set $T = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$ of input-output sequences, this leads to the following linear discriminant function:

$$\bar{f}(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} w_{emis}^{\top} \cdot \Phi_{emis}(\mathbf{x}, \mathbf{y}) + w_{trans}^{\top} \cdot \Phi_{trans}(\mathbf{y}) \tag{2.17}$$

in which w_{emis}, w_{trans}, Φ_{emis} and Φ_{trans} are the respective emission and transition weight and feature vectors. The maximum margin optimization problem in terms of the combined feature representation $\Phi(\mathbf{x}, \mathbf{y})$ is now defined as:

$$\begin{aligned} &\min_{w, b_i, \xi_i} \frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \xi_i \\ &\text{s.t. } z_i(\mathbf{y})(w \cdot \Phi(\mathbf{x}_i, \mathbf{y}) + b_i) \geq 1 - \xi_i \text{ and } \xi_i \geq 0 \quad \forall i, \forall \mathbf{y} \in \mathcal{Y} \end{aligned} \tag{2.18}$$

in which the L_1 norm is used as the regularization penalty, and $z_i(\mathbf{y})$ is defined as

$$z_i(\mathbf{y}) = \begin{cases} 1, & \text{if } \mathbf{y} = \mathbf{y}_i, \\ -1, & \text{otherwise.} \end{cases} \tag{2.19}$$

For more details, the interested reader is referred to (Altun et al., 2003).

2.1.3 Evaluation Measures

Machine learning algorithms are evaluated using a wide variety of evaluation measures. We will limit ourselves to a discussion of the evaluation measures that will be used in the rest of this thesis.

Contingency Table

The performance of classifiers can be summarized by means of a *contingency table* or *confusion matrix*. This is graphically represented in Table 2.1. In this contingency table, the rows represent the actual classes, whereas the columns represent the predictions of the classifier. The last row and column in this table represents the *marginals*, i.e., the row and column sums. The following terms will be used to describe the performance at the class level; positives and negatives that are correctly predicted are referred to as *true positives (tp)* and *true negatives (tn)* respectively. Positives that are incorrectly classified as negatives are called *false negatives (fn)*, whereas negatives that are incorrectly classified as positives are referred to as *false positives (fp)*.

	Predicted $\oplus$	*Predicted* $\ominus$	*Marginals*
Actual $\oplus$	True Positives (tp)	False Negatives (fn)	$pos = tp + fn$
Actual $\ominus$	False Positives (fp)	True Negatives (tn)	$neg = fp + tn$
Marginals	$tp + fp$	$fn + tn$	$tp + tn + fp + fn = n$

Table 2.1: Contingency table

We will now turn to a number of evaluation measures that can be defined in terms of these numbers.

Accuracy

One of the simplest evaluation measures is *accuracy*, which is defined as the proportion of correctly classified test instances, or formally:

$$acc = \frac{tp + tn}{pos + neg} \tag{2.20}$$

Despite its simplicity, the downside of accuracy is that the more prevalent class contributes more strongly. Particularly when the dataset is highly unbalanced (i.e., when there are considerably more positive then negative examples or vice versa), accuracy may give the impression that the classifier does a good job in classifying the instances when this is not the case. For example, for a dataset of 1000 bank accounts, of which 10 are considered fraudulent, a classifier that classifies every bank account as non-fraudulent will obtain an accuracy of $\frac{990}{1000} = 0.99$, or 99%. Although the high accuracy score, this is not the kind of classifier a bank manager would like to have. For unbalanced datasets, where the minority class is of interest, accuracy gives a distorted picture of the results. We will now turn to a number of alternative evaluation measures that give a more detailed view on the results.

Precision, Recall and $F1$ measure

Precision and *recall* are two measures originating from the field of information retrieval, referring to the fraction of retrieved documents that are relevant to the search and the fraction of the documents that are relevant to the query that are successfully retrieved respectively. Currently, these measures are more broadly used to measure the performance of a classifier. They are defined as follows:

$$precision = \frac{tp}{tp + fp} \tag{2.21}$$

$$recall = tpr = \frac{tp}{tp + fn} \tag{2.22}$$

Recall is equivalent to the *true positive rate* (tpr), i.e., the proportion of predicted positives among the actual positives. Similarly, the *false positive rate* (fpr) is defined as the proportion of the negatives incorrectly classified as positives among the actual negatives.

$$fpr = \frac{fp}{fp + tn} \tag{2.23}$$

The F-measure, F_β, was introduced in information retrieval by Van Rijsbergen (1979) to measure the "effectiveness of retrieval with respect to a user who attaches β times as much importance to recall as precision". It is defined as

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{2.24}$$

The commonly used $F1$ *score* is the harmonic mean of precision and recall, obtained by setting β to 1:

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{2.25}$$

We will use the latter score when talking about the F-score in the rest of this thesis.

In order to visualize the performance of a binary classifier, one can use a *receiver operating characteristic*, or ROC curve. It plots the true positive rate against the false positive rate.

Macro- and Micro-average

The aforementioned evaluation measures give results for a single class. When the classification problem consists of multiple classes $\Upsilon = \{\upsilon_i : i = 1, \ldots, q\}$, these results can be averaged over all classes in order to get a view on the overall results. Two commonly used techniques are *macro-* and *micro-averaging*.

Macro-averaging If we let B denote a binary evaluation measure and tp_υ, tn_υ, fp_υ, and fn_υ the true positives, true negative, false positives and false negatives after binary evaluation for label υ, then the macro-averaged score (Tsoumakas et al., 2010) for B is defined as:

$$B_{macro} = \frac{1}{q} \sum_{\upsilon=1}^{q} B(tp_\upsilon, tn_\upsilon, fp_\upsilon, fn_\upsilon) \tag{2.26}$$

As the label set for the training and test data can differ (if there are no instances for a particular class label in the test set), in most cases, the set of class labels from the training data is used to calculate the macro-averaged score.

Micro-averaging Whereas the macro-averaged score weights each class equally, the micro-averaged score (Sebastiani, 2002) joins classification decisions across classes, after which the measure is calculated on the joint contingency table. It is defined as follows:

$$B_{micro} = B\left(\sum_{v=1}^{q} tp_v, \sum_{v=1}^{q} tn_v, \sum_{v=1}^{q} fp_v, \sum_{v=1}^{q} fn_v\right) \tag{2.27}$$

As discussed by Sebastiani (2002), there is no agreement on which one of the two averaging techniques should be used. Some believe that micro-averaging is misleading since more frequent classes are weighted heavier in the average, whereas others believe that classes should be counted proportionally to their frequency. As large classes dominate small classes in micro-averaging, Manning et al. (2008) argues to use macro-averaged scores to get an idea of the performance on small classes.

Although the discussion above focused on binary classification, note that all of the aforementioned evaluation measures and averaging techniques can be generalized to *multi-class* and *multi-label classification problems*, i.e., tasks where the label set consists of more than two classes and where each example can have multiple classes respectively.

2.2 Statistical Relational Learning

As mentioned in the introduction, traditional machine learning methods (e.g., the statistical learners introduced in Section 2.1.2) assume a *propositional representation* of the data. It consists of fixed-length vectors of attribute-value pairs in which each vector represents an example, and each element of the vector represents the value for a particular attribute. A set of examples can thus be represented as a table in which the rows represent the examples that are described in terms of a set of attributes in the columns.

While a lot of successful machine learning systems have been developed which rely on a propositional representation, gradually the need increased to represent more complex, relational data. One approach to take this relational representation into account is *propositionalization* (Kramer et al., 2000), i.e., translating the relational structures into a propositional format that can still be used as input for propositional learners. Despite the advantage of being able to reuse the existing propositional techniques, this translation step also results in information loss (De Raedt, 2008). Since this affects the learning capabilities and generalization power of an algorithm, this motivated research into techniques that can represent multiple entities with their properties and the relationships among them.

This gave rise to the field of *Inductive Logic Programming* (ILP) (Muggleton and De Raedt, 1994) or *logical and relational learning* (De Raedt, 2008), in which *first-order logic* is used as an expressive formalism, both for representing the data as for reasoning on it. It allows one to express complex relational structures and at the same time is able to express additional background knowledge on the domain. *Statistical Relational Learning* (SRL) (Getoor and Taskar, 2007) combines ILP with statistical machine learning techniques and probability theory in order to deal with uncertainty, which is often inherently present in the data.

We will first introduce logic programming as a strong relational formalism on which SRL systems are built, and subsequently discuss two SRL frameworks that are central to this thesis. *kLog* is a language for logical and relational learning with graph kernel–methods. *ProbLog* is a probabilistic extension of Prolog that offers a suite of efficient algorithms for various inference tasks. For additional details, we refer the interested reader to De Raedt (2008) and Getoor and Taskar (2007) for a comprehensive overview of logical and relational learning and statistical relational learning respectively.

2.2.1 Logic Programming

Logic programming is a programming paradigm based on logic. In contrast to imperative programming languages such as Java, in which you describe a number of steps on *how* to solve the problem, logic programming languages are *declarative* in nature, i.e., you describe *what* the solution looks like, but not necessarily how to get there. In order to describe the solution, the search space in which the solution needs to be found is constrained by employing a so-called *declarative bias*. As the name indicates, the bias should be declarative, i.e., explicit and easily understandable for the user. To this end, logic programming uses first-order logic. Prolog is one of the most prominent logic programming languages, and will also be used in this thesis.

We will first introduce the syntax of clausal logic by means of the following example:

Example 2.1. *The following logic program represents the contents of a bag filled with blocks, which can be either cubes or spheres, each with a particular color and weight. Each block also has a identifier, which uniquely characterizes the object. The first part of the program enumerates all spheres and cubes in the bag. The last part of the program states that two cubes have the same color.*

```
sphere(s1,r,2).
sphere(s2,g,2).
sphere(s3,g,3).
```

```
sphere(s4,r,1).

cube(c1,g,1).
cube(c2,r,2).
cube(c3,g,4).
cube(c4,r,3).

cub_eq_col(CubeX,CubeY) ←
   cube(CubeX,Color,_),
   cube(CubeY,Color,_).
```

Example 2.1 contains a number of *constants*, such as `s1`, `c3` and `r`, as well as the *variables* `CubeX`, `CubeY` and `Color`. Both constants and variables are *terms*. If $t_1, ..., t_n$ are terms and a is an n-ary *predicate*, $a(t_1, ..., t_n)$ is an *atom*. Example 2.1 contains three predicates, namely `sphere`, `cube` and `cub_eq_col`. `sphere(s1,r,2)` is an example atom. It describes a red (`r`) sphere with identifier `s1` of a weight 2 kg. An atom is also referred to as a *positive literal*, whereas its negation, e.g., `not(sphere(s1,r,2))` is referred to as a *negative literal*. Note that Example 2.1 used the notational conventions of Prolog, i.e., variable names start with an uppercase letter, while all other syntactic entities start with a lowercase letter. In the rest of this thesis, the `typewriter font` will be used to refer to code snippets that should be read as Prolog code.

We are now able to define a *clause* as a disjunction of literals. When a clause contains only one positive literal, it is referred to as a *definite clause*, which in Prolog notation is denoted by

$$\texttt{h} \leftarrow b_1\texttt{,}\ldots\texttt{,}b_m$$

where the atom `h` is called the *head* and the atoms $b_1, ..., b_m$ are called the *body*. The arrow symbol ← separates the head (the defined atom) from the body (a condition which, if satisfied, makes the head true). All variables in a clause are universally quantified. If a definite clause has the body `true`, the body can be omitted. Such a clause is referred to as a *fact*, e.g., `cube(c1,g,1).` A *logic program* is defined as a finite set of definite clauses.

A term or clause is called *ground* if it does not contain variables. A *substitution* $\theta = \{V_1/t_1, \cdots, V_m/t_m\}$ maps variables V_i to terms t_i. Applying θ to an atom a, denoted as $a\theta$, means that all occurrences of V_i in a are replaced by t_i.

Example 2.2. *Applying* θ = {`Identifier/c5`, `Color/r`, `Weight/6`} *to* `cube(Identifier, Color, Weight)` *yields* `cube(c5,r,6)`

Two terms t_1 and t_2 are *unifiable* if there exist substitutions θ_1 and θ_2 such that $t_1\theta_1 = t_2\theta_2$. A substitution is the *most general unifier* $mgu(a, b)$ of atoms a and b if and only if $a\theta = b\theta$ and for each substitution θ' such that $a\theta' = b\theta'$, there exists a non-trivial substitution γ such that $\theta' = \theta\gamma$. A *non-trivial substitution* is a substitution that maps at least one variable to a term different from itself. Note that a most general unifier is not necessarily unique.

Example 2.3. *A most general unifier of the atoms* `sphere(Identifier,r,Weight)` *and* `sphere(Identifier,Color,W)` *is* $\theta = \{$`Color/r`, `W/Weight`$\}$

Now that we have defined the syntax of clausal logic, we can turn to its semantics. It is based on the concept of *interpretations*. We will describe one particular instance of interpretations, namely the *Herbrand interpretation*. The *Herbrand base* of a logic program is the set of all ground atoms that can be constructed by the terms in this logic program. Each subset of the Herbrand base is a Herbrand interpretation. Formally, a Herbrand interpretation I is a *model* of a clause $h \leftarrow b_1,...,b_m.$ if for every substitution θ with $b_i\theta \in I$, $h\theta$ is contained in I as well. The Herbrand interpretation is a model of the logic program if it is a model of all clauses in the program. The model-theoretic semantics of a logic program is given by its *least Herbrand model*, which is defined as the smallest Herbrand model of this logic program.

Example 2.4. *The Herbrand base of the logic program from Example 2.1 is*
{`sphere(s1,r,2)`, `sphere(s2,g,2)`, `sphere(s3,g,3)`, `sphere(s4,r,1)`,
`cube(c1,g,1)`, `cube(c2,r,2)`,`cube(c3,g,4)`, `cube(c4,r,3)`,
`cub_eq_col(c1,c3)`, `cub_eq_col(c3,c1)`, `cub_eq_col(c2,c4)`,
`cub_eq_col(c4,c2)`}
An example of a possible Herbrand interpretation is
$i = \{$`sphere(s1,r,2)`, `cube(c3,g,4)`, `cub_eq_col(c3,c1)`$\}$.

2.2.2 Relational Learning

The goal of logical and relational learning is now to find a hypothesis h, i.e., a logic program, from a set of positive and negative examples. Analogous to the learning problem for classification as defined in Section 2.1.1, the logical and relational learning problem can be formalized as follows:

Given: A set of training examples T over a language $\mathcal{L}_E$, a background theory B, a hypothesis language $\mathcal{L}_h$ that specifies the clauses that are allowed in the hypotheses, and a relation $covers(e, H, B)$ which determines the classification of an example e with respect to H and B.

Find a hypothesis $h \in \mathcal{H}$ that covers all positive training examples and none of the negative ones with respect to background theory B.

The specific learning setting is determined by $\mathcal{L}_E$, the language chosen for representing the examples, together with the *covers* relation (De Raedt, 1997). Two of the most popular learning settings are *learning from entailment* (Plotkin, 1970) and *learning from interpretations* (De Raedt and Džeroski, 1994).

Learning from entailment In learning from entailment, the examples are definite clauses. A hypothesis h covers an example e with respect to the background theory B if and only if $H \cup B \models e$.

Learning from interpretations In learning from interpretations, the examples are Herbrand interpretations. A hypothesis h covers an example e with respect to the background theory B if and only if e is a model of $B \cup H$.

Whereas in learning from entailment, an example can consist of just a single fact, in learning from interpretations all facts that hold in the example are known. In the latter setting, typically more information is available to the learner. In Part II, a learning from interpretations setting will be adopted, whereas in Part III, we will upgrade learning from entailment to a probabilistic setting. We will now discuss kLog and ProbLog, the two SRL frameworks that will be used in these respective parts.

2.2.3 kLog

kLog[1] (Frasconi et al., 2014) is a language for logical and relational learning with kernels. It is embedded in Prolog, and builds upon and links together concepts from database theory, logic programming and learning from interpretations. As an SRL framework, kLog allows one to declaratively represent both data, background knowledge and the problem description in a concise and high-level fashion. In this sense it is related to standard ILP systems such as Progol (Muggleton, 1995) and Tilde (Blockeel and De Raedt, 1998). In contrast to traditional SRL systems, such as Markov Logic (Richardson and Domingos, 2006) or Bayesian Logic Programs (Kersting and De Raedt, 2007), which combine probabilistic (graphical) models with logical and relational representations, kLog uses a kernel-based approach. It thus does not directly represent a probability distribution. In contrast, kLog employs a logical and relational data representation rooted in the entity/relationship (E/R) model (Chen, 1976) originating from database theory. For each interpretation, kLog computes the corresponding *graphicalization*, that is, it declaratively converts instances into graphs. In practice, the graphicalization corresponds to the *unrolled* E/R model

[1] http://klog.dinfo.unifi.it/

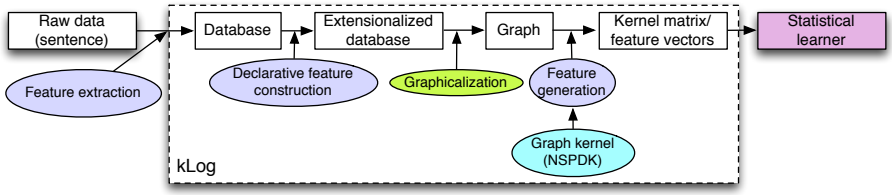


Figure 2.4: General kLog workflow

of the example, which lists all relations that hold in the example. Using these graphicalized interpretations, a new feature space can be defined using graph kernels. In this new (high-dimensional) feature space, the learning of a (linear) statistical model takes place. kLog's workflow is depicted in Figure 2.4. We will now discuss each of the steps in more detail by means of the following artificial example.

Example 2.5. *Consider the following artificial game: equipped with a bag filled with cubes and spheres as described in Example 2.1, the game consists of drawing a random sequence of 5 objects from the bag. The game can only be won if two* spheres *with the same color are drawn without drawing two* cubes *with the same color. In any other case the game is lost.*

Data Modeling

kLog employs a learning from interpretations setting. As introduced before, each interpretation is a set of tuples that are true in the example. kLog assumes a *closed world*, which means that atoms that are not known to be true, are assumed to be false. An example interpretation for one particular instance of the artificial game is shown in Listing 2.1.

```
1 sphere(s1,r,2).                    next_SC(s1,c1).
2 cube(c1,g,1).                      next_SC(s2,c1).
3 sphere(s2,g,2).                    next_S(s2,s3).
4 sphere(s3,g,3).                    next_SC(s3,c2).
5 cube(c2,r,2).
6 diff_shape_eq_weight(s1,c2).
7 sph_eq_col(s2,s3).
8 diff_shape_eq_weight(s2,c2).
```

Listing 2.1: Artificial example: example interpretation for one particular game.

Since kLog is rooted in database theory and each interpretation can be seen as an instance of a (small) relational database, the modeling of the problem domain is done using an entity/relationship model. E/R modeling is a commonly used

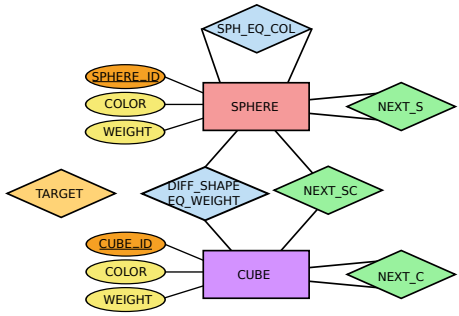


Figure 2.5: Artificial example: E/R diagram modeling the domain

design principle from database theory that is tailored to represent the domain at hand. An E/R model gives an abstract representation of the interpretations. As the name indicates, E/R models consist of *entities* and *relations*. Both entities and relations can have several *attributes*. Some of the attributes are denoted as *key attributes*, and uniquely identify an instance of an entity. Two additional database assumptions are made to ensure that the the subsequent graphicalization procedure is well-defined. First, it is required that the primary key of every relation consists solely of the identifiers of the entities that are involved in the relation. As a special case, relations of zero relational arity (i.e., relations for which the length of the primary key is zero) are allowed. This kind of relations is useful to represent global properties of an interpretation. Second, for every set of entities, there needs to be a distinguished relation of relational arity 1 (to represent the primary key of the entity).

The E/R diagram for the artificial game is shown in Figure 2.5. It contains two entities[2] (i.e., `sphere` and `cube`), which each have a key attribute (i.e., `sphere_id` and `cube_id` respectively) and two regular attributes, namely `color` and `weight`. For example, `sphere(s1,r,2)` represents a red sphere with identifier `s1` and a weight of 2 kg. Several relations exist between the entities. The `next` relations (i.e., `next_s`, `next_c` and `next_sc`) represent the sequence in which the entities are drawn from the bag during the game. For example, a `next_s` relation is present between spheres if they are drawn consecutively from the bag. The relation `sph_eq_col` indicates that two spheres have the same color, whereas the `diff_shape_eq_weight` relation indicates that a sphere and a cube are of different shape, but have an equal weight. The `target` relation is

[2]Note that one could also model spheres and cubes as a single object entity with an additional attribute that represents the shape. However, in order to better illustrate the different concepts, here we used two different entities. Similarly, different relations were used to model the sequence in which the entities are drawn from the bag.

the one that needs to be predicted by the classifier, and represents if a game is won or lost.

The E/R model can now be coded declaratively in kLog using an extension of the logic programming language Prolog. Every entity or relation that will be used in a later step to generate features is declared with the keyword `signature`. This resembles the declarative bias as used in logic programming. There are two types of signatures: *extensional* and *intensional* ones. For extensional signatures, all ground atoms have to be listed explicitly in the database, whereas intensional signatures implicitly define ground atoms using Prolog definite clauses (i.e., *declarative feature construction*). Intensional signatures are mostly used to introduce *additional background knowledge* on the domain in the learning process. The latter is one of the key characteristics of kLog, and one of the major strengths of SRL systems in general. Subsequently, the intensional predicates are grounded. This is a process similar to materialization in databases, that is, the atoms implied by the background knowledge and the facts in the example are all computed using Prolog's deduction mechanism. This leads to the *extensionalized database*, in which both the extensional as well as the grounded intensional predicates are listed.

The declarative kLog model for the artificial game can be found in Listing 2.2. As can be seen, a signature is characterized by a name and a list of typed arguments. There are three possible argument types. First of all, the type can be the name of an entity set which has been declared in another signature (e.g., the `next_s` signature represents the sequence relation between two entities of type `sphere`). The type `self` is used to denote the primary key of an entity. An example is `sphere_id` (line 2), which denotes the unique identifier of a certain sphere in the interpretation. The last possible type is `property`, in case the argument is neither a reference to another entity nor a primary key (e.g., `color` in line 2). Similar to the extensional signature `sph_eq_col` (lines 16-19), which represents the relation between two spheres of equal color, we also defined the relation between two cubes of equal color, `cub_eq_col` by means of an intensional signature.[3]

Graphicalization and Feature Generation

In this step, a technique called *graphicalization* transforms the relational representations from the previous step into graph-based ones and derives features from a grounded entity/relationship diagram using graph kernels. This can be interpreted as unfolding the E/R diagram over the data. Each interpretation is converted into a bipartite graph, for which there is a vertex for every ground

[3]Note that also the `diff_shape_eq_weight` relation could be defined by means of an intensional signature.

```
begin_domain.
signature sphere(sphere_id::self,color::property,weight::property)
    ::extensional.

signature cube(cube_id::self,color::property,weight::property)::
    extensional.

signature next_s(sphere1_id::sphere,sphere2_id::sphere)::
    extensional.

signature next_c(cube1_id::cube,cube2_id::cube)::extensional.

signature next_sc(sphere_id::sphere,cube_id::cube)::extensional.

signature sph_eq_col(sphere1_id::sphere,sphere2_id::sphere)::
    extensional.

signature diff_shape_eq_weight(sphere_id::sphere,cube_id::cube)::
    extensional.

signature cub_eq_col(cube1_id::cube,cube2_id::cube)::intensional.
cub_eq_col(CubeX,CubeY) ←
     cube(CubeX,Color,_),
     cube(CubeY,Color,_).

end_domain.
```

Listing 2.2: Artificial example: declarative kLog model for the artificial game.

atom of every E-relation, one for every ground atom of every R-relation, and an undirected edge $\{e, r\}$ if an entity e participates in relationship r. In contrast to existing propositionalization approaches, graphicalization does not transform the data into an attribute-value format but rather into a graph-based one. This enables the use of graph kernels in the subsequent feature generation step, due to which the full relational representation can be taken into account. Consequently, the resulting feature vectors in a high-dimensional feature space are much richer than most of the other direct propositionalization techniques. Graphicalizations for three instances of the artificial game are shown in Figure 2.6. The top figure depicts the graphicalization of the interpretation of Listing 2.1. Note that for the sake of clarity, attributes of entities and relations are depicted inside the respective entity or relation. We will adopt this convention in the rest of this thesis.

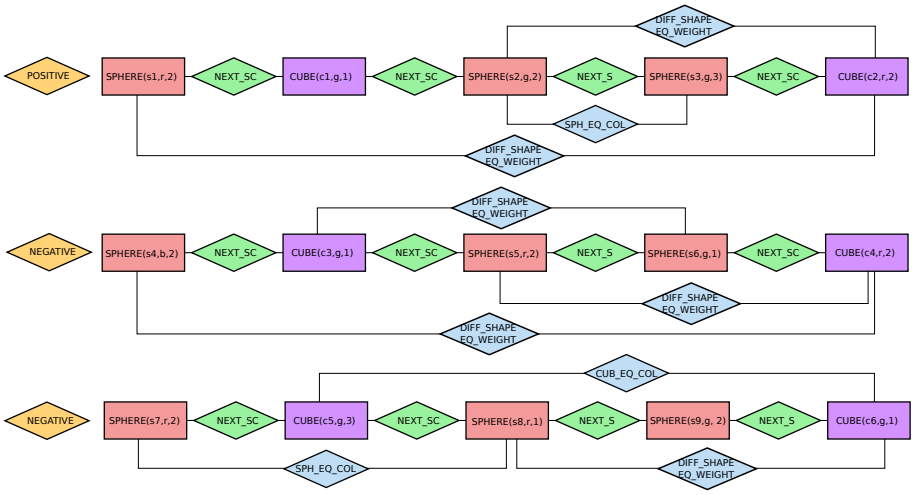


Figure 2.6: Artificial example: graphicalizations (i.e., unrolled E/R) for 3 instances.

The feature generation from the resulting graphs is performed by the Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) (Costa and De Grave, 2010), a particular type of graph kernel. NSPDK is a decomposition kernel (Haussler, 1999), in which pairs of subgraphs are compared to each other in order to calculate the similarity between two graphs. These subgraphs can be seen as circles in the graph, and are defined by three hyperparameters. First of all, there is the root or center of the subgraph, the *kernel point*, which can be any entity or relation in the graph. The entities and relations to be taken into account as kernel points are marked beforehand as a subset of the intensional and extensional domain relations. The *radius* r determines the size of the subgraphs and defines which entities or relations around the kernel point are taken into account. Each entity or relation that is within a number of r edges away from the kernel point is considered to be part of the subgraph. The third hyperparameter, the *distance* d, determines how far apart from each other the kernel points can be. Each subgraph around a kernel point that is within a distance d or less from the current kernel point will be considered. The kernel notion is finally given as the fraction of common fragments between two graphs. For the sake of self-containment we briefly report the formal definitions.

For a given graph $G = (V, E)$, and an integer $r \geq 0$, let $N_r^v(G)$ denote the subgraph of G rooted in v and induced[4] by the set of vertices $V_r^v \doteq \{x \in V :$

[4]In a graph G, the *induced subgraph* on a set of vertices $W = \{w_1, \ldots, w_k\}$ is a graph that has W as vertex set and contains every edge of G whose endpoints are in W.

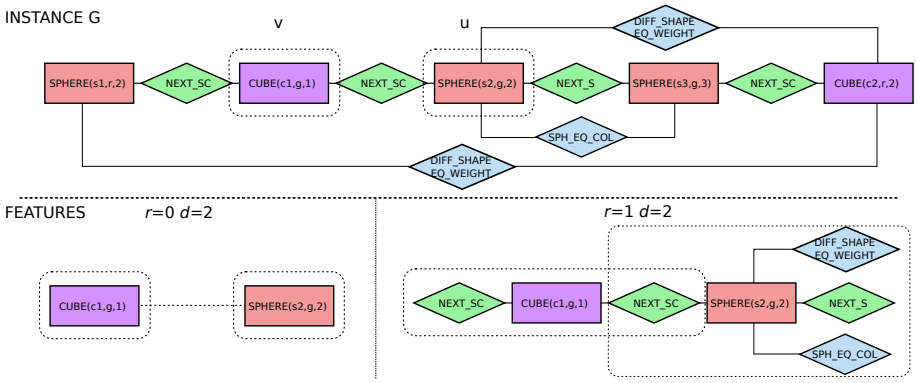


Figure 2.7: NSPDK features. Top: instance with two vertices v, u selected as roots for neighborhood subgraphs at distance $d = 2$. Bottom: neighborhood pairs (NSPDK features) at distance $d = 2$ with radius $r = 0$ and 1 respectively. Note that neighborhood subgraphs can overlap.

$d(x, v) \leq r\}$, where $d(x, v)$ is the shortest-path distance between x and v. A neighborhood $N_r^v(G)$ is therefore a topological *ball* with center v and radius r. The NSPDK feature concept is illustrated in Figure 2.7.

We introduce the relation $R_{r,d}$, which is defined in terms of neighborhood subgraphs, as:

$$R_{r,d} = \{(N_r^v(G), N_r^u(G), G) : d(u, v) = d\} \tag{2.28}$$

It identifies pairs of neighborhoods of radius r whose roots are exactly at distance d. Subsequently:

$$\kappa_{r,d}(G, G') = \sum_{\substack{A,\, B \,\in\, R_{r,d}^{-1}(G) \\ A',\, B' \,\in\, R_{r,d}^{-1}(G')}} \mathbf{1}_{A \cong A'} \cdot \mathbf{1}_{B \cong B'} \tag{2.29}$$

where $R_{r,d}^{-1}(G)$ indicates the multiset of all pairs of neighborhoods of radius r with roots at distance d that exist in G, and where $\mathbf{1}$ denotes the indicator function and $\cong$ the isomorphism between graphs. The normalized version of $\kappa_{r,d}$ is

$$\hat{\kappa}_{r,d}(G,G') = \frac{\kappa_{r,d}(G,G')}{\sqrt{\kappa_{r,d}(G,G)\kappa_{r,d}(G',G')}} \tag{2.30}$$

To increase efficiency and generalization power, the *zero-extension* of K is considered, obtained by imposing upper bounds on the radius and the distance parameter, with values to be determined e.g., via cross-validation:

$$K_{r_{max},d_{max}}(G,G') = \sum_{r=0}^{r_{max}} \sum_{d=0}^{d_{max}} \hat{\kappa}_{r,d}(G,G') \tag{2.31}$$

The result is an extended high-dimensional feature space[5] on which a statistical learning algorithm can be applied. Different learning tasks can be performed on this resulting feature space. To this end, kLog interfaces with several solvers, including LibSVM (Chang and Lin, 2011) and SVM SGD (Bottou, 2010).

2.2.4 ProbLog

ProbLog[6] (De Raedt et al., 2007; Kimmig et al., 2008) is a probabilistic Prolog that allows one to work with probabilistic facts and background knowledge. A ProbLog program consists of a set of definite clauses D and a set of probabilistic facts $p_i :: c_i$, which are facts c_i labeled with a probability p_i that their ground instances $c_i\theta$ are true. It is also assumed that the probabilities of all ground instances $c_i\theta$ are mutually independent.

Given a finite set of possible substitutions $\{\theta_{j1}, \ldots \theta_{ji_j}\}$ for each probabilistic fact $p_j :: c_j$, a ProbLog program $T = \{p_1 :: c_1, \cdots, p_n :: c_n\} \cup D$ defines a probability distribution

$$P(L \mid T) = \prod_{c_i\theta_{ji} \in L} p_i \prod_{c_i\theta_{ji} \in L_T \setminus L} (1 - p_i) \tag{2.32}$$

over ground subprograms $L \subseteq L_T = \{c_1\theta_{11}, \ldots c_1\theta_{1i_1}, \cdots, c_n\theta_{n1}, \ldots, c_n\theta_{ni_n}\}$ ProbLog is then used to compute the *success probability*

$$P_s(T \models q) = \sum_{\substack{L \subseteq L_T \\ L \cup D \models q}} P(L|T) \tag{2.33}$$

[5]Note that often a graph kernel is just an efficient procedure to compute a dot product between two instances and the feature representation is only available in an implicit fashion. For those kernels the explicit representation would lead to unacceptable storage and running times. Due to the specific assumptions in the NSPDK graph kernel however, while maintaining a very large overall feature space dimension, we operate with a small number of features for each instance, with the size being typically linear (with a small multiplicative constant) w.r.t. the input graph size (see (Costa and De Grave, 2010) for a detailed analysis).

[6]http://dtai.cs.kuleuven.be/problog/

of a query q in a ProbLog program T, where $P(q|L \cup D) = 1$ if there exists a θ such that $L \cup D \models q\theta$, and $P(q|L \cup D) = 0$ otherwise. In other words, the success probability of query q corresponds to the probability that the query q is *entailed* using the background knowledge together with a randomly sampled set of ground probabilistic facts.

As a clarifying example, consider the following windsurfing problem. It is inspired by Quinlan's playtennis example (Quinlan, 1986). The difference between playing tennis and going windsurfing is that windsurfing typically needs to be planned ahead of time, say on the previous day. The effect is that the weather conditions on the next day will still be uncertain at the time of deciding whether to go surfing or not. The forecast might state that tomorrow the probability of precipitation (pop) is 20%, the wind will be strong enough with probability 70%, and the sun is expected to shine 60% of the time, which could be represented by the facts:

```
0.2::pop(t).      0.7::windok(t).      0.6::sunshine(t).
```

where the `t` indicates the identifier for the example. Furthermore, additional background knowledge can be provided by means of the following logical rules:

```
surfing(X):- not pop(X), windok(X).
surfing(X):- not pop(X), sunshine(X).
```

where the argument `X` specifies the identifier of the example. The first rule states that if the expected precipitation is low and the wind is ok, the surfing is likely to be good. There is thus a declarative logical reading of these rules, but also a probabilistic one. The lower the precipitation is and the higher the probability of windok and sunshine the higher the probability that the surfing will be enjoyable. Assuming all facts in the description are independent, ProbLog can now compute the probability of the query `surfing(t)` as follows:

$$\begin{aligned} P(\texttt{surfing(t)}) &= P((\neg\texttt{pop(t)} \wedge \texttt{windok(t)}) \vee (\neg\texttt{pop(t)} \wedge \texttt{sunshine(t)}) \\ &= P((\neg\texttt{pop(t)} \wedge \texttt{windok(t)}) \vee (\neg\texttt{pop(t)} \wedge \texttt{sunshine(t)} \\ &\quad \wedge \neg\texttt{windok(t)})) \\ &= 0.8 \times 0.7 + 0.8 \times 0.6 \times 0.3 = 0.704 \end{aligned}$$

where the rewriting is needed to make the two events mutually exclusive.

2.3 Natural Language Processing

As already mentioned in the introduction, understanding natural language by means of a computer is a challenging problem that requires one to solve a set of different subproblems. For example, for a computer (e.g., a robot), being able to interpret a particular sentence means knowing the meaning of the individual words, learning how these words stand in relation to one another, and being able to relate this to what has been said earlier in the conversation. Each of these challenges is situated in one of the subfields of linguistics: phonology, morphology, lexicography, syntax, semantics, and discourse. For every subfield a computational complement exists. Since natural language can contain information at different levels of granularity, from simple word or token-based representations, over syntactic representations at sentence level, to high-level representations across documents, in most cases, information from different levels needs to be combined in order to do the analysis.

In this section, we discuss the most relevant tasks that will be used throughout the rest of the thesis. Some of these tasks are considered to be solved by the natural language processing community (i.e., on most morphological, lexical and syntactic tasks, promising results have been obtained over the last decade), whereas other (semantically-oriented) tasks are still a challenge. However, most of the tasks that we outline below, will be considered as preprocessing steps for our methods, for which state-of-the-art techniques were used. For more details, we refer the interested reader to Manning and Schütze (1999) and Mitkov (2003) for a thorough introduction to the field.

2.3.1 Morphology

Natural languages are characterized by a richness of hundreds of thousands of words, which each describe a particular concept in the real world. While a number of these words drop out of use over time, new words are invented on a daily basis. These words are constructed from a finite collection of smaller units. This is exactly the topic of *morphology*, which studies how words and word forms are constructed from smaller units, *morphemes*, in a systematic way. *Computational morphology* deals with the automated processing of words in both their written and spoken form. We frequently used one particular application of computational morphology as a preprocessing step in our methods, namely *lemmatizers*. However, we will first discuss *stemming*, the naive predecessor of lemmatization.

Stemming *Stemming* is the process of normalizing the morphological variants of a word. The resulting word is referred to as the *stem* of the original word.

This is done by removing the affixes (prefixes or suffixes) from the word. For example, the words *learner*, *learning* and *learned* can all be reduced to their stem *learn*. Note that the stem is not necessarily identical to the morphological root of the word. For example, the stem of *mining*, *mine*, *mined* and *miner* is *min*, not *mine*. Different types of stemming algorithms exist, with the Porter stemmer (Porter, 1980) being one of the most widely used algorithms to date.

Lemmatization For almost all applications, more accurate syntactic and semantic information on individual words is required, in which case stemming does not suffice. A word *lexeme* pairs a particular form of a word with its meaning, and a *lemma* is the grammatical form to represent this lexeme, i.e., the base or dictionary form of a word. Lemmatization uses a vocabulary and morphological analysis of the word to obtain the lemma. This is a non-deterministic process, as it depends on the context in which a particular word is used. For example, when lemmatizing the word *saw*, the result can either be *saw*, referring to the tool, if the word is used a a noun, or *see*, in case the word is used as a verb.

2.3.2 Lexicography and Lexical Analysis

A *lexicon* is the inventory of the lexemes of a particular language. A well-known instantiation of a lexicon is a *dictionary*, which is essentially an alphabetical listing of the lexicon of a given language. Lexicons, as inventories of words, exist in different formats and are used for a wide variety of NLP tasks. *Lexicography* refers to the process of compiling lexicons. *Computational lexicography* is the study of the representation and use of lexical knowledge in NLP systems.

Tokenization and sentence splitting In order to construct these lexicons from existing collections of texts (i.e., *corpora*), the texts need to be segmented into linguistic units such as words, punctuation symbols, numbers, etc. This process is called *tokenization*, whereas *tokenizers* are the methods to perform this task. In most cases, sequences of words are delimited by sentence boundaries, which requires methods to segment text into sentences. This is referred to as *sentence splitting*. These tasks are relatively simple, especially in the case of so-called segmented languages, such as English, where words and sentences are separated by spaces and punctuation symbols. Consequently, most tokenization and sentence splitting algorithms are based on regular expressions, which keep track of a set of these symbols as possible delimiters for words or sentences.

2.3.3 Syntactic Analysis

Syntax refers to the knowledge of the *structural relationships* between words, i.e., the way that words are arranged together. The goal of *syntactic analysis* is thus to determine this underlying structure. Two of the most important tasks in this process are *part-of-speech tagging* and *parsing*.

Part-of-speech tagging

Based on the *syntactic role* a word fulfills in a sentence, it can be assigned a particular tag that indicates this role. Some of the most frequent roles are nouns, verbs, adverbs and adjectives. The goal of *part-of-speech tagging* (POS tagging) is to assign these tags to words in a sentence. One of the most commonly used tag sets is the Penn Treebank tag set (Marcus et al., 1993). As an example, consider the following annotated sentence:

NNP/Natural NNP/Language NNP/Processing VBZ/is DT/an JJ/exciting NN/research NN/domain ./!

For example, the tag VBZ indicates that *is* is a verb in the third person singular present, DT that *an* is a determiner and NN that *research* is a noun.

As language is often ambiguous, a particular word can take on different roles dependent on the context, which increases the complexity of the problem. However, over the past decade, a lot of progress has been made on this task, resulting in state-of-the-art POS taggers with high accuracy (e.g., Toutanova et al. (2003)), due to which POS tags are often used as reliable features for machine learning systems in NLP tasks.

Parsing

Parsing assigns a syntactic analysis to a sequence of words according to a formal grammar. Whereas part-of-speech tagging assigns a syntactic meaning to the individual words in a sentence, parsing focuses on the *relations* between the words. The most basic approach is *chunking* or *shallow parsing*, which partitions the input into a sequence of non-overlapping units, and assigns a syntactic category to each of these sequences. Two other, more advanced methods for finding the syntactic structure of a sentence are *constituent-based* and *dependency-based parsing*.

Chunking In contrast to part-of-speech tagging, which focuses on assigning labels to individual words, *chunking* or *shallow parsing* assigns tags to *phrases*,

i.e., groups of word that belong together. Consider the same example sentence, which is now annotated with chunk tags:

NP/[Natural Language Processing] VP/[is] NP/[an exciting research domain] !

The example contains two of the most common tags; NP indicates that *Natural Language Processing* and *an exciting research domain* are two noun phrases, whereas VP shows that *is* is a verb phrase. Chunking can be seen as in intermediate step towards full parsing.

Constituent-based parsing A *constituent* is defined as a group of words that may behave as a single unit or phrase. Constituent-based parsing organizes the words in a sentence into nested constituents. This structure is ordered as a tree, due to which the result is referred to as a *parse tree*. The tree is constructed according to the rules in a formal grammar, which can either be constructed by hand by a linguist, or can be induced automatically from a *treebank* (i.e., a text corpus in which each sentence is annotated with its syntactic structure).

For English, most commonly a *context-free grammar* (CFG) is used. A CFG is a formal grammar consisting of production rules of the form $N \rightarrow t$, in which N represents a non-terminal symbol, and t a string of terminal and/or non-terminal symbols. In parsing, the set of terminal symbols consists of the words in the vocabulary that appear in the sentences generated by the grammar, whereas the non-terminal symbols are the grammatical categories. Using the production rules that form the the grammar, the non-terminal symbols can then be replaced with terminal symbols.

A constituent-based parse tree for the example sentence is given in Figure 2.8. Among others, the parse tree illustrates the production rule $S \rightarrow NP\ VP$, which states that a sentence S can be decomposed in a noun phrase NP (*Natural Language Processing*) and a verb phrase VP (*is*).

Dependency-based parsing *Dependency-based parsing* uses *dependency grammars*[7] to determine the syntactic structure of a sentence. In contrast to the context-free grammars used in constituent-based parsing, dependency grammars see all nodes as terminal symbols, i.e., they do not distinguish between terminal and non-terminal symbols. The dependency parse tree shows which words depend on (i.e., are arguments of or modify) which other words.[8]

[7] Note that also dependency grammars can be induced automatically from a corpus.

[8] Note that a dependency parse tree can be translated into a constituent-based parse tree and vice versa.

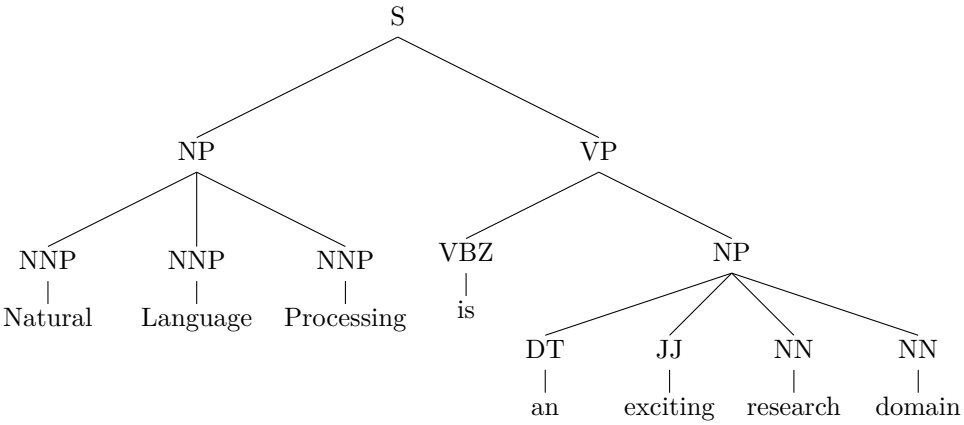


Figure 2.8: Constituent-based parsing example.

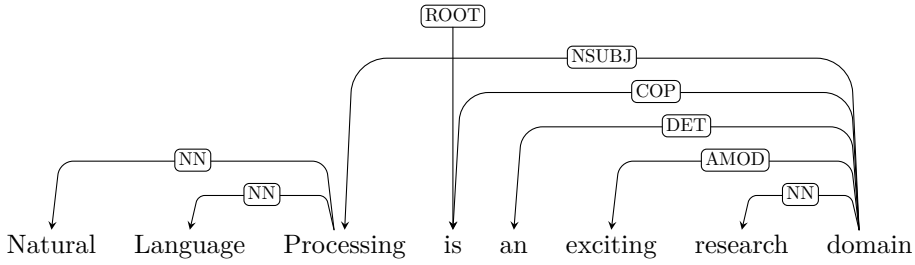


Figure 2.9: Dependency-based parsing example.

Figure 2.9 shows a dependency parse tree for the example sentence. The relation COP indicates that *domain* is the complement of the copular verb *is*. As for constituent-based parsing, DET indicates that *an* is the determiner of *domain*.

2.3.4 Semantic Analysis

Whereas syntax refers to the structure, *semantics* studies the *meaning* that is used for understanding and conveying human expression through language. *Semantic analysis* deals with giving words, phrases, sentences or paragraphs meaning, i.e., coupling them to elements and concepts in the real world. The goal is to relate their syntactic structure to their language-independent meaning. We will briefly discuss four of the most commonly studied semantic analysis problems that are considered benchmark tasks in NLP.

Named entity recognition *Named entity recognition* (NER) determines which words or phrases in the text map to proper names. The assigned tags represent the type of the element in the world, such as the names of persons, organizations, locations, etc. A first task in this process is to locate the proper names in a sentence, whereas the second step classifies each of the identified proper names into one of the categories. NER is a subtask of information extraction. As an example, consider the following sentence:

The ORGANIZATION/[University of Leuven] is a university in LOCATION[Belgium] that was founded in DATE[1425].

for which the named entity recognizer was able to identify that *University of Leuven* is an organization, *Belgium* is a location and *1425* is a date.

Word sense disambiguation A lot of words can have several meanings. For example, the word *bank* can refer to the financial institution, the border of a river, or to one of its meanings when used as a verb. As knowing the meaning of a word is a primary condition for natural language understanding, the goal of *word sense disambiguation* (WSD) is to determine the meaning of a word given its context. A number of lexical databases list words together with their possible senses, and thus are helpful resources in solving this task.

WordNet[9] (Miller, 1995) is one of the most frequently used lexical databases of English. It groups nouns, verbs, adjectives and adverbs into sets of synonyms, referred to as *synsets*. Each synset is accompanied by a short definition. In addition to the dictionary functionality, WordNet is invested with an ontological structure that interlinks synsets by means of semantic and lexical relations. The most frequent relation among synsets is the super-subordinate relation (also referred to as hyperonymy-hyponymy, or more commonly as the IS-A relation, e.g., *car* and *vehicle*). Other encoded relations are meronymy, the part-whole relation (e.g., *car* and *wheel*), and antonymy (e.g., *young* and *old*). Verb synsets are also arranged into hierarchies.

Semantic role labeling A *semantic role* is defined as the relationship that a syntactic constituent has with a predicate, i.e., the verb. The goal of *semantic role labeling* is now to identify all syntactic constituents that fill a semantic role, and determine this role. We will illustrate this by means of the following labeled sentence:

[A0 They] [V visited] [A1 Bruges] [AM-TMP during the weekend].

[9] http://wordnet.princeton.edu

They, labeled as A0, refers to the visitors and forms the agent of the *visit* (V) action, whereas *Bruges* (A1) is the visited place and is the object of the action. The phrase *during the weekend* is an adjunct that refers to the timing of the action. Intuitively, one uses semantic role labeling when answering wh-questions (e.g., *Who visited Bruges?*, *What place did they visit?*). Semantic role labeling is also referred to as *shallow semantic parsing*. Just as chunking gave a shallow view on the structure of a sentence, semantic role labeling can be seen as a shallow view on its meaning.

Coreference resolution An *anaphor* is a word or phrase that refers to an other word or phrase that was previously mentioned in the text. The latter is called the *antecedent*. When both the anaphor and the antecedent refer to the same entity in the real world, they are called *coreferent*. As the name indicates, *coreference resolution* refers to the identification of words or phrases in a text that refer to the same entity in the real world. This is illustrated in the following example:

$$[\text{The manager}]_1 \text{ said } [\text{he}]_1 \text{ would propose } [\text{it}]_2 \text{ to } [\text{the Board}]_3.$$

The words and phrases between brackets are called *mentions*. Their subscripts indicate the coreferences. *The manager* and *he* are coreferent, as they both refer to the same entity in the real world.

2.4 Conclusions

As this thesis is situated at the intersection between statistical relational learning and natural language processing, this chapter reviewed the relevant background knowledge on the relevant concepts and techniques in both of these domains. First, the basic principles and notation of machine learning that will be used and extended in the rest of this thesis were discussed. Subsequently, logic programming was introduced as a strong relational formalism on which SRL systems are built, followed by a discussion of kLog and ProbLog, two SRL frameworks that are central to this thesis. Finally, an overview of the most relevant natural language processing tasks was given, that will be used throughout the rest of the thesis, mostly as preprocessing steps for our methods. The principles and techniques in which this thesis is founded thus form a strong basis on which will be built in the rest of this thesis.

Chapter 3

Related Work

This chapter will provide an overview of related work at the intersection of statistical relational learning and natural language processing. In the following chapters, some tasks in natural language learning will be tackled using SRL techniques. Related work which is specific to these tasks will be discussed at the start of the respective chapters.

The foundations for the interest in statistical relational learning of natural language were laid when the importance of the role of logic in natural language processing became clear. Logic-based approaches enabled an increased flexibility and expressivity. Furthermore, their clean representation of a wide range of linguistic information proved well-suited for a number of problems in NLP. For example, the grammatical production rule $S \rightarrow NP\ VP$ introduced in the parsing example of Section 2.3.3 can naturally be represented by the definite clause grammar rule $s(Start, End) \rightarrow np(Start, Middle), vp(Middle, End)$, where the variables indicate index vertices in between words. This representation of grammars, referred to as Definite Clause Grammars (Colmerauer, 1978; Pereira and Warren, 1986), is one of the prime successes of *logic programming* for natural language processing. Dahl (1994) studies the use of logic programming for NLP from a theoretical perspective. Covington (1993) examines the application of the logic programming language Prolog for NLP from a practical point of view.

The downside of logic programming is that it requires the programmer to manually encode the language descriptions, which is infeasible in a real-world setting. This problem was partly alleviated by the introduction of statistical NLP approaches, which were able to learn from examples in large text corpora. However, the resulting models were less interpretable than their

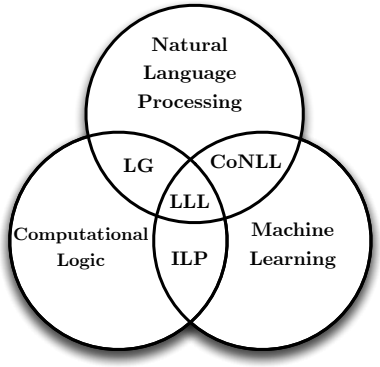


Figure 3.1: Situating Learning Language in Logic (Adapted from Džeroski et al., 1999).

logical counterparts, due to which alternatives were sought that offered the best of both worlds. The solution was offered by *inductive logic programming*.

3.1 Inductive Logic Programming

As outlined before, in contrast to logic programming, which is concerned with deductive inference, *inductive* logic programming uses inductive inference. This allows one to find hypotheses by generalizing from examples in the presence of background knowledge. The application of inductive logic programming to NLP led to the research domain of *Learning Language in Logic* (LLL). Whereas definite clause grammars, as an example of *logic grammars* (LG), combined computational logic and natural language processing, LLL is situated at the intersection of machine learning, NLP and computational logic, as illustrated in Figure 3.1.

The motivation for the use of ILP for NLP was summarized by Muggleton in (Cussens and Džeroski, 2000) as follows:

> *"From the NLP point of view the promise of ILP is that it will be able to steer a mid-course between the two alternatives of large scale, but shallow levels of analysis, and small scale, but deep and precise analyses, and produce a better balance between breadth of coverage and depth of analysis."*

A general introduction to the field is given by Džeroski et al. (1999). Main results in the area have been reported in the Learning Language in Logic workshop

series (Cussens and Džeroski, 2000). In this section, we will give an overview of the area. For a comprehensive survey, we refer the interested reader to the proceedings of the LLL workshops[1]. Other research efforts along this line of research took place in various initiatives of the ACL special interest group on Natural Language Learning[2] (SIGNLL) and the European Association for Logic, Language and Information[3] (FoLLI).

3.1.1 ILP for Morphological and Syntactic Processing Tasks

Initial research in LLL mainly focused on natural language processing problems for which ILP could overcome the shortcomings of logic programming and statistical approaches. As outlined before, ILP has increased the flexibility using a rich knowledge representation language and has limited the amount of manual feature engineering, which were two of the main issues of empirical, propositional NLP systems.

One of these problems is *parsing*. An early approach to grammar learning using ILP was presented by Wirth (1988, 1989). It starts from a set of two clauses and uses ILP to find the missing clause in order to learn a small definite clause grammar. CHILL (Zelle and Mooney, 1993) was one of the first specialized systems applying ILP to NLP. Similar to the approach of Wirth, it starts from an overly-general parser, and subsequently learns heuristics from a training corpus of parsed sentences, which are added to the parser to control it. This method was able to achieve results comparable to contemporary statistical methods. Also other ILP approaches to parsing were developed. For example, Cussens and Pulman (2000) used ILP within a chart-parsing framework for grammar learning. Given a general grammar and a set of unparsable sentences, ILP finds the missing grammar rules that are needed to parse the sentences. Besides this top-down approach, the same authors also present a bottom-up approach on the same task, in which they use linguistic constraints to generate new grammar rules, starting from a set of highly-specific rules. Cussens et al. (1997) demonstrated the use of Stochastic Logic Programs, a generalization of stochastic grammars, for learning grammars from children's books. The EMILE system of Adriaans and de Haas (2000) uses a less expressive variant of first-order logic, substructural logic, for ILP in order to reduce the computational cost. Its applicability for language learning was illustrated on grammar induction. Finally, the GRIND system of Nepil (2001) combines ILP with transformation-based learning, an error-driven rule learning technique. Besides parsing, grammar learning has also been employed for other tasks, e.g., to recognize a specific class of proteins using the ILP system CPROGOL (Muggleton et al., 2000).

[1] http://www.cs.york.ac.uk/aig/lll/
[2] http://ifarm.nl/signll/
[3] http://www.folli.info/

ILP approaches have been developed for other syntactic tasks as well. Eineborg and Lindberg (2000) presented an overview of ILP for *part-of-speech tagging*. One particular instance of such a system uses the ILP framework PROGOL to induce the tagging rules in the form of definite clauses (Cussens, 1997). A PROGOL-based POS-tagger was also integrated in RTDF (Alexin et al., 2001), a rule-based tagger. Jorge and de Andrade Lopes (2000) described the application of iterative induction, in which a theory is iteratively specialized to cover more examples, to part-of-speech tagging. Also ALEPH, a successor of the PROGOL system, has proved to be useful for syntactic processing tasks (such as part-of-speech tagging (Nepil et al., 2001) and chunking (Konstantopoulos, 2002).

Also tasks in *morphology* proved helpful in illustrating the additional capabilities of ILP for natural language processing. For example, Mooney and Califf (1995) showed that their ILP system, FOIDL, was able to outperform neural network and decision tree methods for the generation of past tenses of English verbs. Muggleton and Bain (1999) were able to improve on the results of FOIDL using analogical prediction, a technique on the intersection between memory-based learning and ILP. FOIDL has also been used to learn the inflectional paradigms (i.e., the different modifications of a word to express different grammatical categories) of Slovene nouns (Džeroski and Erjavec, 1997). The same authors also employed CLOG, a system similar to FOIDL that learns first-order decision lists, to lemmatize Slovene words (Džeroski and Erjavec, 2000). A combination of ILP (by means of the PROGOL system) and kNN to lemmatize Czech words was presented by Popelínský and Pavelek (1999). This technique was later integrated into a system for recognizing and tagging compound verb groups (i.e., sets of words that together determine the meaning of a verb) in Czech (Žáčková et al., 2000). Manandhar et al. (1998) compared CLOG and FOIDL for learning inflectional paradigms of nouns and adjectives in different languages. The clausal discovery engine CLAUDIEN (Dehaspe et al., 1995) was used for the discovery of diminutive forms in Dutch. A general survey of ILP approaches to learning tasks in morphology was offered by Kazakov (2000) .

3.1.2 ILP for Semantic Processing Tasks

Besides syntactic parsing, the CHILL system mentioned in the previous section also translates English database queries into an executable logical form, the so-called *semantic parsing*. It proved to be more accurate than a manually-encoded program. Mooney (1996) gave an overview of the evolution of the system and outlined a number of challenges for the application of ILP to NLP problems; namely solving NLP problems with *generic* ILP systems and extending ILP systems beyond classification, allowing them to generate output rather than testing existing tuples. This shift from classification to disambiguation in ILP

for solving NLP tasks was also addressed by Thompson and Califf (2000) and Riezler (2000). In the former, the authors presented an active learning approach using CHILL and RAPIER (Califf and Mooney, 2003), a bottom-up relational learner, both for semantic parsing and learning information extraction rules. The latter paper presented an approach to probabilistic modeling of constraint-based grammars with log-linear distributions. The COCKTAIL system (Tang and Mooney, 2001) combined CHILL with mFOIL, an ILP algorithm for rule learning, for semantic parsing. Liakata and Pulman (2004a,b) presented a method using WARMR, a relational frequent pattern mining algorithm, to learn a simple domain theory (i.e., a set of rules that describe entities and relations between these entities) from text.

The successful application of ILP on the aforementioned syntactic tasks, and the initial results of CHILL and RAPIER for semantic parsing and information extraction, have led to an increasing interest in applying ILP for tasks focused on semantics. An initial ILP approach for *document classification* was presented by Cohen (1995). He represented documents as a collection of words with their position, with some additional predicates to allow for phrases and extract sequence information. Interestingly, the author also compared the relational method to its propositional counterpart. The aim of the SRV system presented by Freitag (1998) was to offer a general-purpose learning approach for *information extraction*, by separating domain-specific information from the learning algorithm. The system's feature representation includes both syntactic and lexical information. The goal of Junker et al. (1999) was to develop a unified representation of typical text patterns, which enables one to formulate rule learning problems for text classification and information extraction, and can be used in standard ILP techniques. Related to this, Sébillot et al. (2000) presented an ILP approach for automatically extracting a semantic lexicon consisting of noun-verb pairs which could serve as an index for *information retrieval* systems. Only pairs that belong to a selected subset of meaningful semantic roles are extracted. The extraction is based on the part-of-speech tags of the words in the surrounding context of the pairs. This work was extended by Claveau et al. (2003) with the inclusion of semantic tagging and additional contextual information. Whereas all previous systems focused on English, Esposito et al. (2000) studied the learning of information extraction rules for Italian starting from a logic representation of parsed sentences.

Due to the ability of ILP to incorporate background knowledge, its flexible representation of semantic relations, and motivated by the successful applications for semantic parsing and the extraction of information extraction rules, the interest rose to apply ILP for *ontology learning*. The ASIUM system (Nédellec, 1999) learns ontologies and verb categorization frames in the form of logical predicates from parsed corpora. These enabled the induction of a taxonomic

hierarchy by conceptual clustering. Aitken (2002) presented an ILP approach using the rule-based learning algorithm FOIL for extracting ontological relations from Web text in order to extend the Semantic Web. Related to ontology learning, Cimiano et al. (2008) presented an ILP approach for computing intensional answers given a set of extensional answers returned as a result of a user query to an information system. The more recent work of Lisi (2012) discussed the application of ILP for learning onto-relational rules, i.e., the rules that complement and extend ontologies on the Semantic Web. Nakabasami (2001) presented a system to identify the differences between documents. In a first step, memory-based learning is employed to determine interesting lexical and semantic word features. Subsequently, this information serves as background knowledge for ALEPH, which induces clauses that are used to determine the difference between documents. Intermediate user feedback is given to the system to increase the quality of the results.

Specia et al. (2006) and Specia (2006) investigated the role of integrating additional background knowledge into the learning process for *word sense disambiguation*. The declarative specification of additional background knowledge is one of ILP's distinguishing characteristics when compared to statistical approaches. This was done in two different ways. In a first approach, ILP was used both to specify the background knowledge and perform the learning, whereas in a second approach it was employed solely for the construction of interesting features which were subsequently used by standard statistical learning algorithms. Results showed that the approaches, both integrating a wide range of syntactic and semantic information, were able to outperform contemporary state-of-the-art statistical learners using shallow syntactic features. The authors also participated to the SemEval 2007-Shared Task on lexical sample word sense disambiguation (Specia et al., 2007). Yang et al. (2008) presented an entity-mention model for *coreference resolution* using ILP. It can capture information beyond single mention pairs and express the relations between an entity and its mention, in contrast to contemporary state-of-the-art entity-mention models. Furthermore, rules expressing the existence of a coreference relation between an entity and a mention in terms of the declaratively specified features can be learned.

Also in the context of the genic interaction challenge (Nédellec, 2005), a shared task at LLL'05 which focused on *information extraction from biomedical texts*, a number of ILP approaches were developed. Given sentences describing interactions between genes and proteins, where proteins are the agents of interaction and genes are the targets, the goal of the challenge was to induce rules for extracting gene/protein interactions. GLEANER (Goadrich et al., 2004), the system of Goadrich et al. (2005), and the approach of Popelínský and Blaťák (2005) used ALEPH. The former system first learns a set of clauses, and

subsequently combines them into a thresholded disjunctive clause, with the aim of selecting the optimal tradeoff between precision and recall. The latter approach learns two sets of rules to extract genic interactions: one set for simple examples, and a more general set for all examples. The authors also compare the relational representation with a propositionalized representation. Riedel and Klein (2005) noted that due to the small size of the training set, the ILP methods they tried were not able to generalize well. They decided to explore the combination of ILP and statistical methods. To this end, they turned to statistical relational learning, which we will now discuss in detail.

3.2 Statistical Relational Learning

As illustrated in the previous section, the interpretability of the logical representations, and the possibilities to model relational information and declaratively specify additional background knowledge, resulted in a number of successful applications of inductive logic programming for natural language processing tasks. At around the same time, statistical approaches for NLP tasks were on the rise. This awakened the interest in applying statistical relational learning, which extends ILP with statistical learning theory, to NLP problems.

A wide range of statistical relational learning systems exist (Getoor and Taskar, 2007). Two of these, kLog and ProbLog, were already discussed in Section 2.2. In principle, many SRL systems are useful to solve NLP problems. In this section, we will discuss the main results in this area according to the technique with which the problem was solved.

3.2.1 Markov Logic

One of the most popular SRL formalisms, *Markov logic* (Richardson and Domingos, 2006), was one of the first techniques to be applied to NLP tasks. Markov logic extends first-order logic by attaching weights to formulas. This set of weighted first-order clauses is referred to as a *Markov logic network* (MLN). It can can be seen as a template for constructing Markov networks, a particular type of undirected graphical model that allows to represent relational information.

Bunescu and Mooney (2004, 2007) studied the problem of information extraction from biomedical texts. The goal is to identify relations between words or phrases that occur in different parts of a sentence or paragraph. This typically requires one to take into account some context, and integrate features from different levels (e.g., morphologic, syntactic and semantic information). Inspired by the integer programming approach of Roth and Yih (2004), Bunescu and Mooney (2004) investigated the use of Relational Markov Networks (Taskar et al., 2002)

for the information extraction problem. The system is able to outperform a traditional conditional random fields approach. It illustrates the ability of SRL to integrate uncertain evidence at multiple levels in order to collectively determine a globally coherent solution. This formed the inspiration for a number of applications of Markov networks and Markov logic for both supervised and unsupervised learning tasks in NLP.

In the *supervised* setting, McCallum and Wellner (2004) illustrated the use of Markov networks for noun phrase coreference resolution. Due to their relational nature, their models did not assume that pairwise coreference decisions should be made independently from each other, in contrast to most of the earlier approaches. Furthermore, they can represent features at different levels simultaneously. Singla and Domingos (2006) presented a follow-up approach that represents the noun phrase coreference resolution problem in Markov logic. It allowed them to model the problem using a small number of predicates, and enabled them to outperform the original approach of McCallum and Wellner (2004). A similar joint learning approach combining pairwise classification and mention clustering with Markov logic was presented by Song et al. (2012).

Poon and Domingos (2007) studied information extraction, but besides segmentation (i.e., locating the candidate fields) (Bunescu and Mooney, 2004), they also performed entity resolution (i.e., identifying duplicate records). This was done using joint inference, i.e., the segmentation and entity resolution are performed in a single inference process. Satpal et al. (2011) used MLNs to extract data from the Web, integrating both properties of individual words, as well as the structure of the pages and sites. Riedel et al. (2009) presented a Markov logic approach for bio-molecular event extraction, in which a joint probabilistic model over events in a sentence is learned. By using a similar approach, but a novel formulation for the model, Poon and Vanderwende (2010) were able to improve on the state-of-the-art results on this task. Yoshikawa et al. (2010) extended the earlier methods by incorporating coreference relations as additional cross-sentence information and tackled the task of cross-sentence event-argument relation extraction. Yoshikawa et al. (2009) focused on the extraction of temporal relations. By using Markov logic, logical constraints that hold between events and time expressions (e.g., before, after, or overlap) can easily be incorporated, which were neglected by previous approaches. Furthermore, as in earlier work, these relations can be predicted jointly. The TIE system of Ling and Weld (2010) went one step further, in that it was able to extract facts from text while inducing as much temporal information as possible. Furthermore, the authors used global inference to enforce transitivity to bound the start and end times for each event. The systems of UzZaman and Allen (2010) combine deep semantic parsing, Markov logic networks and conditional random fields. A number of other approaches for information extraction using MLNs exist.

For example, the StatSnowball system (Zhu et al., 2009) was developed for open information extraction (i.e., extracting relations without predefining which ones), the approach of Yu and Lam (2008) for extracting relations between entities in Wikipedia articles.

A number of approaches were proposed for solving some other semantic tasks using Markov logic. Toutanova et al. (2008) presented a model for semantic role labeling, and showed that the SRL approach was able to outperform a state-of-the-art approach that did not include dependencies among different arguments, but only considered local context. Meza-Ruiz and Riedel (2009) combined semantic role labeling with predicate senses using a Markov logic formulation in which they jointly identify predicates, arguments and senses. They also discussed the benefit of this joint approach when compared to a pipeline system. Che and Liu (2010) extended this approach into a joint model of semantic role labeling and word sense disambiguation, in which all senses are used.

The application of Markov logic for *unsupervised* learning was also explored. One of the first unsupervised approaches was presented by Kok and Domingos (2008). In a first step, a set of tuples is extracted from text. Subsequently, these tuples are used to induce semantic networks consisting of general concepts and relations by jointly clustering the objects and relations in the tuples. Poon and Domingos (2008) presented an unsupervised coreference resolution approach. Their MLN approach leverages syntactic relations that are useful for the task and can easily be represented. It was the first unsupervised approach that was as accurate as supervised systems on this task. Hou et al. (2013) presented a global model using MLNs for bridging anaphora resolution, a particular type of coreference resolution.

USP (Poon and Domingos, 2009) was the first approach for unsupervised semantic parsing. This is done by converting dependency trees into quasi-logical forms. These are subsequently clustered together to abstract away the syntactic variations with the same meaning. The clustering is performed by Markov logic, which starts by clustering tokens of the same type together, after which expressions whose subexpressions belong to the same clusters are recursively joined. This work was extended by the same authors to OntoUSP, a system that induces and populates a probabilistic ontology based on unsupervised semantic parsing (Poon and Domingos, 2010). Since the clusters induced by USP do not align with the concepts in a database, Poon (2013) combined unsupervised semantic parsing with grounded learning from a database. Related to this is the Markov logic-based framework of Niepert et al. (2010) for ontology matching, and their declarative framework for web data integration (Niepert et al., 2011). The overall goal is to arrive at an integrated approach for machine

reading[4], i.e., the automated knowledge extraction from text (Poon, 2011).

3.2.2 kLog

As already indicated in Section 2.2.3, kLog will be one of the two SRL frameworks which we will focus on in the rest of this thesis. Besides a number of successful applications in computer vision, e.g., for hierarchical image understanding (Antanas et al., 2012) and scene classification (Antanas et al., 2013), kLog has also shown its effectiveness for natural language processing. Kordjamshidi et al. (2012) presented a relational representation in kLog for the task of spatial role labeling, where the goal is to extract generic spatial semantics from natural language. The authors point out the appropriateness of the NSPDK graph kernel to capture long-distance dependencies in language, and the flexibility of the logical language for performing experiments in relational domains.

3.2.3 Other SRL Frameworks

Kurihara and Sato (2006) proposed a variational Bayesian learning method for probabilistic context-free grammars. This method was later added to the SRL system PRISM (PRogramming In Statistical Modeling) (Sato and Kameya, 1997; Sato et al., 2008). As in ProbLog, a PRISM program defines a probability distribution over a set of Herbrand interpretations, but in contrast to ProbLog, it imposes a number of constraints on the allowed programs to limit complexity.

Church (Goodman et al., 2008) is a probabilistic programming language which adds probabilistic semantics to the programming language Scheme. O'Donnell et al. (2009) illustrated its use for modeling language structure. To this end, the authors introduce fragment grammars as a structure that optimizes the amount of linguistic knowledge that needs to be stored and the computations that need to be executed in order to construct linguistic structures (e.g., sentences).

As Poon and Domingos (2007), Singh et al. (2009) focused on joint inference for information extraction in the context of citation matching. Both approaches use a conditionally-trained factor graph, a particular type of graphical model, on which inference is performed. However, in contrast to the first-order logic representation as used by Poon and Domingos, Singh et al. leverage Factorie (McCallum et al., 2009), a probabilistic programming language that uses imperative procedures to define the factor template structure.

Blog (Milch et al., 2005), an acronym for Bayesian logic, is an SRL system for generative learning that uses elements of first-order logic and approximate inference. It has been used on the information extraction task in the context

[4] A detailed discussion of related work on machine reading is offered in Chapter 8.

of citation matching discussed in the previous paragraph (Carbonetto et al., 2005). This approach was later adapted by Yu and Lam (2012) to work in a discriminative learning setting and to be able to incorporate more complex dependencies among inputs. The work of Culotta et al. (2007) on coreference resolution uses a similar representation as BLOG, but also works in a discriminative learning setting.

Raghavan et al. (2012) explored the use of Bayesian Logic Programs (BLP) (Kersting and De Raedt, 2007) in the context of machine reading. As indicated before, related work on this topic will be discussed in detail in Chapter 8.

Probabilistic Soft Logic (PSL) (Kimmig et al., 2012) is an SRL modeling language whose models are templates for hinge-loss Markov random fields, a particular class of probabilistic graphical models. Its use for natural language processing has been explored by Beltagy et al. (2014), who use it for assessing the semantic similarity of natural-language sentences. To this end, PSL was used to combine logical and distributional representations of meaning. The distributional information, i.e., statistics on contextual data from large corpora to predict semantic similarity of words and phrases, is represented as weighted inference rules.

A number of approaches that are closely related to SRL have been proposed for natural language processing tasks. For example, a number of techniques using *integer linear programming* have been proposed. These include the work by Roth and Yih (2007), who developed a linear programming formulation to address global inference for entity and relation identification. Another example is the work by Jenatton et al. (2012), who propose a *latent factor model* for multi-relational data, and illustrate their approach on learning semantic representations of verbs. The main difference with the definition of SRL adopted in this thesis, is that no explicit underlying logical representation is used.

3.3 Graph Kernels

As indicated before, kLog is one of the two SRL frameworks that will be used and extended in this thesis. Since it is rooted in learning with graph kernels, we also provide a brief overview of related work in this area. A general survey on kernels for structured data is offered by Gärtner (2003). A wide range of approaches that use kernels specifically for natural language processing tasks exist, for example for named-entity tagging (Cumby and Roth, 2003), chunking (Daumé and Marcu, 2005), relation extraction (Zelenko et al., 2003), or question answering (Moschitti and Zanzotto, 2007). A recent general overview on kernel methods for natural language processing can be found in the ACL tutorial by Moschitti (2012).

As explained in Section 2.2.3, the NSPDK kernel employed by kLog is an instance of a decomposition kernel, which decomposes a discrete structure into several parts, and computes the similarity between the instances starting from the similarity between these fragments. As shown by Collins and Duffy (2002), this type of kernels are effective for natural language processing. The authors describe kernels for various NLP structures (such as sequences and trees), and show that this type of kernels offers a computationally feasible representation in high-dimensional feature spaces. The authors illustrate their effectiveness on parsing English sentences. Moschitti (2004) used convolution kernels for semantic role labeling. The weighted decomposition kernel (Menchetti et al., 2005), a predecessor of the NSPDK kernel, has also proved successful for natural language processing, as illustrated by Costa et al. (2006) for named entity recognition and prepositional phrase attachment ambiguity resolution, i.e., determining if a prepositional phrase is part of a noun phrase or an argument of a verb. To the best of our knowledge, kLog is however the only framework that tailors graph kernels for statistical relational learning.

3.4 Conclusions

In this chapter, we gave an overview of related work at the intersection of inductive logic programming and statistical relational learning on the one side, and natural language processing on the other, throughout the history of the former research fields. It showed that the logical and relational representation of ILP, and its combination with statistical learning in SRL has already proved successful in a large number of tasks in NLP. The goal of this thesis is to deepen these insights and advance the research in this domain by building on these findings.

Part II

Graph Kernel–Based Relational Learning of Natural Language

Chapter 4

Sentence Classification with Sentence Context

In this chapter we will explore the use of graph kernel–based relational learning for binary sentence classification. The graph-based approach offers a flexible way to represent relational information that is inherently present in natural language, and enables the use of sentence context. The task under consideration is *hedge cue detection*, where the goal is to distinguish factual from uncertain information. In addition to results that outperform the state-of-the-art, a detailed analysis of the influence of the relational representation is provided. To this end, the relational approach is contrasted with propositional approaches for both lazy and eager learning. As part of this investigation, a novel lazy relational learning technique is contributed.

This chapter starts with a discussion of the task, hedge cue detection (Section 4.1), and an overview of related work (Section 4.2). In Section 4.3, the method is outlined, after which the results are presented in Section 4.4. An analysis of the advantages of a relational representation for NLP tasks is the topic of Section 4.5, in which we also introduce a new relational memory-based learning approach. Finally, Section 4.6 concludes this chapter.

This chapter is based on

> Mathias Verbeke, Paolo Frasconi, Vincent Van Asch, Roser Morante, Walter Daelemans, and Luc De Raedt. Kernel-based logical and relational learning with kLog for hedge cue detection. In Stephen H. Muggleton, Alireza Tamaddoni-Nezhad, and Francesca A. Lisi, editors, *Proceedings*

of the 21st International Conference on Inductive Logic Programming, Inductive Logic Programming, Windsor Great Park, UK, 31 July 2011 - 3 August 2011, pages 347–357. Springer, 2012.

and

Mathias Verbeke, Vincent Van Asch, Walter Daelemans, and Luc De Raedt. Lazy and eager relational learning using graph-kernels. In Laurent Besacier, Adrian-Horia Dediu, and Carlos Martín-Vide, editors, *Proceedings of the Second International Conference on Statistical Language and Speech Processing, International Conference on Statistical Language and Speech Processing, Grenoble, France, 14-16 October 2014*, pages 171–184. Springer, 2014.

4.1 The Task: Hedge Cue Detection

Information Extraction (IE) is a subdomain of Natural Language Processing concerned with the automatic extraction of structured, factual information from unstructured or semi-structured machine-readable texts. Since it has been shown that a number of subfields of IE, such as question answering (Riloff et al., 2003) and IE from biomedical texts (Medlock and Briscoe, 2007; Szarvas, 2008), benefit from being able to distinguish facts from unreliable or uncertain information, research about hedge cue detection has increased in recent years.

Hedge cues are linguistic devices that indicate whether information is being presented as uncertain or unreliable within a text (Lakoff, 1973; Hyland, 1998). They are lexical resources used by the author to indicate caution or uncertainty towards the content of the text, and in this sense they can be taken as signals of the presence of an author's opinion or attitude. Hedge cues can be expressed by several word classes: modal verbs (e.g., *can, may*), verbs (e.g., *seem, appear*), adjectives (e.g., *possibly, likely*), etc. Furthermore hedge cues can be expressed by multiword expressions, i.e., expressions that contain more than a word, with a non-compositional meaning. The latter means that the meaning of the expression cannot be derived from the individual meanings of the words that form the expression. This can be seen from Example 4.1[1], where *call into question* is a multiword hedge cue.

Example 4.1. The low results {**call into question** the applicability of this method}.

[1] All example sentences in this chapter were taken from the CoNLL 2010-Shared Task dataset (Farkas et al., 2010), which will also be used for the evaluation of our approach and is discussed in Section 4.4.1.

Neither the verb *call* nor the noun *question* are hedge cues on their own, but the whole phrase conveys a speculative meaning, which explains why the sentence would be marked as a hedged sentence.

Recently, the NLP community has shown interest in problems that involve analysing language beyond the propositional meaning of sentences, i.e., whether the sentence is true or false. Apart from performing well-established NLP tasks such as parsing or semantic role labeling, there is a growing interest in tasks that involve processing non-propositional aspects of meaning, i.e., opinions, attitudes, emotions, or figurative meaning. To perform these tasks, the local token-based approaches based on the lexico-syntactic features of individual words no longer suffice. The broader context of words on the sentence or discourse level has to be considered in order to account for aspects of meaning that are expressed by certain combinations of words, like "call into question" in the sentence above. Performing hedge cue detection involves finding the linguistic expressions that express hedging. In many cases it is not possible to know whether a word belongs to a hedge cue without taking into account its context, i.e., one needs to consider the part of the text that surrounds this word and determines its meaning.

This formed our motivation to explore the use of graph kernel–based relational learning with kLog. In this chapter we will study its use for sentence classification using sentence context. The question we would like to answer in this chapter is whether a logical and relational learning approach is able to process contextual aspects of language at the sentence level. As we will see, the results indicate that kLog is suitable for this task.

4.2 Related Work

Although the term *hedging* was already introduced by Lakoff in 1973 (Lakoff, 1973), and has been studied in theoretical linguistics for two decades (Hyland, 1998), the interest from the computational linguistics community has only arisen in recent years. Light et al. (2004) introduced the problem of identifying speculative language in bioscience literature. The authors used a hand-crafted list of hedge cues to identify speculative sentences in MEDLINE abstracts. They also presented two systems for the automatic classification of sentences in abstracts; one based on SVMs, the other one based on substring matching. Medlock and Briscoe (2007) extended this work and discussed the specificities of hedge classification as a weakly supervised machine learning task for which they presented a probabilistic learning model. In addition, they offered an improved and expanded set of annotation guidelines and provided a publicly available dataset. Based on this work, Medlock (2008) carried out experiments using an expanded feature space and novel representations. Szarvas (2008) followed

Medlock and Briscoe (2007) in classifying sentences as being speculative or non-speculative. He developed a Maximum Entropy classifier that incorporates bigrams and trigrams in the feature representation and performs a reranking-based feature selection procedure. Kilicoglu and Bergler (2008) applied a linguistically motivated approach to the same classification task by using knowledge from existing lexical resources and incorporating syntactic patterns. Additionally, hedge cues were weighted by automatically assigning them an information gain measure and semi–automatically determined weights based on their types and centrality to hedging.

Ganter and Strube (2009) were the first ones to develop a system for automatic detection of sentences containing *weasels* in Wikipedia. As Ganter and Strube indicated, weasels are closely related to hedges and private states, i.e., states that are not open to objective observation or verification. They experimented with two classifiers, one based on words preceding the weasel and another one based on syntactic patterns. The similarity of the results of the two classifiers on sentences extracted from Wikipedia showed that word frequency and distance to the weasel tag provide sufficient information. However, the classifier that used syntactic patterns outperformed the classifier based on words on data manually re-annotated by the authors, suggesting that the syntactic patterns detected weasels that had not yet been tagged.

The increased attention to hedge detection is reflected by the fact that it became a subtask of the BioNLP-Shared Task in 2009 (Kim et al., 2009) and the topic of the Shared Task at CoNLL 2010 (Farkas et al., 2010). The latter comprised two levels of analysis: the goal of task 1 is to learn to detect sentences containing uncertainty, whereas the objective of task 2 is to resolve the in-sentence scope of hedge cues. We will focus on task 1. As noted by Farkas et al. (2010), the approaches to this task can be classified into two major categories. Several systems approached the problem as a sentence classification problem and used a bag-of-words (BoW) feature representation. Alternatively, the individual tokens of the sentence can be classified one by one, instead of classifying the sentence as a whole. In a postprocessing step, the sentences that contain hedge cues are then classified as uncertain.

During the shared task, i.e., an evaluation of several systems on a particular topic that allows participating systems to be compared in a systematic way, Georgescul (2010) obtained the best score for the closed task[2] of in-domain Wikipedia hedge cue detection with a macro-averaged F1-score of 75.13%.[3]

[2] A shared task often comprises two different tracks; a *closed* and a *open* one. In the former, only the data provided by the organizers of the task can be used, while in the latter, systems can make use of additional resources.

[3] This equals an F1-score on the UNCERTAIN class of 60.17%, but we prefer reporting the macro-averaged F1-score because it takes the performance on both class labels into account.

Interestingly, Georgescul (2010) also reports the scores for a simple but effective baseline algorithm: if a test sentence contains any of the hedge cues occurring in the training corpus, the sentence is labeled as UNCERTAIN. For the Wikipedia test data this baseline system obtains a macro-averaged F1-score of 69%. During the shared task and for the Wikipedia data, only the top 3 is able to do better than the baseline on the UNCERTAIN class.

The system of Georgescul (2010) obtained the best score in the shared task although it does not use any intricate features. Each hedge cue of the training set is taken as a feature, and prediction occurs directly at the sentence level. For generalization, the set of hedge cues is extended with n-gram subsets of the cues. In a sense, this system resembles a bag-of-words approach.

Chen and Di Eugenio (2010) propose a two-staged model, enabling them to incorporate token/sentence context and metadata information. Their system, reaching the second position with a macro-averaged F1-score of 73.36%, detects the hedge cues in a separate first step. The second step is an aggregation step in which the predicted cues are used to predict whether a sentence is CERTAIN or UNCERTAIN.

The definition of the hedge cue detection task resembles a multiple-instance learning problem (Dietterich et al., 1997). In (binary) multiple-instance learning, objects consist of multiple instances or features vectors. An object is labeled positive if at least one instance of the object is labeled positive, otherwise the object is labeled negative. However, the multiple-instance problem relies on the notion that there is a region in the multidimensional space that contains at least one instance of each positive object and no instance of any negative object. In our research, many tokens can be a cue making a sentence UNCERTAIN, but an a priori motivation for the assumption that the different cues share a single, confined region of the vector space cannot be given.

4.3 Method

As outlined in the introduction, the increasing importance of relational information for current natural language processing tasks requires an appropriate representation for encoding this type of features. This motivated the use of the graph-based relational representation and feature construction technique of kLog for modeling the hedge cue detection problem. It offers a way of taking into account the necessary context at sentence-level and is able to represent both the lexico-syntactic information as well as the sequence information and dependency relationships. Furthermore, its declarative approach, which is characteristic of SRL systems, offers a flexible experimentation approach and an interpretable representation, which is especially useful from a computational linguistics point

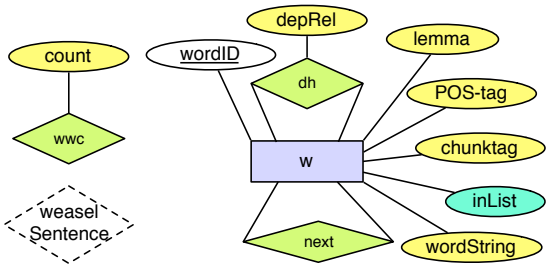


Figure 4.1: E/R diagram modeling the hedge cue detection task[4].

of view.

4.3.1 Data Modeling

As introduced in Section 2.2.3, kLog is built upon a logical and relational data representation rooted in the entity-relationship model. For the problem under consideration, the E/R-model is shown in Figure 4.1. It gives an abstract representation of the interpretations, which are sentences in the current problem. A sentence consists of a number of consecutive words `w`, for which the order is represented by the `next` relation. There are also dependency relations between certain words, which represent the structure of syntactic relations between the words of a sentence. This is modeled by the `dh` relation, where its property `depRel` specifies the type of the dependency. To illustrate this, consider the following example of a hedged (i.e., uncertain) sentence:

Example 4.2. Often the response variable may not be continuous but rather discrete.

In this sentence, a dependency relation exists between the determiner *the* and the noun *variable*, where the first is a noun modifier of the latter.

Other properties of the word that are taken into account as features are the word string itself, its lemma, the part-of-speech tag (i.e., the linguistic type of the word in the sentence), the chunk tag (which indicates that a word is part of a subsequence of constituents) and a binary feature that represents whether the word is part of a predefined list of speculative strings. `wwc` is a property of the sentence as a whole, and represents the number of weasel words in the sentence. `weaselSentence` represents the target relation that we want to predict.

[4]Note that both `wwc` and `weaselSentence` are relations of zero relational arity, which are used to represent global properties of an interpretation.

```
wwc(2).                                    next(w2,w3).
w(w1,often,rb,i-advp,1,often).             next(w3,w4).
w(w2,the,dt,i-np,0,the).                   next(w4,w5).
w(w3,response,nn,i-np,0,response).         dh(w1,w5,adv).
w(w4,variable,nn,i-np,0,variable).         dh(w2,w4,nmod).
w(w5,may,md,i-vp,1,may).                   dh(w3,w4,nmod).
next(w1,w2).                               dh(w4,w5,sbj).
...
```

Listing 4.1: Example of a partial interpretation z.

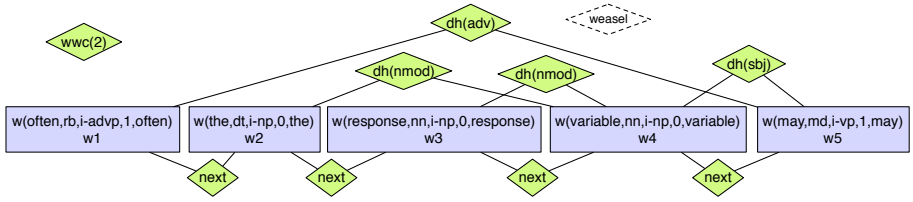


Figure 4.2: Graphicalization G_z of (partial) interpretation z (Listing 4.1)

This E/R model representation can be transformed into a kLog script that describes (the structure of) the data. Listing 4.1 shows a (partial[5]) example interpretation z, that is a grounded version of the E/R-model as shown in Figure 4.1 for a part of the sentence in Example 4.2. For example, `w(w1,'often',rb,i-dvp,1,'often')` specifies the entity for the word *often* with identifier `w1`, POS-tag *rb* (denoting an adverb), chunk tag *i-dvp* (indicating an adverbial phrase), which is in the predefined list of speculative strings and has an identical lemma. The atom `next(w1,w2)` represents the sequence relation between words `w1` and `w2`. The atom `dh(w2,w4,nmod)` represents the dependency relation from the example above, namely that word `w2` (*the*) is a noun modifier (`nmod`) of word `w4` (*variable*).

These interpretations are then graphicalized, i.e., transformed into graphs. This can be interpreted as unfolding the E/R diagram over the data, for which an example is given in Figure 4.2. It represents the graphicalization of the partial interpretation in Listing 4.1.

This forms the input to the next level, where the NSPDK graph kernel is applied to convert these graphicalized interpretations into extended, high-dimensional feature vectors. This is illustrated in Figure 4.3, showing a graphicalization

[5]Note that we refer to the example interpretation as partial, since – for the ease of representation – it does not contain the full sentence.

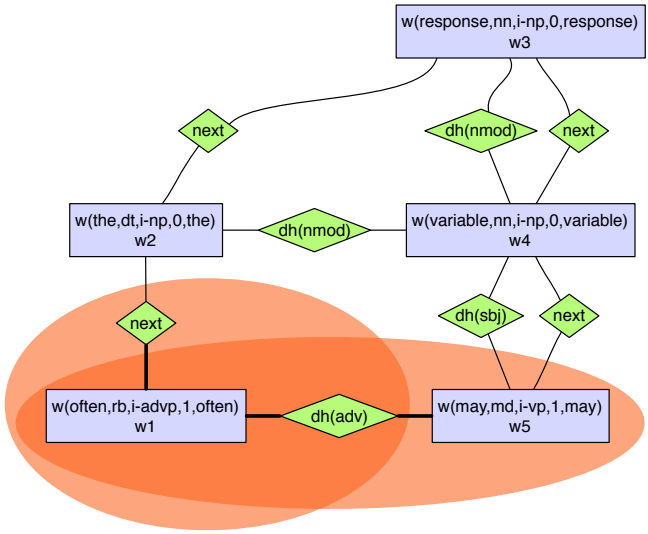


Figure 4.3: Illustration of the NSPDK subgraph concept on graphicalization G_z with hyperparameters distance 1 and radius 1, and word entity `w1` and dependency relation `dh(adv)` as current kernel points.

G_z of (partial) interpretation z in which the entities of the word *often* and the dependency relation *dh(adv)* are the current kernel points of the neighborhood subgraphs indicated in orange. The distance indicates the number of hops between the two subgraphs that are to be compared for feature generation. Distance 1 implies that two subgraph are only compared to one another if their respective centers are one hop or less away from each other. The radius determines the size of the subgraph. The edges in bold indicate the span.

Modeling also plays an important role, which is demonstrated by means of the dependency relation `dh`. More pairs of words are taken into account than just the regular bigrams, for which the words need to be adjacent. As can be seen, also the (non-adjacent) words $w1$ and $w5$, respectively *often* and *may*, are considered through the `dh` relation. This takes more context into account, which demonstrates the power of the graph-based approach of kLog. Also the background knowledge can have an impact on the results, as we will discuss next.

The result is a propositional learning setting, for which any statistical learner can be used. In this case, we used an SVM for parameter learning.

```
cw(CW,L,P) ← w(W,_,P,_,1,L), atomic_concat(c,W,CW).
leftof(CW,L,P) ← cw(W,_,_), atomic_concat(c,W,CW), next(W1,W),
    w(W1,_,P,_,_,L).
rightof(CW,L,P) ← cw(W,_,_), atomic_concat(c,W,CW), next(W,W1),
    w(W1,_,P,_,_,L).
```

Listing 4.2: Additional background knowledge for the hedge cue detection task.

4.3.2 Background Knowledge

kLog is built on deductive databases. This means that besides listing the tuples and atoms of an interpretation explicitly, they can also be deduced using rules. In kLog this is achieved by using intensional signatures, whereby tuples can be defined through definite clauses as in Prolog. This is very useful for introducing additional background knowledge into the learning process. Since the newly constructed ground facts in the database are used to construct graphs from which features are derived during graphicalization, this amounts to the ability of constructing features in a declarative fashion. We introduced the piece of background knowledge shown in Listing 4.2. The first predicate retains each word that appear in a predefined list of weasel words compiled from the training data (i.e., all words for which the `inList` property is equal to 1), together with its lemma and POS-tag. Furthermore, also the two surrounding words in the sentence and their respective lemmas and POS-tags are retained.

4.4 Evaluation

Before turning to the obtained results, we will introduce the CoNLL 2010-Shared Task dataset which was used for the experiments, briefly outline the preprocessing steps and discuss the influence of the kLog hyperparameters.

4.4.1 Dataset

For our experiments, we used the CoNLL 2010-Shared Task dataset (Farkas et al., 2010) on Wikipedia, one of the current benchmark datasets for hedge cue resolution. The Wikipedia paragraphs were selected based on the weasel tags that were added by the Wikipedia editors, and were subsequently manually annotated. The proportion of training and test data, and their respective class ratios can be found in Table 4.1.

	Train	TrainDS	Test
CERTAIN	8,627	2,484	7,400
UNCERTAIN	2,484	2,484	2,234
total	11,111	4,968	9,634

Table 4.1: Number of sentences per class in the training, downsampled training (Section 4.5) and test partitions (CoNLL 2010-Shared Task, Wikipedia dataset).

Preprocessing

To preprocess the dataset, the approach of Morante et al. (2010) was followed, in which the input files where converted into a token-per-token representation, following the standard CoNLL format (Buchholz and Marsi, 2006). In this format a sentence consists of a sequence of tokens, each one starting on a new line. Consequently the data was processed with the Memory-based Shallow Parser (MBSP) (Daelemans and van den Bosch, 2009) in order to obtain lemmas, part-of-speech tags, and syntactic chunks, and with the MaltParser (Nivre, 2006) to obtain dependency trees.

4.4.2 Parametrization

From the kernel definition it follows that the distance and radius parameters strongly influence the results. Consequently, it is important to make a deliberate choice during parametrization. For the task at hand, expert knowledge and the literature suggest using bigrams (one word before *or* one word after the word in focus) or trigrams (one word before *and* one word after the word in focus), since unigrams include too little context, and 5-grams introduce too much noise. To this end, we performed a 10-fold cross-validation on the training set, using all combinations of distances 0, 1, 2 and radii 0, 1, 2 for the kLog hyperparameters. The setting with both distance and radius set to 1 gave the best cross-validation results (an F-measure of 60.59, where we took 60.2, the F-measure of the top performing system in the CoNLL 2010-Shared Task, as decision threshold). As kernel points, the center word relation `cw`, the dependency relation `dh`, and the count of the number of weasel words at sentence level `wwc` were taken.

As indicated before, the result is a propositional learning setting for which any statistical learner can be used. In this case, we used the linear kernel of LibSVM (Chang and Lin, 2011), for which we optimized the regularization parameter and the class weighting parameters as part of the cross-validation process.

Official Rank	System	Precision	Recall	F1
-	**kLog**	67.04	56.77	61.48
1	Georgescul	72.0	51.7	60.2
2	Ji[7]	62.7	55.3	58.7
3	Chen	68.0	49.7	57.4
4	Morante	80.6	44.5	57.3
5	Zhang	76.6	44.4	56.2

Table 4.2: Evaluation performance on the test set in terms of precision, recall and F1 of the top 5 CoNLL 2010-Shared Task systems and the kLog approach for the Wikipedia dataset.

4.4.3 Results

The results of our approach are listed in Table 4.2, together with the results of the five best listed participants in the CoNLL 2010-Shared Task.[6] As can be noted, kLog outperforms the systems in terms of F-measure.

We also calculated the scores of our system without the additional background knowledge as discussed in the previous section. This resulted in a decrease of 2.66 in F-measure, from 61.48 to 58.82, which shows the advantage of the declarative feature construction. This is - combined with the powerful graph kernel - one of the main strengths of kLog. We will now turn to a detailed analysis of the influence of using a relational representation.

4.5 Advantage of a Relational Representation

Thanks to its focus on relations between abstract objects, graph kernel–based relational learning offers the possibility to model a problem on different levels simultaneously, and provides the user with the possibility to represent the problem at the right level of abstraction. For example, sentence classification can be carried out using instances on the token level, without having to resort to a two-step system in which the first step consists of labeling the tokens and the second step is an aggregation step to reach a prediction on the sentence level. Attributes on a higher level, e.g., sentences, can be predicted on the

[6]Note that our system does not have an official rank, as it was developed after the shared task and thus was not part of the official evaluation during the task. However, the results were calculated using the task's official scorer.

[7]Note that this system used a cross-dataset approach, in which also the CoNLL 2010-Shared Task biological dataset was used to train the system.

basis of lower level subgraphs, e.g., sequences of tokens, taking into account the relations in the latter, e.g., the dependency tree.

The goal of this section is to offer new insights into the advantages of using a declarative, relational representation for natural language processing problems. To this end, we will compare several machine learning techniques along two dimensions on the hedge cue detection task.

The first is concerned with whether they deal with a *propositional* or a *relational* representation. Learning techniques can also be distinguished along another dimension that indicates whether they are *eager* or *lazy*. As introduced in Section 2.1.2, eager techniques (such as SVMs) compute a concise model from the data, while lazy (or memory-based) learning (MBL) techniques simply store the data and use (a variant) of the famous kNN algorithm to classify unseen data. Today, eager methods are much more popular than memory-based ones. Nevertheless, it has been argued (Daelemans and van den Bosch, 2009) that MBL is particularly suited for NLP, since language data contains in addition to regularities, many subregularities and productive exceptions. That is, low-frequency or atypical examples are often not noise to be abstracted from in the model, but an essential part of it. *Lazy* learning may identify these subregularities and exceptions, while *eager* learning often discards them as noise. MBL has proven to be successful in a wide range of tasks in computational linguistics (e.g., for learning of syntactic and semantic dependencies (Morante et al., 2009b) and event extraction (Morante et al., 2009a)).

As part of this comparison of propositional versus eager learners, we will first introduce a novel relational memory-based learning technique, together with a brief overview of related work in this area.

4.5.1 Relational Memory-based Learning

A number of approaches have combined relational and instance-based learning. RIBL (Emde and Wettschereck, 1996) is a *relational instance-based learning* algorithm that combines memory-based learning with inductive logic programming. It was extended by Horváth et al. (2001) to support representations of lists and terms. Besides the standard similarity measures for numerical and discrete attributes, this version also incorporated a similarity measure based on the concept of edit distance for attributes with lists and terms. Armengol and Plaza (2001) introduced Laud; a distance measure that can be used to estimate similarity among relational cases. Since it only considered the leaves of this structure, Shaud (Armengol and Plaza, 2003) was proposed as an improvement that was able to take into account the complete structure provided by the feature terms. Ramon (2002) proposes a set of methods to

perform instance-based learning using a relational representation language, and extends distances and prototypes to more complex objects.

kLog can be considered as a logical language for feature generation, starting from a relational learning problem, after which these features can be used in a statistical learner. This *explicit* combination of *relational with eager learning* is relatively rare. Traditional SRL systems such as Markov Logic or Bayesian Logic Programs (Kersting and De Raedt, 2007) are related in that they implicitly result in a graphical model for each instance representing a class of probability distributions. Also generalizations of SVMs for relational structures and the use of SVM-like loss functions in SRL have been explored (e.g., max-margin Markov networks (Taskar et al., 2003)).

In order to construct a relational memory-based learner, the relational information constructed with kLog, and MBL are combined, using the NSPDK graph kernel as a relational distance measure. The similarities between the instances are readily available from the kernel matrix (also known as the *Gram matrix*), which is calculated by the graph kernel, and thus can be exploited efficiently. A kernel κ can be easily transformed into a distance *metric*, using $d_\kappa(x,y) = \sqrt{\kappa(x,x) - 2\kappa(x,y) + \kappa(y,y)}$. This will be referred to as *kLog-MBL*. We both employed a regular kNN setup (Fix and Hodges, 1951), referred to as *kLog-MBL (NW)*, as well as a distance-weighted variant (Dudani, 1976), referred to as *kLog-MBL (W)*. In the latter, a neighbor that is close to an unclassified observation is weighted more heavily than the evidence of another neighbor, which is at a greater distance from the unclassified observation.

4.5.2 Dataset

As can be seen from Table 4.1, the data is unbalanced. This can lead to different issues for machine learning algorithms (Van Hulse et al., 2007). For memory-based learning, the majority class tends to have more examples in the k-neighbor set, due to which a test document tends to be assigned the majority class label at classification time. As a result, the majority class tends to have high classification accuracy, in contrast to a low classification accuracy for the minority class, which affects the total performance and partly obfuscates the influence of the distance measure (Tan, 2005).

Since the goal is to show the influence of the relational representation and distance measure, we want to reduce the influence of the imbalancedness of the dataset. Several approaches have been proposed to deal with this (i.e., adjusting misclassification costs, learning from the minority class, adjusting the weights of the examples, etc.). One of the two most commonly used techniques to deal with this problem is sampling (Chawla, 2005), where the training dataset

is resized to compensate for the imbalancedness. We created a downsampled version of the training set. This was done in terms of the negative examples (the CERTAIN sentences), i.e., we sampled as many negative examples as there are positive examples. We will refer to this dataset as *TrainDS*. The number of sentences per class in this dataset is listed in Table 4.1.

4.5.3 Baseline and Benchmarks

The goal is to examine the behavior of a system that uses a relational, graph-based representation to classify the sentence as a whole (i.e., using the graphicalization process) and contrast it with lazy and eager learning systems that do not use this extra step. For this reason, several baselines and benchmarks are included in the result table (Table 4.3).

The first, simple baseline is a system that labels all sentences with the UNCERTAIN class. This enables us to compare against a baseline where no information about the observations is used.

The first group of benchmarks consists of systems that operate without relational information. These systems typically use a two-step approach; first the individual words in the sentence are classified, and then the target label for the sentence is determined based on the number of tokens that are labeled as hedge cues. This requires an extra parameter to threshold this number of individual tokens from which the sentence label is derived (i.e., if more than $X\%$ of the token-level instances are marked as being a hedge cue, the sentence is marked as UNCERTAIN). To optimize this parameter, the training set was split in a reduced training set and a validation set (70/30% split). The influence of this parameter is discussed in more detail in Section 4.5.5.

The first of these systems is the Tilburg Memory-based Learner[8] (TiMBL), a software package implementing several memory-based learning algorithms, among which IB1-IG, an implementation of k-nearest neighbor classification with feature weighting suitable for symbolic feature spaces, and IGTree, a decision-tree approximation of IB1-IG. We will use it in the same setup and with the same feature set as Morante et al. (2010). They used 5% as the percentage threshold for sentences, however, our optimization procedure yielded better results with a 30% threshold. In the result table, these variants are referred to as *TiMBL (5%)* and *TiMBL (30%)*.

In order to parameterize TiMBL for the word classification, we used paramsearch[9] (van den Bosch, 2004), which is a wrapped progressive sampling approach for algorithmic parameter optimization for TiMBL. The IB1 algorithm

[8] http://ilk.uvt.nl/timbl/
[9] http://ilk.uvt.nl/paramsearch/

was chosen as optimal setting, which is the standard MBL algorithm in TiMBL. To parameterize the algorithm, there is a choice of metrics that influence the definition of similarity. The overlap metric was chosen as the best one for this task, which assigns a weight to each feature, determining its relevance in solving the task. The relevance weights are computed with gain ratio, and the number of most similar memory patterns on which the output class is based was set to 5. Furthermore, the neighbors are not weighted as a function of their distance. Note that since we use a different, balanced training corpus, the results reported in Table 4.3 are not directly comparable with the results by Morante et al. (2010).

As support vector machines[10] are considered one of the most prominent methods for NLP tasks today, they will be used here as a representative eager learning method. In a first step the (non-graphicalized representation of the) data is converted into binary feature vectors.[11] Subsequently, the SVMs are optimized in terms of the cost parameter C (and *gamma* in the case of the RBF kernel) and feature normalization using a grid search with 10-fold cross-validation on the reduced training set. Hereafter the percentage threshold is optimized on the validation set. The SVM without relational information is referred to as *SVM* in the result tables.

We contrasted these systems with a lazy and eager learning approach that use the graph-based relational representation from kLog. For *kLog-SVM*, we used an SVM as statistical learner at the end of the kLog workflow. *kLog-SVM* reports the results when using the parameter settings as described in Section 4.4.2, for which the cost parameter of the SVM was optimized using cross-validation on the downsampled training set.

The second pair of relational systems uses memory-based learning. To this end, the relational information constructed with kLog and memory-based learning are combined, as discussed in Section 4.5.1. The value of k was optimized using the reduced training and validation set as discussed above.

4.5.4 Performance

Table 4.3 contains the macro-averaged F1-scores of these seven systems. Looking at the table, one may conclude that all systems perform better than the UNCERTAIN baseline.

[10] We used the implementation available in scikit-learn (Pedregosa et al., 2011), which in turn is based on libsvm (Chang and Lin, 2011).

[11] The exact implementation is available at `http://www.cnts.ua.ac.be/~vincent/scripts/binarize.py`

The systems are best compared in a pairwise manner. A first interesting observation is that the memory-based learners that use a relational representation (i.e., *kLog-MBL*), perform better than those that use a propositional approach, i.e., the *TiMBL* systems. The weighted and unweighted variants of *kLog-MBL* score equally well.

	Baseline	TiMBL (5%)	TiMBL (30%)	SVM
Precision	11.59	65.65	69.45	73.03
Recall	50.00	67.83	76.76	79.00
F1-score	18.82	50.92	69.34	74.59
	kLog-MBL (NW)	kLog-MBL (W)	kLog-SVM	
Precision	73.02	73.02	75.37	
Recall	75.24	75.24	73.99	
F1-score	73.96	73.96	74.63	

Table 4.3: General evaluation of the different systems on the CoNLL 2010-Shared Task Wikipedia corpus with downsampled training set. All scores are macro-averaged.

For the SVM setups, *SVM* and *kLog-SVM* score equally well. At first sight, the relational representation does not appear to add much to the performance of the learner. However, when comparing the relational approach on the full (unbalanced) dataset to an *SVM* using the propositional representation, *kLog-SVM* performs better (Table 4.4). Furthermore, when comparing the results of the regular *SVM* on the balanced and full dataset, a decrease in performance is observed, which is not present for the *kLog-SVM* setup. This indicates that the relational representation increases the generalization power of the learner. We also compared the *kLog-SVM* setup to the best scores obtained during the CoNLL 2010-Shared Task, *viz.* Georgescul (2010), in which *kLog-SVM* is able to slightly outperform the state-of-the-art system. This can be attributed to the relational representation, which offers the possibility to model the sentence as a whole and perform the classification in a single step (i.e., avoiding the need for a two-step approach where first token-based classification is performed followed by a thresholding step to obtain the sentence-level classification). We will study this effect in more detail in the next section. In addition, the relational representation is able to model the relations between the words in the sentence explicitly. The graph kernel thus appears to provide a good way to translate the context of the words in a sentence.

	Georgescul (2010)	SVM	kLog-SVM
Precision	79.29	81.59	77.27
Recall	72.80	68.96	74.16
F1-score	75.13	72.17	75.48

Table 4.4: Comparison of kLog-SVM with the state-of-the-art results and an SVM using the propositional representation on the full dataset. All scores are macro-averaged.

4.5.5 Level of Abstraction

The graph-based representation has the advantage that attributes on a higher level, e.g., sentences, can be predicted on the basis of lower-level subgraphs, e.g., tokens. It furthermore enables the learner to take into account the relations in the latter, e.g., the dependency tree. This leads to a one-step classification, without the need for an additional thresholding parameter to go from the lower-level classification (e.g., the classification of the individual tokens) to the higher level (e.g., the sentences). The goal of this section is to show when sentence-based systems are more fit for the task than token-based systems.

The baseline system predicts only one type of class label, namely the UNCERTAIN class. The other systems label sentences with both labels and apart from the observation that one system is more inclined to assign the CERTAIN label than the other, the general scores are not of much help to get more fundamental insights. For this reason, an extra dimension is introduced, namely sentence length. It is an intuitive dimension and other dimensions, like the number of uncertainty cues, are indirectly linked to the sentence length. Figure 4.5 shows the *evolution* of the macro-averaged F1-score when the sentences to be labeled contain more tokens. To create this figure, the sentences are distributed over 9 bins centered on multiples of 10. The last bin contains all larger sentences.

Figure 4.4 shows the fraction of the corpus that is included in each bin (solid line) and the fraction of sentences in each bin that is labeled as UNCERTAIN (dashed line). There are fewer long sentences, and long sentences tend to be labeled as UNCERTAIN. As a sanity check, we can look at the behavior of the baseline system in Figure 4.5. The observation of an increasing number of UNCERTAIN sentences with increasing sentence length (Figure 4.4) is consistent with the increasing F1-score for the baseline system in Figure 4.5.

A more interesting observation is the curve of *TiMBL (5%)*, which quickly joins the baseline curve in Figure 4.5. Although this system performs better than the baseline system, it behaves like the baseline system for longer sentences.

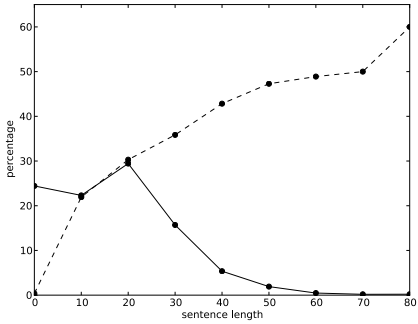


Figure 4.4: The fraction of sentences in the Wikipedia test corpus with a given sentence length (solid line) and the proportion of UNCERTAIN sentences in each bin (dashed line).

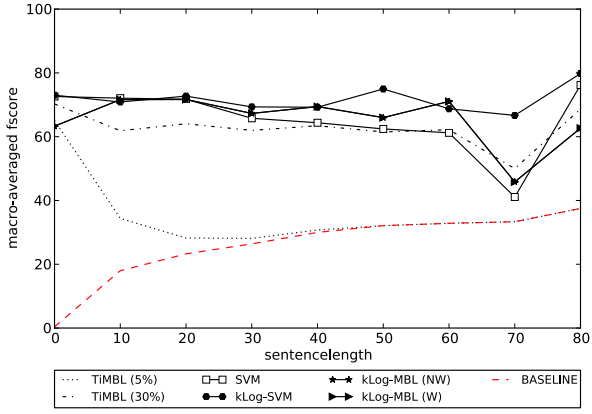


Figure 4.5: Macro-averaged F1-score as a function of sentence length, expressed in number of tokens.

Because a large fraction of the sentences is short, this undesirable behavior is not readily noticeable when examining the scores of Table 4.3. Optimizing the threshold can be a solution to this problem. Changing the threshold influences the chance of a sentence being labeled as UNCERTAIN depending on sentence length. Increasing the threshold leads to a more unequal distribution of this chances over sentence length. As a result, the behavior of the optimized TiMBL system is more stable with varying sentence length (see *TiMBL (30%)*).

The token-based systems (*SVM* and *TiMBL (30%)*) behave very similarly after optimization of the threshold. Indeed, the *SVM* and optimized *TiMBL* curves follow more or less the same course; a course that is different from the other

systems. This indicates that by using a two-step approach, the choice of the classifier is of lesser importance. This behaviour is also noticeable for *kLog-MBL* in the case of longer sentences, albeit to a more limited extent. However, when contrasting the kLog-based systems, the curves of the *kLog-SVM* and *kLog-MBL* systems diverge for longer sentences, indicating the importance of the classifier.

The importance of the threshold parameter for the propositional, two-step approaches (*TiMBL (5%)*, *TiMBL (30%)* and *SVM*) may be an argument to opt for relational systems. Indeed, the threshold is an extra parameter that has to be learned during training, and it may introduce errors because of its rigidity. It is the same fixed value for all sentences, and it weakens the, possibly positive, influence of the classifier. The kLog-based systems do not require such a threshold and are thus able to *dynamically* look for the best prediction on sentence level using the dependencies between the separate tokens.

The claim that dynamically looking for the best prediction on sentence level is better, is based on the observation that, in general, the *kLog*-based systems perform better than their non-relational counterparts. For the dataset under consideration, the *SVM* system does not perform differently in F1 than the *kLog-SVM* system; but if we look at their behavior in Figure 4.5, we see that for almost all sentence lengths *kLog-SVM* performs better. Furthermore, as shown in Section 4.5.4, *kLog-SVM* generalizes better to the unbalanced version of the dataset when compared to *SVM*, and also produces the most stable predictions across all sentence lengths.

4.6 Conclusions

The increasing importance of relational information for current (semantic) natural language processing tasks requires an appropriate representation to encode this type of features. In this chapter, we studied the use of graph kernel–based relational learning for hedge cue detection, a binary sentence classification task. The graph-based feature construction approach of kLog proves to be well-suited to represent contextual, relational information at the sentence level. Furthermore, its declarative feature representation offers a flexible and interpretable experimentation approach, which enables the introduction of additional background knowledge.

Subsequently, we used the task of hedge cue detection to evaluate several types of machine learning systems along two dimensions; the relational representation was contrasted with propositional approaches for both lazy and eager learning. The results show that relational representations are useful, especially for dealing with sentences in which complex, long-distance dependencies amongst constituents need to be captured. The relational representation also enables one-step

classification, without the need for an additional thresholding parameter to go from word-level to sentence-level predictions. As part of this investigation, a novel lazy relational learning technique was proposed, which shows that the relational feature construction approach of kLog can be used in both an eager type of learner as well as in a memory-based learning setting.

Chapter 5

Sentence Classification with Document Context

Motivated by the results from the previous chapter, where we showed that a statistical relational learning approach using kLog is able to process the contextual aspects of language improving on state-of-the-art results for hedge cue detection, in this chapter we will expand the context to be taken into account to the document level.

The task under consideration is the identification of evidence-based medicine categories. *Evidence-based medicine* is an approach in which clinical decisions are supported by the best available findings gained from scientific research. This requires efficient access to such evidence. To this end, abstracts of publications in evidence-based medicine can be labeled using a set of predefined medical categories, the so-called *PICO* criteria. We will present an approach to automatically annotate sentences in medical abstracts with these labels. As indicated by Kim et al. (2011), both the structural information of the words in the sentence, and that of the sentences in the document are important features for this task. Furthermore, sequential information can leverage the dependencies between different sentences in the text. To illustrate this, consider the following two subsequent sentences:

Example 5.1. Subfoveal choroidal neovascular membranes (CNV) are a cause of significant visual impairment. Laser treatment of such lesions results in visual loss.

The sequence of the words in the sentence is important to determine that

both sentences deal with the same subject, namely visual impairment. The structure of the sentences helps to identify the relations between them, namely that *such lesions* probably refers to the injury described in the first sentence, namely *subfoveal choroidal neovascular membranes*. Furthermore, from the used verbs, it may be clear that the first sentence describes a cause, whereas the second states an effect. The combination of this contextual information is an indication that both sentences have the same class label. In this case, both sentences are describing background information on the performed medical research. Integrating this information requires an adequate representation, which motivates the use for a graph kernel–based relational learning approach.

In comparison to the hedge cue detection task, the automatic identification of PICO categories adds two levels of complexity. First, besides the relations between the words in the sentence, now also the relations between the sentences in the document become important. In the proposed approach, we first generate a feature space with kLog that captures the intrasentential properties and relations. Hereafter, these features serve as input for a structured output support vector machine that can handle sequence tagging (Tsochantaridis et al., 2004), in order to take the intersentential features into account. Second, since there are more than two categories, and each sentence can have multiple labels, the problem is now a multiclass multilabel classification task.

The main contribution of this chapter is that we show that kLog's relational nature and its ability to declaratively specify and use background knowledge is beneficial for natural language learning problems where document context is important. This is shown on the NICTA-PIBOSO corpus, for which we present results that indicate a clear improvement on a memory-based tagger. Furthermore, to the best of our knowledge, our approach was the first to outperform the results obtained by the system of Kim et al. (2011), the original benchmark on this task, by integrating structural background knowledge as distinguishing features.

This chapter starts with a discussion of the task, evidence-based medicine category identification in Section 5.1. Subsequently, related work in this area is discussed in Section 5.2. In Section 5.3, the method is outlined, after which it is evaluated in Section 5.4. Finally, Section 5.5 concludes this chapter.

This chapter is based on

> Mathias Verbeke, Vincent Van Asch, Roser Morante, Paolo Frasconi, Walter Daelemans, and Luc De Raedt. A statistical relational learning approach to identifying evidence based medicine categories. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning,*

Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP - CoNLL), Jeju Island, Korea, 13-14 July 2012, pages 579–589. Association for Computational Linguistics, 2012.

5.1 The Task: Evidence-based Medicine Category Identification

Evidence-based medicine (EBM) or *evidence-based practice (EBP)* is an approach to clinical problem-solving based on "systematically finding, appraising, and using contemporaneous research findings as the basis for clinical decisions" (Rosenberg and Donald, 1995). It combines clinical expertise, the preferences and values of the patient and the best available evidence to make good patient care decisions. To this end, clinical research findings are systematically reviewed and appraised. The evidence-based process consists of four steps: (1) Formulating a question based on a patient's problem; (2) Searching the literature for relevant clinical articles; (3) Evaluating the evidence; and (4) Implementing useful findings in clinical practice.

Given the amounts of medical publications available in databases such as PubMed, efficient access to such evidence is required. In order to facilitate the search for scientific evidence in the literature, medical documents are labeled using a set of predefined medical categories, the *PICO criteria* (Armstrong, 1999). The PICO concepts are: primary Problem (P) or population, main Intervention (I), main intervention Comparison (C), and Outcome of intervention (O). The PICO concepts help the medical practitioner determine what terms are important in a query, and therefore offer him support in building the query which is sent to the search repositories. Once the documents are found, they need to be read by a person who eliminates the irrelevant ones.

The search process thus consists of matching a query, which is composed according to the PICO criteria, with scientific evidence found in articles labeled according to the same standard. Consequently, automating the annotation of these research articles would give access to a larger amount of scientific evidence, and could help doctors in their practice. This has initiated research into automatic approaches to annotate sentences in medical documents with the PICO labels. Efforts in this direction from the NLP community so far have focused on corpus annotation (Demner-Fushman and Lin, 2007; Kim et al., 2011), text categorization (Davis-Desmond and Mollá, 2012), and question-anwering (Niu et al., 2003; Demner-Fushman and Lin, 2007). A corpus to train EBM summarization systems was annotated by Mollá and Santiago-Martínez (2011). The corpus contains documents from the "Clinical Inquiries" section

of the *Journal of Family Practice*. For each question the corpus provides information about the evidence-based answer and the answer justifications.

5.2 Related Work

The first attempt to classify sentences labeled with PICO concepts was presented by Demner-Fushman and Lin (2007). They present a rule-based approach to identify sentences where PICO concepts occur and a supervised approach to classify sentences that contain an *Outcome*. The features used by this classifier are n-grams, position, and semantic information from the parser used to process the data. The system is trained on 275 manually annotated abstracts. The reported accuracies range from 80% for *Population*, 86% for *Problem*, 80% for *Intervention*, and up to 95% for *Outcome*. The output of their classification system is used in a relevance-scoring algorithm for MEDLINE citations.

Kim et al. (2011) perform a similar classification task in two steps. First a classifier identifies the sentences that contain PICO concepts. Subsequently, another classifier assigns PICO tags to the sentences found to be relevant by the previous classifier. The system is based on a Conditional Random Fields (CRF) algorithm and is trained on the NICTA-PIBOSO corpus. This dataset contains 1,000 medical abstracts that were manually annotated with an extension of the PICO tagset, for which the definitions are listed in Table 5.1. The annotation is performed at sentence level and one sentence may have more than one tag. An example of an annotated abstract from the corpus can be found in Appendix A. The features used by the algorithm include features derived from the context, semantic relations, structure and sequencing of the text. The system is evaluated for 5-way and 6-way classification and results are provided apart for structured and unstructured abstracts. The F-score for structured abstracts is 89.32% for 5-way classification and 80.88% for 6-way classification, whereas for unstructured abstracts it is 71.54% for 5-way classification and 64.66% for 6-way classification.

Chung (2009) also uses a CRF to classify sentences labeled with PICO concepts by combining them with general categories associated with rhetorical roles: *Aim*, *Method*, *Results* and *Conclusion*. Her system is tested on corpora of abstracts of randomized control trials. First, structured abstracts with headings labeled with PICO concepts are used. A sentence-level classification task is performed, which assigns only one rhetorical role per sentence. The F-scores obtained range from 93% to 98%. Subsequently another sentence-level classification task is performed to automatically assign the labels *Intervention*, *Participant* and *Outcome Measures* to sentences in unstructured and structured abstracts

Background	Material that informs and may place the current study in perspective, e.g., work that preceded the current; information about disease prevalence; etc.
Population	The group of individual persons, objects or items comprising the study's sample, or from which the sample was taken for statistical measurement.
Intervention	The act of interfering with a condition to modify it or with a process to change its course (includes prevention).
Outcome	The sentence(s) that best summarizes the consequences of an intervention.
Study Design	The type of study that is described in the abstract.
Other	Any sentence not falling into one of the other categories and presumed to provide little help with clinical decision making, i.e., non-key or irrelevant sentences.

Table 5.1: Definitions of the semantic tags used as annotation categories (taken from Kim et al., 2011).

without headings. F-scores of up to 83% and 84% are obtained for *Intervention* and *Outcome Measure* sentences.

Other work aimed at identifying rhetorical zones in biomedical articles. In this case, areas of text are classified in terms of the rhetorical categories *Introduction*, *Methods*, *Results* and *Discussion* (IMRAD) (Agarwal and Yu, 2009) or richer categories, such as *problem-setting* or *insight* (Mizuta et al., 2006).

Shortly after our approach was published, the evidence-based medicine category identification problem became the topic of the ALTA 2012-Shared Task (Amini et al., 2012), illustrating the importance of the research problem. The organizers propose both a naive baseline and a benchmark system. The baseline relies on the most frequent label occurring in the training data, given the position of the sentence. The benchmark is based on the approach of Kim et al. (2011), but uses a different feature set. The individual words and their part-of-speech tags are used as lexical features. The structural features are largely similar to the ones we used for the structural abstracts (see Section 5.3.2). In addition to the position of the sentences in the document, for structured abstracts, each sentence is labeled with the rhetorical heading preceding it. Whereas here this labeling needs to be done as a preprocessing step, our approach allows one to declaratively specify these features (see Section 5.3.2).

Two of the top-performing systems use a two-layered architecture, in which the first step is used for feature learning. The system of Lui (2012) uses a stacked

logistic regression classifier with a variety of feature sets. The algorithm learns the importance of each feature set by generating a distribution over the class labels for each training instance. Subsequently, these vectors are concatenated, after which the full feature vectors serve as input for another logistic regression classifier. The test data are then projected into this stacked vector space, in order to be classified. Mollá (2012) presents an approach using cluster-based features. In a first step, a collection of multiple binary classifiers (one per target label) is trained to label the sentences with a set of types. Subsequently, these types are used to cluster the documents based on the distribution of the sentence types. The resulting clusters and some additional features are used to train the final classifiers. Sarker et al. (2013) divide the multi-class classification problem into different binary ones. An SVM using a customized feature set per class label is learned. As the benchmark system provided by the organizers, Gella and Long (2012) present a CRF-based system, but improved it with feature selection and bootstrapping.

Hassanzadeh et al. (2014) propose a combination of different kinds of features and compare different classification methods on this feature space. They combine token-level and sentence-level features in a propositional feature vector that captures both positional as well as sequential information. Furthermore, statistical features under the form of sentence-wide token statistics and inferred sequential features derived from the co-occurrence of similar types of sentences are added. This feature space is tested with four classification methods, namely CRF, SVM, Naive Bayes and Multinomial Logistic Regression, for which the CRF obtains the best results.

Both the top-performing systems of the ALTA 2012-Shared Task and the system of Hassanzadeh et al. (2014) to a certain extent use document-level information. The main difference is that our relational approach allows one to represent all features in a single representation, as indicated in the previous chapter. Furthermore, its declarative nature enables flexible experimentation and interpretable features.

5.3 Method

In the previous chapter, we studied the use of graph kernel–based relational learning for sentence classification using sentence context, which was illustrated using the hedge cue detection task. We showed that the relational representation of the domain is able to take the contextual aspects of language into account. Whereas there we only used the relations at the sentence level, the identification of PICO categories in abstracts also requires one to take into account various relations between the sentences of an abstract. Furthermore, the sentences may have multiple labels, which turns this into a structured output task where

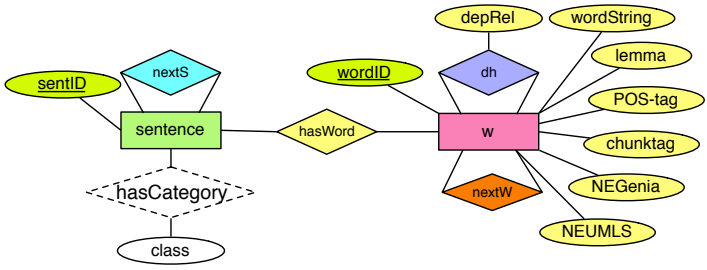


Figure 5.1: E/R diagram of the sentence identification task.

the output is a sequence of sets of labels attached to the sentences in a given document.

5.3.1 Data Modeling

The E/R model for the problem under consideration is shown in Figure 5.1; it provides an abstract representation of the examples, i.e., medical abstracts. As for the hedge cue detection task, this abstract representation can be unrolled for each example, resulting in a graph; cf. Figure 5.2 for an example document. This relational database representation will serve as the input for kLog.

In this case, the *entities* are the words and sentences in the abstract. Each entity again has a number of properties attached to it, depicted by the ovals, and a primary key which serves as a unique identifier (underlined properties). As in database theory, each entity corresponds to a tuple in the database. Listing 5.1 shows an example of a partial interpretation z. For example, `w(w4_1,'Surgical','Surgical',b-np,jj,'O','O')` specifies a word entity, with `w4_1` as identifier and the other arguments as properties. We take the token string itself, its lemma, the part-of-speech tag and the chunk tag into account as lexical information. We also include some semantic information, namely two binary values indicating whether the word is a (biological) named entity (`NEGenia` and `NEUMLS`). The atom `sentence(s4,4)` represents a sentence entity, with its index in the abstract as a property.

Furthermore, the E/R diagram also contains a number of *relations* represented by the diamonds. An example relation is `nextW(w4_2,w4_1)`, which (as was the case for the hedge cue detection task) indicates the sequence of the words in the sentence. The atom `dh(w4_1,w4_2,nmod)` specifies that word `w4_1` is a noun modifier of word `w4_2`, and thus serves to incorporate the dependency relationships between the words. The atom `hasCategory(s4,'background')` signifies that sentence `s4` is a sentence describing background information. This

```
sentence(s4,4).
hasCategory(s4,'background').
w(w4_1,'Surgical','Surgical',b-np,jj,'O','O').
hasWord(s4,w4_1).
dh(w4_1,w4_2,nmod).
nextW(w4_2,w4_1).
w(w4_2,'excision','excision',i-np,nn,'O','O').
hasWord(s4,w4_2).
dh(w4_2,w4_5,sub).
nextW(w4_3,w4_2).
w(w4_3,'of','of',b-pp,in,'O','O').
hasWord(s4,w4_3).
dh(w4_3,w4_2,nmod).
nextW(w4_4,w4_3).
w(w4_4,'CNV','CNV',b-np,nn,'B-protein','O').
hasWord(s4,w4_4).
dh(w4_4,w4_3,pmod).
nextW(w4_5,w4_4).
...
```

Listing 5.1: Part of an example interpretation z, representing the example sentence in Figure 5.2.

relation is the target relation that we want to predict for this task. It will not be taken into account as a feature, but is listed in the database and only used during the training of the model.

In the third step, the interpretations are *graphicalized*. An example illustrating this process is given in Figure 5.2.[1] The obtained graphs can then be used in the next step for *feature generation*, by means of the NSPDK graph kernel. The result of this graphicalization and feature generation process is an extended, high-dimensional feature space, which serves as input for a statistical learner in the next step.

5.3.2 Background Knowledge

As for the hedge cue detection task, we can again introduce additional background knowledge by means of declarative feature construction using intensional signatures. This will prove very helpful for the task under consideration. We make a distinction between the features used for structured and unstructured abstracts.

[1] Note that for the ease of representation, we only show the words with their respective entities. However, their properties (e.g., the lemma, part-of-speech tag, etc.) are also taken into account during graphicalization.

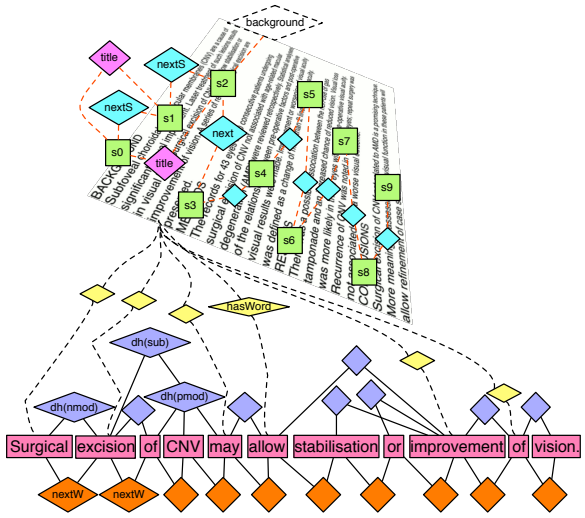


Figure 5.2: Graphicalization G_z of interpretation z.

For structured abstracts, two intensional relations were defined. The relation `lemmaRoot(S,L)` is specified as:

```
lemmaRoot(S,L) ←
    hasWord(S,I),
    w(I,_,L,_,_,_,_),
    dh(I,_,root).
```

For each sentence, it selects the lemmas of the root word in the dependency tree. The following relation tries to capture the document structure imposed by the section headers present in the structured abstracts. First, the helper predicate `hasHeaderWord(S,X)` identifies whether a sentence is a header of a section. In order to realize this, it selects the words of a sentence that contain more than four characters (to discard short names of biological entities), which all need to be uppercase.[2]

```
hasHeaderWord(S,X) ←
    w(W,X,_,_,_,_,_),
    hasWord(S,W),
    (atom(X) -> name(X,C) ; C = X),
    length(C,Len),
    Len > 4,
    all_upper(C).
```

[2]Based on the regular expression for identifying section headers in MEDLINE abstracts from (Hirohata et al., 2008).

Subsequently, all sentences below a certain section header are marked as belonging to this section, which is done by the intensional relation `hasSectionHeader(S,X)`.

```
hasSectionHeader(S,X) ←
   nextS(S1,S),
   hasHeaderWord(S1,X).
hasSectionHeader(S,X) ←
   nextS(S1,S),
   not isHeaderSentence(S),
   once(hasSectionHeader(S1,X)).
```

Since the unstructured abstracts lack section headers, other features are needed to distinguish between the different sections. The `lemmaPOSRoot` relation is similar to the `lemmaRoot` relation which was used for structured abstracts, but, in addition to the lemma, also takes the part-of-speech tag of the root word into account.

```
lemmaPOSRoot(S,L,P) ←
   hasWord(S,I),
   w(I,_,L,_,P,_,_),
   dh(I,_,root).
prevLemmaRoot(S,L) ←
   nextS(S1,S),
   lemmaRoot(S1,L,_).
```

To illustrate the importance of the part-of-speech tag of the root word, consider the following example sentence:

Example 5.2. There were no significant changes in the hormone levels in the SCI subjects throughout the experiment.

In this sentence, the root word *were* has *VBD* as its part-of-speech tag, indicating a verb in the past tense. Taking this into account as a feature during learning increases the similarity of this sentence with sentences that also express a past event. In this case, the use of the past tense can be an indication of a sentence describing the *outcome* of a performed medical experiment, which is the class label of the example sentence. Also the relation `prevLemmaRoot` proves to be very informative. It adds the lemma of the root word in the previous sentence as a property to the current sentence under consideration.

5.3.3 Learning

The constructed feature space contains one feature vector per sentence. This implies that the sequence information of the sentences at the document level is not taken into account yet. Since the order of the sentences in the abstract is a

valuable feature for this prediction problem, a learner that reflects this in the learning process is needed. Therefore we opted for SVM-HMM[3] (Tsochantaridis et al., 2004), which is an implementation of structural support vector machines for sequence tagging (as discussed in Section 2.1.2). In contrast to a conventional Hidden Markov Model, SVM-HMM is able to use these entire feature vectors as observations, and not just atomic tokens.

In our case, the instances to be tagged are formed by the sentences for which feature vectors were created in the previous step. The *qid* is a special feature that is used in the structured SVM to restrict the generation of constraints. Since every document needs to be represented as a sequence of sentences, in SVM-HMM, the qid's are used to obtain the document structure. The order of the HMM was set to 2, which means that the two previous sentences were considered for collective classification.

5.4 Evaluation

We evaluate the performance of kLog against a baseline system and a memory-based tagger (Daelemans and van den Bosch, 2009). The results are also compared against those from Kim et al. (2011), which is the original benchmark system for this task.

5.4.1 Datasets

We perform our experiments on the NICTA-PIBOSO dataset from Kim et al. (2011) (kindly provided by the authors). It contains 1,000 abstracts of which 500 were retrieved from MEDLINE by querying for diverse aspects in the domains of traumatic brain injury and spinal cord injury. The dataset consists of two types of abstracts. If the abstract contains section headings (e.g. *Background*, *Methodology*, *Results*, etc.), it is considered to be *structured*. This information can be used as background knowledge in the model. The other abstracts are regarded as *unstructured*. To diversify the corpus, the remaining 500 abstracts were randomly sampled from a set of queries covering different medical issues.

The definitions of the semantic tags used as annotation categories are a variation on the PICO tag set, with the addition of two additional categories (see Table 5.1 in Section 5.2). Each sentence can be annotated with multiple classes. This renders the task a multiclass multilabel classification problem. The statistics on this dataset can be found in Table 5.2.

In order to apply the same evaluation setting as Kim et al. (2011), we also tested our method on the external dataset of Demner-Fushman et al. (2006).

[3] http://www.cs.cornell.edu/people/tj/svm_light/svm_hmm.html

	All	S	U
Nb. Abstracts	1000	376	624
Nb. Sentences	10379	4774	5605
- Background	2557	669	1888
- Intervention	690	313	377
- Outcome	4523	2240	2283
- Population	812	369	443
- Study Design	233	149	84
- Other	1564	1034	530

Table 5.2: Number of abstracts and sentences for structured (S) and unstructured (U) abstract sets, including the number of sentences per class (taken from Kim et al., 2011).

It consists of 100 abstracts, of which 51 are structured. Because the semantic tag set used for annotation slightly differs from the one presented in Table 5.1, and to make our results comparable, we use the same mapping as used by Kim et al. (2011).

Preprocessing

The sentences were preprocessed with a named entity tagger and a dependency parser.

Named entity tagging was performed with the BiogaphTA named entity module, which matches token sequences with entries in the UMLS database.[4] UMLS integrates over 2 million names for some 900,000 concepts from more than 60 families of biomedical vocabularies (Bodenreider, 2004). The tagger matches sequences with a length of a maximum of 4 tokens. This covers 66.2% of the UMLS entries. With UMLS, different token sequences referring to the same concept can be mapped to the same concept identifier (CID). The BiographTA named entity tagger was evaluated on the BioInfer corpus (Pyysalo et al., 2007) obtaining a 72.02% F1 score.

Dependency parsing was performed with the GENIA dependency parser GDep (Sagae and Tsujii, 2007), which uses a best-first probabilistic shift-reduce algorithm based on the LR algorithm (Knuth, 1965) extended by the pseudo-projective parsing technique. This parser is a version of the KSDep dependency parser trained on the GENIA Treebank for parsing biomedical text. KSDep was evaluated in the CoNLL 2007-Shared Task[5] obtaining a Labeled Attachment

[4]From UMLS, only the MRCONSO.RRF and MRSTY.RRF files are used.

[5]More information can be found on the website of the CoNLL 2007-Shared Task `http://nextens.uvt.nl/depparse-wiki/SharedTaskWebsite`.

Score of 89.01% for the English dataset. GDEP outputs the lemmas, chunks, Genia named entities and dependency relations of the tokens in a sentence.

5.4.2 Baseline and Benchmarks

We compare the kLog system to three other systems: a baseline system, a memory-based system, and the scores reported by Kim et al. (2011).

The memory-based system that we use is based on the memory-based tagger MBT[6] (Daelemans and van den Bosch, 2009). This machine learner is originally designed for part-of-speech tagging. It processes data on a sentence basis by carrying out sequential tagging, *viz.* the class label or other features from previously tagged tokens can be used when classifying a new token. In our setup, the sentences of an abstract are taken as the processing unit and the collection of all sentences in an abstract is taken as one sequence.

The features that are used to label a sentence are the class labels of the four previous sentences, the ambitags[7] of the following two sentences, the lemma of the dependency root of the sentence, the position of the sentence in the abstract, the lemma of the root of the previous sentence, and section information. For each root lemma, all possible class labels, as observed in the training data, are concatenated into one ambitag. These tags are stored in a list. An ambitag for a sentence is retrieved by looking up the root lemma in this list. The position of the sentence is expressed by a number. Section information is obtained by looking for a previous sentence that consists of only one token in uppercase. Finally, basic lemmatization is carried out by removing the trailing *S* from the identified section headers. All other settings of MBT are the default settings, and to prevent overfitting, no feature optimization nor feature selection was carried out.

When a class label contains multiple labels, for example *population* and *study design*, these labels are concatenated in an alphabetically sorted manner. This method reduces the multilabel problem to a problem with many different labels, i.e., the label powerset method of Tsoumakas et al. (2010).

The baseline system is exactly the same as the memory-based system except that no machine learner is included. The most frequent class label in the training data, i.e., *Outcome*, is assigned to each instance. The memory-based system enables us to compare kLog against a basic machine learning approach, using few features. The majority baseline system enables us to compare the

[6] `http://ilk.uvt.nl/mbt` [16 March 2012]

[7] An ambitag is a symbolic label defining for a word which different tags it can have according to the corpus (Daelemans et al., 2010).

memory-based system and kLog against a baseline in which no information about the observations is used.

Note that, although the systems participating in the ALTA 2012-Shared Task were also evaluated on the NICTA-PIBOSO corpus, a direct comparison is not possible, since in the ALTA evaluation a fixed train-test split was used, whereas we performed cross-validation on the full dataset. Furthermore, for the shared task, the abstracts were not separated based on their structure but were left interspersed in both the training and test set, and AUC was chosen as the main evaluation metric.

5.4.3 Parametrization

The kLog hyperparameters, namely the distance d and radius r, influence the amount of sentence context that is taken into account during learning. This requires a deliberate choice during parametrization. From a linguistic perspective, the use of unigrams is justifiable, since most phrases that contain clues on the structure of the abstract (e.g. *evaluation*, *methodolody*) can be expressed with single words. This is reflected by a distance and radius both set to 1. This also enables us to capture the relational information attached to the word in focus, i.e., the current kernel point. This is confirmed by cross-validation for the hyperparameters. Besides the word entities `w`, also the `sentence` entities and the respective intensional relations for both structured and unstructured abstracts are selected as kernel points. The additional contextual information at the sentence level is thus taken into account as well, when generating the extended high-dimensional feature vectors for the interpretations by means of the graph kernel.

At this point, only the sequence information at word level is taken into account by kLog. Since we use a sequence labeling approach as statistical learner, i.e., SVM-HMM, at the level of the abstract this information is however implicitly taken into account during learning. For SVM-HMM, the cost parameter C, which regulates the trade-off between the slack and the magnitude of the weight-vector, and ϵ, that specifies the precision to which constraints are required to be satisfied by the solution, were optimized by means of cross-validation. For the other parameters, the default values[8] were used.

5.4.4 Results

Experiments are run on structured and unstructured abstracts separately. On the NICTA-PIBOSO corpus, we performed 10-fold cross-validation. Over all folds, all labels, i.e., the parts of the multilabels, are compared in a binary

[8] As indicated in Section 5.3.3, we used the implementation of Thorsten Joachims.

CV/6-way	MBT		Kim et al.		kLog	
Label	**S**	**U**	**S**	**U**	**S**	**U**
Background	71.00	61.30	81.84	68.46	86.19	76.90
Intervention	24.30	6.40	20.25	12.68	26.05	16.14
Outcome	87.90	70.40	92.32	72.94	92.99	77.69
Population	50.60	15.90	56.25	39.80	35.62	21.58
Study Design	45.90	13.10	43.95	4.40	45.50	6.67
Other	86.10	20.90	69.98	24.28	87.98	24.42

Table 5.3: F1-scores per class for structured (S) and unstructured (U) abstracts.

	Baseline		MBT		kLog	
Method	**S**	**U**	**S**	**U**	**S**	**U**
CV/6-way	43.90	41.87	80.56	57.47	84.29	67.14
CV/5-way	61.79	46.66	86.96	64.37	87.67	72.95
Ext/5-way	66.18	6.76	36.34	11.56	20.50	14.00
Ext/4-way	30.11	27.23	67.29	55.96	50.40	50.50

Table 5.4: Micro-averaged F1-scores obtained for structured (S) and unstructured (U) abstracts, both for 10-fold cross-validation (CV) and on the external corpus (Ext).

way between gold standard and prediction. Summing all true positives, false positives, true negatives, and false negatives over all folds leads to micro-averaged F-scores. This was done for two different settings. In one setting, *CV/6-way*, we combined the labeling of the sentences with the identification of irrelevant information, by adding the *Other* label as an extra class in the classification. The results are listed in Table 5.3.

For this setting, kLog is able to outperform both MBT and the system of Kim et al. (2011), for both structured and unstructured abstracts on all classes except *Population*. From Table 5.4, where the micro-averaged F1-scores over all classes and for all settings are listed[9], it can be observed that kLog performs up to 3.73% better than MBT over structured abstracts, and 9.67% better over unstructured ones.

Although to a lesser extent for the structured abstracts, the same pattern can be observed for the *CV/5-way* setting, where we tried to classify the relevant

[9]Note that the results for Kim et al. (2011) are not listed in this table, as we did not have the individual contingency tables available to calculate the micro-averaged F1-scores for this setting.

CV/5-way	MBT		Kim et al.		kLog	
Label	**S**	**U**	**S**	**U**	**S**	**U**
Background	87.10	64.90	87.92	70.67	91.45	80.06
Intervention	48.00	6.90	48.08	21.39	45.58	22.65
Outcome	95.80	75.90	96.03	80.51	96.21	83.04
Population	70.90	21.40	63.88	43.15	63.96	23.32
Study Design	50.00	7.40	47.44	8.60	48.08	4.50

Table 5.5: F1-scores per class for 5-way classification over structured (S) and unstructured (U) abstracts.

	MBT		Kim et al.		kLog	
Label	**S**	**U**	**S**	**U**	**S**	**U**
			Ext/5-way			
Background	58.90	15.70	56.18	15.67	58.30	29.10
Intervention	21.50	13.80	15.38	28.57	40.00	34.30
Outcome	29.30	17.80	81.34	60.45	27.80	24.10
Population	10.70	17.80	35.62	28.07	5.60	28.60
Other	40.70	3.50	46.32	15.77	11.40	8.50
			Ext/4-way			
Background	90.40	67.50	77.27	37.50	65.00	68.60
Intervention	29.00	23.10	28.17	8.33	28.10	32.30
Outcome	74.10	74.60	90.50	78.77	72.40	72.70
Population	48.70	23.80	42.86	28.57	11.80	15.40

Table 5.6: F1-scores per class for 5-way and 4-way classification over structured (S) and unstructured (U) abstracts on the external corpus.

sentences only, without considering the irrelevant ones. The per-class results for this setting are shown in Table 5.5.

For the external corpus, the results are listed in Table 5.6. Although kLog performs comparably for the individual classes *Background* and *Intervention*, its overall performance is worse on the structured abstracts. In case of the unstructured abstracts, kLog performs better on the majority of the individual classes and in overall performance for the 5-way setting, and comparable for the 4-way setting.

As a general observation, it is important to note that there is a high variability between the different labels. Due to kLog's ability to take the structured input into account, we assume a correlation between the structure of sentences with a particular label and the prediction quality. An extensive error analysis could allow one to detect patterns which may give rise to additional declarative

background knowledge to incorporate into the model.

The follow-up approach by Hassanzadeh et al. (2014) reports improved results on the NICTA-PIBOSA dataset. However, this requires a two-step approach, in which an increased feature space is used and the rhetorical structure first needs to be preprocessed and encoded propositionally, in contrast to our dynamic declarative approach. To the best of our knowledge, no improvements on the dataset by Demner-Fushman et al. (2006) have been reported so far.

5.5 Conclusions

In this chapter, we presented a graph kernel–based relational learning approach for the automatic identification of PICO categories in medical abstracts. To this end, the sentence-level approach from the previous chapter was extended to the document level. As there are more than two class labels and each sentence can be labeled with more than one label, also the learning setting was upgraded from a binary to a multiclass-multilabel classification task.

Besides the relational representation, also two other properties of kLog proved particularly helpful. Declarative feature construction via intensional signatures enabled to exploit additional structural regularities and improve the generalization power of the learner. Furthermore, in contrast to other SRL systems, the relational high-dimensional feature space generated by kLog can be used as input for any statistical learner. Since this task can be cast as a sequence labeling problem, an SVM-HMM was used which enabled to take into account the sequence relation between the sentences using their relational representation as observations. On the majority of the class labels, kLog is able to perform comparably to or better than a memory-based tagger and the original benchmark system on this task by Kim et al. (2011). This can be attributed to the combination of kLog's aforementioned distinguishing characteristics, namely the ability to include additional document context via declarative feature construction, and the use of the relational representation as input to a specialized statistical learner, which made it possible to exploit the sequence relation between the sentences.

Chapter 6

kLogNLP

In the previous chapters we have shown that the graph kernel–based relational learning approach of kLog has already been successful in solving a number of natural language processing tasks. The success of the approach can be attributed to kLog's distinguishing characteristics. First, it is able to transform relational representations into graph-based ones, which allows one to incorporate structural features into the learning process. Subsequently, kernel methods are used to work in an extended high-dimensional feature space, which is much richer than most of the direct propositionalisation approaches. Second, it uses the logic programming language Prolog for defining (additional) background knowledge, which renders the model very interpretable and offers a flexible experimentation environment. Finally, in contrast to other SRL systems, the relational high-dimensional feature space generated by kLog can be used as input for any statistical learner.

These properties prove especially advantageous in the case of NLP. The graphical approach of kLog is able to exploit the full relational representation that is often a natural way to express language structures, and in this way enables the learner to fully exploit contextual features. On top of this relational learning approach, the declarative feature specification allows one to include additional background knowledge, which is often essential for solving NLP problems.

In this chapter, we will present *kLogNLP*, a natural language processing module for kLog. This module enriches kLog with NLP-specific preprocessors, enabling the use of existing libraries and toolkits within an elegant and powerful declarative machine learning framework. Starting from a dataset and a declaratively specified E/R model of the domain, it transforms the dataset into a graph-based relational format. We propose a general model that fits most

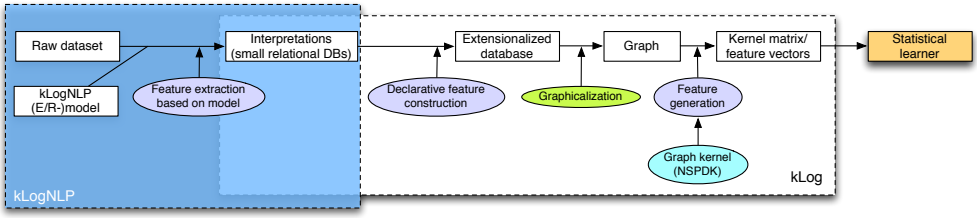


Figure 6.1: General kLog workflow extended with the kLogNLP module

NLP tasks, which can be extended by specifying additional relational features in a declarative way. The resulting relational representation then serves as input for kLog, and thus results in a full relational learning pipeline for NLP.

kLogNLP is most related to Learning-Based Java (LBJ) (Rizzolo and Roth, 2010) in that it offers a declarative pipeline for modeling and learning NLP tasks. The aims are similar, namely abstracting away the technical details from the programmer, and leaving him to reason about the modeling. LBJ focuses on the learning side, by the specification of constraints on features which are reconciled at inference time, using the constrained conditional model framework. Due to its embedding in kLog, kLogNLP focuses more on the relational modeling, in addition to declarative feature construction and feature generation using graph kernels. kLog in itself is related to several other frameworks for relational learning, of which a detailed discussion can be found in (Frasconi et al., 2014).

This chapter is structured according to the kLogNLP workflow depicted in Figure 6.1, which is an extension of the general kLog workflow as discussed in Section 2.2.3. We will discuss the two parts that differ from the original workflow, namely the data modeling in Section 6.1, and the extensional feature extraction in Section 6.2. Section 6.3 concludes this chapter.

This chapter is based on

> Mathias Verbeke, Paolo Frasconi, Kurt De Grave, Fabrizio Costa, and Luc De Raedt. kLogNLP: Graph kernel-based relational learning of natural language. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Annual Meeting of the Association for Computational Lingusitics (ACL), Baltimore, Maryland, USA, 22-27 June 2014*, pages 85–90. Association for Computational Linguistics, 2014.

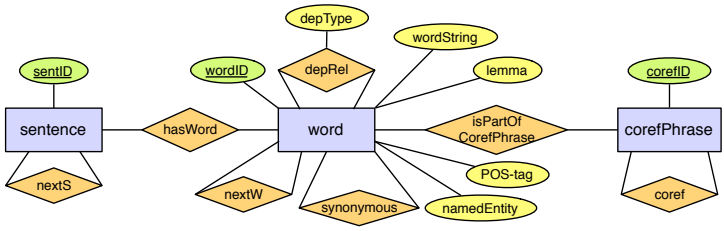


Figure 6.2: Entity-relationship diagram of the kLogNLP model

6.1 Data Modeling: the kLogNLP Model

As kLogNLP is an extension of kLog, it learns from interpretations. In the NLP setting, an interpretation most commonly corresponds to a sentence or a document. The scope of an interpretation is either determined by the task (e.g., for document classification, each interpretation will at least need to comprise a single document), or by the amount of context that is taken into account (e.g., in case the task is sentence classification, the interpretation can either be a single sentence or a full document, depending on the scope of the context).

Since in NLP, most tasks are situated at either the document, sentence, or token level, we propose the E/R model in Figure 6.2 as a general domain model suitable for most settings. It is a generalization of the models used for the hedge cue detection task (Chapter 4) and the evidence-based medicine identification problem (Chapter 5), and is able to represent interpretations of documents as a sequence (`nextS`) of `sentence` entities, which are composed of a sequence (`nextW`) of `word` entities. Besides the sequence relations, the dependency relations between words (`depRel`) are taken into account, where each relation has its type (`depType`) as a property. Furthermore, possible synonymy relations (`synonymous`) are taken into account. Since both single words and phrases can be coreferent with one another, the `corefPhrase` entity is introduced to model coreference relations. It represents a word or phrase that participates in a coreference relation (`coref`) with another word or phrase.

The entities in our model also have a primary key, namely `wordID`, `sentID`, and `corefID` for words, sentences and phrases participating in a coreference relation respectively. Additional properties can be attached to words such as the `wordString`, its `lemma`, `POS-tag`, and `namedEntity` type.

The E/R model of Figure 6.2 is coded declaratively in kLog as shown in Listing 6.1. As discussed in Section 2.2.3, each signature is of a certain type; either `extensional` or `intensional`. kLogNLP only acts at the extensional level.

```
begin_domain.

signature sentence(sent_id::self)::extensional.

signature nextS(sent_1::sentence, sent_2::sentence)::extensional.

signature word(word_id::self,
         word_string::property,
         lemma::property,
         postag::property,
         namedEntity::property
         )::extensional.

signature nextW(word_1::word, word_2::word)::extensional.

signature corefPhrase(coref_id::self)::extensional.

signature isPartOfCorefPhrase(coref_phrase::corefPhrase, word::
    word)::extensional.

signature coref(coref_phrase_1::corefPhrase, coref_phrase_2::
    corefPhrase)::extensional.

signature synonymous(word_1::word, word_2::word)::extensional.

signature dependency(word_1::word,
          word_2::word,
          dep_rel::property
         )::extensional.

kernel_points([word]).

end_domain.
```

Listing 6.1: Declarative representation of the kLogNLP model

6.2 Extensional Feature Extraction

As discussed before, kLog assumes a closed world, which means that atoms that are not known to be true, are assumed to be false. For extensional signatures, this entails that all ground atoms need to be listed explicitly in the relational database of interpretations. These atoms are generated automatically by the kLogNLP module based on the kLog script and the input dataset. Considering the defined attributes and relations in the model presented in Listing 6.1, the module interfaces with NLP toolkits to preprocess the data to the relational format of kLog. The user can remove unnecessary extensional signatures or modify the number of attributes given in the standard kLogNLP script as given in Listing 6.1 according to the needs of the task under consideration.

An important choice is the inclusion of the `sentence` signature. By inclusion, the granularity of the interpretation is set to the document level, which implies that more context can be taken into account. By excluding this signature, the granularity of the interpretation is set to the sentence level. Also note that the model follows the basic principles of entity-relationship modeling; the removal of an entity that appears as an argument type in other relations, also causes these relations to be removed (e.g., removal of the `sentence` signature also invalidates the `nextS` signature, as it has arguments of type `sentence`).

Currently, kLogNLP interfaces with the following NLP toolkits:

NLTK The Python Natural Language Toolkit (NLTK)[1] (Bird et al., 2009) offers a suite of text processing libraries for tokenization, stemming, tagging and parsing, and offers an interface to WordNet.

Stanford CoreNLP Stanford CoreNLP[2] provides part-of-speech tagging, named entity recognition, parsing and coreference resolution functionality.

Listing 6.2 illustrates the declarative specification of a learning task in kLogNLP, in this case a classification task with 5-fold cross-validation. The preprocessing toolkit to be used can be set using the kLogNLP flags mechanism, as illustrated by line 3. Subsequently, the `dataset` predicate (illustrated in line 4) calls kLogNLP to preprocess a given dataset.[3] This is done according to the specified kLogNLP model, i.e., the necessary preprocessing modules to be called in the preprocessing toolkit are determined based on the presence of the entities, relationships, and their attributes in the kLogNLP script. For example, the presence of `namedEntity` as a property of `word` results in the addition of a named entity recognizer during preprocessing. The resulting set of interpretations is output to a given file. In case several instantiations of a preprocessing module are available in the toolkit, the preferred one can be chosen by setting the name of the property accordingly. The names as given in Listing 6.1 outline the standard settings for each module. For instance, in case the Snowball stemmer is preferred above the standard (Wordnet) lemmatizer in NLTK, it can be selected by changing `lemma` into `snowball` as name for the word lemma property (line 9 of Listing 6.1).

Optionally, additional extensional signatures can easily be added to the knowledge base by the user, as deemed suitable for the task under consideration. At each level of granularity (document, sentence, or word level), the user is given the corresponding interpretation and entity IDs, with which additional

[1] `http://www.nltk.org/`

[2] `http://nlp.stanford.edu/software/corenlp.shtml`

[3] Currently supported dataset formats are directories consisting of (one or more) plain text files or XML files consisting of sentence and/or document elements.

```
1  experiment :-
2      % kLogNLP
3      klognlp_flag(preprocessor,stanfordnlp),
4      dataset('/home/dataset/train/','trainingset.pl'),
5      attach('trainingset.pl'),
6      % Kernel parametrization
7      new_feature_generator(my_fg,nspdk),
8      klog_flag(my_fg,radius,1),
9      klog_flag(my_fg,distance,1),
10     klog_flag(my_fg,match_type,hard),
11     % Learner parametrization
12     new_model(my_model,libsvm_c_svc),
13     klog_flag(my_model,c,0.1),
14     kfold(target,5,my_model,my_fg).
```

Listing 6.2: Predicate for classification with 5-fold cross-validation.

extensional facts can be added using the dedicated Python classes. As may be clear, the kLog script presented in Listing 6.1 can also be extended using declarative feature construction with *intensional* signatures.

For declaratively specified features, there is no need to reprocess the dataset each time a new feature is introduced, which renders experimentation very flexible. Note that changes in the extensional signatures do require reprocessing the dataset. However, for different runs of an experiment with varying parameters for the feature generator or the learner, kLogNLP uses a caching mechanism to check if the extensional signatures have changed, when calling the `dataset` predicate.

The feature generator is initialized using the `new_feature_generator` predicate. Its hyperparameters (e.g., maximum distance and radius, and match type) can be set using the kLog flags mechanism (Listing 6.2, lines 6-10). In the last step, different learning tasks can be performed on the resulting extended feature space. As indicated before, to this end, kLog interfaces with several solvers, including LibSVM and SVM SGD. Lines 11-14 (Listing 6.2) illustrate the initialization of LibSVM and its use for 10-fold cross-validation.

6.3 Conclusions

In this chapter, we presented kLogNLP, a natural language processing module for kLog. It enriches kLog with NLP-specific preprocessors, resulting in a full relational learning pipeline for NLP tasks, embedded in an elegant and powerful declarative machine learning framework. The presented model is a generalization of the models from the previous chapters and is fit to be used

as a basis for most NLP tasks. The model can be tailored to meet the needs of the task at hand, and is subsequently used to dynamically transform the dataset into the graph-based relational format of kLog by interfacing with different existing libraries and toolkits. The resulting relational representation serves as input for the rest of the kLog workflow and enables one to exploit the advantages of its distinguishing characteristics as outlined in the previous chapters. The kLogNLP module is available under an open-source license at http://dtai.cs.kuleuven.be/klognlp.

Chapter 7

Relational Regularization and Feature Ranking

In the preceding chapters we have investigated and illustrated graph kernel–based relational learning on different natural language processing tasks. The main focus was on the modeling of the domain and the inclusion of the appropriate background knowledge using logic programming to solve the learning task. The goal of this chapter is to gain deeper insights in the contribution of the individual features (i.e., kLog's extensional and intensional signatures). To this end, we will lift regularization to a *relational* level, in order to achieve a higher interpretability and increase performance. Furthermore, we will present two relational feature selection approaches. Although motivated from an NLP perspective, note that the presented methods can also be used for other domains; this will be illustrated by a problem in chemoinformatics.

As introduced in Section 2.1.1, a regularization procedure is a way to introduce information, independently from the evidence present in the data, in order to solve an ill-posed problem or to prevent overfitting. Typically one tries to enforce simplicity constraints, for example reducing the effective degrees of freedom in a model by tying together the model parameters in some prescribed way. We introduce a novel type of regularization, called *relational regularization*, that takes into account the relational structure of the domain as specified in an E/R model. These relations are used to tie the parameters of a predictive linear model.

Based on this concept, we present an embedded and a wrapper approach for *relational feature selection and ranking*. This will allow us to get deeper insights

into the relative importance of the elements in the E/R model. The proposed methods are able to identify the most relevant sets of predicates, which yields more readily interpretable results than selecting low-level propositionalized features.

The proposed techniques are implemented in kLog. However, the techniques presented can also be adapted to other relational learning frameworks.

This chapter is organized as follows. First, we will discuss related work in Section 7.1, before illustrating the problem setting by means of an artificial example in Section 7.2. Section 7.3 discusses relational regularization, before turning to relational feature selection and ranking in Section 7.4. In Section 7.5, the approach is evaluated on two classification tasks in natural language processing and bioinformatics, before concluding in Section 7.6.

This chapter is based on

> Fabrizio Costa, Mathias Verbeke, and Luc De Raedt. Relational regularization and feature ranking. In *Proceedings of the 14th SIAM International Conference on Data Mining, SIAM International Conference on Data Mining, Philadephia, Pennsylvania, USA, 24-26 April 2014*, pages 650–658. SIAM, 2014.

7.1 Related Work

Feature selection has a long tradition in machine learning and has also been studied for logical and relational learning. For instance, Alphonse and Matwin (2002) upgrade propositional methods for feature selection in relational learning using a transformation to multi-instance learning rather than to graphs. However, they neither employ a notion of locality, to link the importance of features that are close in a relational representation of the domain, nor a notion of regularization, to use this information to enforce simplicity constraints on the model. Jensen and Neville (2002) analyze the feature selection bias caused by linkage and autocorrelation, two notions capturing important properties about the topology of relational data. They also use this to leverage the performance of relational learners (Neville and Jensen, 2005), but do not provide a regularization approach.

Regularization has been used for learning the structure of Markov Logic networks, e.g., Huynh and Mooney (2008) first derive a large set of clauses and then determine their weights using L_1 regularization. This set of clauses is fixed in the second step, and they apply regularization techniques which do not take into account the relational topology. Quanz and Huan (2009) extend

the L_2 regularized logistic regression to aligned graph classification, however a propositional (vector) representation is used. Furthermore, all instances have the same graph structure, where the relations are assumed known and fixed. In our case, the graphs associated to examples are all different, and generated declaratively. Zhou and Schölkopf (2004) propose a general regularization framework for graphs. However, they do not work within a logical setting and they deal with single graph domains. In this work, rather than considering graphs representing relations *between* instances, we use a graph representation to encode the instances themselves, consisting of entities and relations, defined in a logical framework. Finally, the proposed method for relational regularization is also related to structured regularization. Usual sparsity methods, such as lasso (Tibshirani, 1996), constrain the cardinality of the support of the model that is learned by selecting a number of variables. Structured sparsity not only constrains the cardinality of the support, but also constrains the structure of the sparsity pattern. Group lasso (Yuan and Lin, 2006) considers a partition of the variables in a certain number of disjoint groups and regularizes the parameters according to these groups. Several variants of group lasso were presented, e.g., for dealing with overlapping groups (Jenatton et al., 2011). Graph lasso (Jacob et al., 2009) deals with variables that lie on a graph. As connected variables are more likely to be simultaneously relevant, graph lasso selects variables according to this graph structure. Whereas these methods require the groups to be fixed in advance, our approach determines these groups based on the E/R-model of the domain. Furthermore, as we work in a logical setting, our method also enables one to take declarative features features into account.

7.2 Overview

We will first illustrate the concepts and techniques used in this chapter by means of the artificial game introduced in Section 2.2.3. To recapitulate, the game consists of drawing a random sequence of 5 objects from a bag filled with cubes and spheres. Each object has a certain color and weight. The game can only be won if two spheres with the same color are drawn without drawing two cubes with the same color. In any other case the game is lost. Recall that besides `sphere` and `cube` entities, also some relations were defined. The extensional relations `next_s`, `next_c` and `next_sc` indicate the sequence relations in which objects are drawn from the bag. The relation `sph_eq_col` indicates that two spheres have the same color, whereas `diff_shape_eq_weight` indicates the relation between a sphere and a cube with equal weight. We also defined one intentional relation, namely `cub_eq_col`, between two cubes with the same color.

Relational Regularization and Feature Selection

Our proposal is based on two core ideas. The first is to define features on the E/R diagram, called *high-level (relational) features*, and features on the grounded version of the E/R model of each example, called *low-level (graph) features*. The second idea is to tie together sets of low-level features into high-level features.

From a logical and relational perspective, the features correspond to queries. The *low-level features* are derived from the *unrolled* E/R diagram and correspond to (particular) subgraphs of a graphicalized instance.[1] Some possible low-level features were already illustrated in Figure 2.7, e.g., one possible feature corresponds to the (ground) query `sphere(s1,r,2)` $\wedge$ `cube(c1,g,1)` (with `s1` and `c1` the identifiers of the entities).

The *high-level features* that we employ are derived from the E/R diagram. These can be obtained by replacing the values for the attributes in the low-level features by *variables* in the query (represented by a capital letter or a capitalized string, e.g., `Col`). At present we consider single vertices or paths of length l starting from nodes representing entities. For instance, selecting the vertex for `sphere` would result in the (non-ground) query `sphere(S,Col,W)`. This corresponds to all sphere entities in the unrolled E/R diagrams of the example instances. The path of length 3 `sphere - next_sc - cube` corresponds to the query `sphere(S,Col1,W1)` $\wedge$ `next_sc(S,C)` $\wedge$ `cube(C,Col2,W2)`. This selects all `sphere` and `cube` entities that are linked by a `next_sc` relation in the unrolled E/R diagrams of the instances. The difference between the low and high-level features is thus that the high-level features do not contain any constants and are represented by a subgraph of the E/R diagram, whereas the low-level ones contain values for all the involved attributes, and are represented by subgraphs of the *unrolled* E/R diagram. This type of query thus makes abstraction of the specific constants in the database and acts as a kind of template. It can have multiple occurrences in a database (that correspond to different possible answers to the query).

The definition of the high-level features (through the E/R diagram) is again reminiscent of inductive logic programming systems and constitutes the declarative bias of the system. In kLog terminology, the high-level features correspond to sets of intensional or extensional signatures that are connected in the E/R diagram of the domain. The reader should keep in mind that alternative biases can in principle be incorporated as well.

Notice that one can look for occurrences of a high-level feature in the database.

[1] More specifically, as our approach is embedded in kLog, which is based on the NSPDK kernel, we shall consider all pairs of near neighborhood subgraphs.

This can be realized either at the relational level (using, e.g., Prolog's query answering mechanism) or by graph matching (subgraph isomorphism testing between the feature and the graphicalized database).

In order to tie low-level features to high-level ones, we consider where the high-level feature occurs in the graphicalized examples. All the low-level graph features that are in the neighborhood of such an occurrence are collected in what we call the *coverage* of the high-level feature (see Figure 7.1). The underlying assumption is that predicates that are near in the graphicalized representation (in terms of shortest path distance), are likely to be more related in some meaningful way. Both the *relational regularization* and the *relational feature selection* techniques are based on this notion: the *regularization* is achieved tying the importance score of all low-level features that belong to the same coverage; while *feature selection* is obtained by ranking the high-level features according to the aggregated score of all low-level features that belong to the respective coverage. These notions are formally elaborated in Section 7.3 and 7.4.

Experimental Results

To illustrate the regularization setting in advance, we already provide the results on the artificial example. As candidate high-level relational features we consider the single vertices in the E/R diagram together with paths of the type entity-relation-entity. The resulting *relational feature ranking* is listed in Table 7.1. As expected, the `sph_eq_col` relation between two `spheres` is ranked highest, followed by `spheres` and `cubes`. The relations between two cubes are ranked lowest (i.e., they are highly discriminative for the negative examples). `diff_shape_eq_weight` relations between two distinct objects can be seen as neutral high-level features.

#	Relational features	Score
1	sphere - sph_eq_col - sphere	22.90
2	sphere	2.11
3	cube	-1.48
4	sphere - diff_shape_eq_weight - cube	-3.62
4	cube - diff_shape_eq_weight - sphere	-3.62
5	cube - cub_eq_col - cube	-11.78

Table 7.1: High-level feature ranking (Artificial example)

We also compared the performance of relational regularization to a standard regularization technique on a propositionalized representation. The latter is

obtained by considering a bag-of-words representation of the entity and relation nodes. The results are listed in Table 7.2. Both in terms of F-measure and area under the ROC curve (AUC), relational regularization shows a significant improvement over the non-relational case.

	Non-relational	Relational
Precision	79.63	80.65
Recall	84.31	98.04
F1	81.91	88.50
AUC	84.72	90.91

Table 7.2: Relational vs. non-relational regularization (Artificial example)

Explicit Features

In order to obtain the low-level features for the relational representation, kLog's graph kernel decomposition as performed by NSPDK will be used. However, in contrast to the standard kLog setup as presented in Section 2.2.3, where a kernelized learning machine based on NSPDK would only need the efficient computation of $\kappa_{r,d}$, here we want to explicitly extract the (low-level) feature encoding induced by the graph kernel. This gives us the flexibility to manipulate and combine the importance score associated with each feature, which is not achievable when working in the implicit form.

The technique, based on ideas presented by Costa and De Grave (2010), works in two steps: 1) a graph invariant encoding $\mathcal{L}^g(A)$ is constructed as a sorted list of annotated edges for a neighborhood graph, where the annotation uses an efficient quasi-canonical vertex relabeling; 2) the list is then hashed $H(\mathcal{L}^g(A)) \to \mathbb{N}$ to obtain an integer that is used to compose the final feature identifier. Whereas Costa and De Grave (2010) only used this technique to speed up the kernel computation, here we propose to expose the pseudo-identifiers and use them as feature indicators.

7.3 Relational Regularization

We will first introduce the technique used to link the explicit low-level feature representation in the graph to the high-level relational feature representation in the E/R diagram. After that we will introduce the relational regularization scheme used to enforce the dependency of the importance scores between all low-level graph features related to the same high-level relational feature.

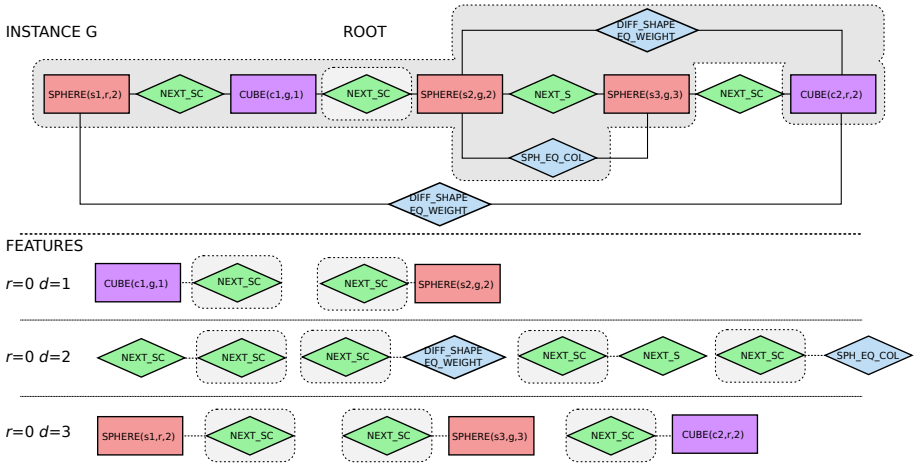


Figure 7.1: Feature co-occurrence. Top: neighborhood with $d_{max} = 3$ for root node of type NEXT_SC. Bottom: co-occurring features with $r_{max} = 0$: (r=0, d=1), (r=0, d=2) and (r=0, d=3).

Definitions: Topological Co-occurence, Co-occurrence Matrix, Coverage, and High-level Relational Features

We say that two low-level graph features are *topologically co-occurring* when they are both occurring in the same neighborhood subgraph *and* share at least one root vertex (see Figure 7.1). Formally, for a given example instance R and its graphicalization G, two features $(A, B), (A', B') \in R^{-1}(G)$ co-occur in a local neighborhood if they are of the form

$$(A, B) = (N_r^v(G), N_r^u(G) : d(v, u) = d) \tag{7.1}$$

and

$$(A', B') = (N_{r'}^v(G), N_{r'}^{u'}(G) : d(v, u') = d') \tag{7.2}$$

where N represents a neighborhood subgraph as introduced in Section 2.2.3. Note that the features need not have the same radius r or the same distance d between roots. Recall from Section 2.2.3 that to increase the efficiency and generalization power, the zero-extension of the NSPDK kernel is considered, which is obtained by imposing upper bounds on the radius and distance parameter. Intuitively, the maximum distance d_{max} controls the degree of locality, while the maximum radius r_{max} controls the complexity of the low-level graph features.

Given a dataset of instances $\hat{R}$ and its correspondent set of graphicalized instances $\hat{G}$, we define the *co-occurence matrix* as the matrix W which stores in

W_{ij} the aggregate number of times that feature i was topologically co-occurring with feature j in all the graphs in $\hat{G}$. We indicate with $\Phi^v_{r_{max},d_{max}}(G)$ the multiset of all graph features (with radius bounded by r_{max} and distance bounded by d_{max}) that are topologically co-occurring in the neighborhood of a root vertex v with radius d_{max} and have v as common root vertex.

Let $H \in \mathcal{H}$ be the (graph) representation of an E/R model; a *high-level relational feature* then corresponds to a (homomorphic) subgraph h of H. In the simplest case h can be a single vertex in H. We now define a mapping $\Pi : \mathcal{H} \times \mathcal{R} \mapsto \mathcal{G}$ which, for a given example instance $R \in \mathcal{R}$ (for which there exist a graphicalization $g(R) = G \in \mathcal{G}$) maps a subgraph h of the E/R diagram H to the corresponding subgraphs[2] of G in the graphicalized domain, that is, we map high-level features to low-level subgraphs. Let us now consider the set of all vertices in all graphs that are the result of the mapping of a given high-level feature. We are interested in the low-level features that can be generated starting from these vertices: we call this multiset the *coverage* of the high-level feature. Formally, given a subgraph $h \in H$ of an E/R diagram, we define the *coverage of* h as the multiset:

$$C(h) = \bigcup_{R \in \hat{R}} \bigcup_{v \in V(\Pi(h,R))} \Phi^v_{r_{max},d_{max}}(g(R)) \tag{7.3}$$

where R is taken over all instances in a dataset $\hat{R}$ and v is taken in all vertices that map to h according to the unfolding of the E/R element in each graphicalized instance $g(R) = G$. Using the coverage notion we can now link the importance of high-level relational features with the importance of low-level graph features. Given an importance score vector w associated to the low-level graph features, we define the importance score S of a subgraph h of the E/R diagram as the sum of the scores for the features in its coverage:

$$S(h) = \sum_{\phi_i \in C(I)} w_{\phi_i} \tag{7.4}$$

In the present implementation we do not consider sophisticated ways to construct the high-level feature domain, but instead consider only single vertices or paths of length 3, that is, triplets of the form entity-relation-entity. However, using a combinatorial decomposition approach, this can be extended to produce more complex high-level feature domains.

[2]Note that these subgraphs do not need to be connected.

Relational Regularization with Co-occurence Constraints

In the following we introduce the relational regularization scheme used to enforce dependency between all low-level graph features related to the same high-level relational feature.

The regularization technique is best introduced together with an associated learning algorithm. Here we employ models of the linear type, i.e.,

$$f(x) = w^\top x \tag{7.5}$$

where $x \in \mathbb{R}^p$ is a vector input in the high-dimensional space induced by the graph decomposition approach detailed in Section 2.2.3 and $w \in \mathbb{R}^p$ is the associated parameter vector. Given a dataset of input-output pairs $\{(x_i, y_i)\}_{i=1}^n$ describing a learning problem, and a choice for a loss function ℓ, we look for a parameter assignment that minimizes the empirical risk

$$\mathbb{E}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i) \tag{7.6}$$

We solve the minimization problem with a stochastic gradient descent technique, which simplifies the ordinary gradient descent algorithm and estimates the gradient direction on a per instance basis with the following iterative update[3]:

$$w_{t+1} = w_t - \gamma \nabla_w \ell(f(x_i), y_i) \tag{7.7}$$

As loss function we consider the regularized SVM loss:

$$\ell(f(x_i), y_i) = max\{0, 1 - y_i w^\top x_i\} + \lambda w^\top w \tag{7.8}$$

We now introduce the *relational regularizer*. Given the co-occurence matrix W we build the Laplacian matrix

$$L = D - W \tag{7.9}$$

with diagonal matrix D having entries $D_{ii} = \sum_j W_{ij}$, and its normalized version

$$\hat{L} = D^{-1/2} L D^{-1/2} \tag{7.10}$$

Imposing a penalty on the size of $w^\top \hat{L} w$ is equivalent to penalizing the difference between parameter w_i and parameter w_j when W_{ij} is large. This can be seen considering that

$$w^\top L w = \frac{1}{2}\sum_{i,j}(w_i - w_j)^2 W_{ij} \tag{7.11}$$

[3]We employ a decreasing gain $\gamma \approx \frac{1}{t}$ to ensure fast convergence as suggested by Bottou (2010).

hence for a given large W_{ij}, minimizing $w^\top Lw$ implies enforcing $w_i - w_j \to 0$. To see why, we rewrite

$$\sum_{i,j}(w_i - w_j)^2 W_{ij} = \sum_i w_i^2 D_{ii} + \sum_j w_j^2 D_{jj} - 2\sum_{i,j} w_i w_j W_{ij} = 2w^\top Lw \quad (7.12)$$

Similar considerations hold for $\hat{L}$. In practice this implies that the regularization term $w^\top \hat{L}w$ enforces the binding of parameters associated to features that are topologically co-occurring.

Adding this novel regularization term to the loss function we obtain:

$$\ell(f(x_i), y_i) = max\{0, 1 - y_i w^\top x_i\} + \lambda w^\top w + \rho w^\top \hat{L}w \quad (7.13)$$

Taking the derivatives gives us the corresponding new update rule for the stochastic gradient descent algorithm:

$$\begin{array}{ll} w_{t+1} = w_t - \gamma_t(\lambda w + \rho \hat{L}w) & \text{if } y_t w^\top x_t > 1 \text{ and} \\ w_{t+1} = w_t - \gamma_t(-y_t x_t + \lambda w + \rho \hat{L}w) & \text{otherwise} \end{array} \quad (7.14)$$

We keep the regularization term $w^\top w$ in place since, at times, the relational manifold assumption can hold to a lesser degree. In this case it is important to be able to tradeoff between the two regularizers via λ and ρ.

Note that in principle, our notion of locality and regularization could be applied to any dataset that can be interpreted as an E/R diagram with corresponding features. For example, in the context of Markov Logic our notion of locality might be applied to the grounded Markov Logic network where the (ground) clauses would act as the features and the entities would be the ground atoms.

7.4 Relational Feature Selection

We have introduced a technique to tie low-level propositional features to the high-level relational feature representation in the E/R diagram. In addition we have introduced a relational regularization scheme to enforce dependency between all low-level graph features related to the same high-level relational feature. We now turn our attention to the relational feature selection step.

Given a set of explicitly available low-level features, a standard feature selection approach selects a subset of these to maintain or enhance the performance of an associated predictive model. In the proposed approach, instead we lift the feature selection process one step higher, to a relational level. This implies that our goal is not to select important propositional features, but rather high-level relational features that are discriminative for the learning task.

There are three main strategies for performing feature subset selection, namely filter, embedded, and wrapper techniques (Guyon and Elisseeff, 2003). *Wrappers* consider the learner as a black box, and use it to score subsets of features according to their predictive power. *Filters* select subsets of variables as a preprocessing step. In contrast to the wrapper methods, filters use a performance measure other than the error rate to score a particular subset of features. This thus happens independently of the downstream learner. *Embedded methods* perform variable selection as part of the model construction process during training. In these methods, learning is interleaved with the feature selection procedure, and they are usually specific to a given learning machine.

Here we present an embedded and a wrapper strategy, both employing a linear model equipped with the relational smoother introduced in Section 7.3. The intuitive difference between the two approaches is that while the (more computationally efficient) embedded method is forced to consider the full E/R model all at once, the wrapper method analyzes the importance of isolated subsets of interacting entities and relations, modifying the E/R model under examination.

7.4.1 Embedded Strategy

Embedded methods incorporate variable selection as part of the training phase and are tightly linked to the learning algorithm.

Our approach is rooted in the AROM (Approximation of the zeRO norm Minimization) method (Weston et al., 2003). It is an approximation for the following optimization problem: $\min_w ||w||_0^0$ subject to:

$$y_i(w^\top x_i) \geq 1 \tag{7.15}$$

Here the solution is a separating hyperplane with the minimal number of nonzero elements in the parameter vector w. Since this is a combinatorially hard problem (Amaldi and Kann, 1998), Weston et al. (2003) suggest to minimize

$$\min_w \sum_{j=1}^{p} \ln(\epsilon + |w_j|) \tag{7.16}$$

instead, subject to:

$$y_i(w^\top x_i) \geq 1 \tag{7.17}$$

for which they can prove the bound:

$$||w_l||_0^0 \leq ||w_0||_0^0 + O\left(\frac{1}{\ln \epsilon}\right) \tag{7.18}$$

Algorithm 1 Embedded Relational Feature Selection

Input: predicate set S, radius r, distance d
Output: importance for each element in set S
Note: "$\cdot$" is a component-wise multiplication
G = RelationalDBGraphicalization(S)
X = ExplicitFeatureExtraction(G, r, d) ▷ X is the set of all vector representations
$z \leftarrow (1, \ldots, 1)$ ▷ z is a vector of scaling factors
repeat
  for $x_i \in X$ **do**
    $\bar{x}_i \leftarrow x_i \cdot z$ ▷ rescale all instances
  end for
  let w^* be the solution of the SVM problem on $\bar{X}$ with relational regularization
  $z \leftarrow z \cdot w^*$ ▷ update scaling factors
until convergence
$\hat{z} \leftarrow$ binarize(z) ▷ set to 1 nonzero features
for $x_i \in X$ **do**
  $\hat{x}_i \leftarrow x_i \cdot \hat{z}$ ▷ consider only nonzero features for each x_i
end for
let $\hat{w}^*$ be the solution of the SVM problem on $\hat{X}$ with relational regularization
return ComputePredicateImportance($\hat{w}^*$)

where w_l is the minimizer of the proposed approximation and w_0 is the minimizer of the original problem. The idea is that, because of the form of the logarithm (i.e., it decreases quickly to zero compared to how fast it increases for large values), significantly increasing a specific w_j can be compensated by setting to zero another w_i rather than attempting some compromise between them, thus encouraging sparse solutions. A similar reasoning holds in the case of non (linearly) separable data.

Interestingly, Weston et al. (2003) propose to solve the optimization problem with a very simple algorithm (see Algorithm 1), where input vectors are iteratively scaled by the optimization solution w. Here, differently from Weston et al. (2003), we solve an SVM optimization problem with relational regularization, which imposes the additional co-occurence constraints (as explained in the previous section) on this optimization problem. Therefore the low-level features are regularized according to the patterns defined by the high-level relational features. Put differently, the rescaling of the input vectors is done based on the relational features, upgrading the approach of Weston et al. (2003). As suggested in the original algorithm, in the last step we binarize the scaling vector z to have entries in $\{0, 1\}$. Only at this point we solve the SVM problem on the selected subset of nonzero features. This is an important step, given that the iterative algorithm scales w as a function of the iteratively *rescaled* input x_i and, in order to be used on the original input, it would require a compensatory rescaling.

Algorithm 2 Wrapper Relational Feature Selection

Input: predicate set S, radius r, distance d, number of predicate sets ns, number of ranked predicate sets to retain k, ranking measure m
Output: importance for each element in set S
$predicateSets$ = GeneratePossiblePredicateSets(S)
$sampledPredicateSets$ = SamplePredicateSets($predicateSets$, ns)
for each $predicateSet_i$ **in** $sampledPredicateSets$ **do**
  G_i = RelationalDBGraphicalization($predicateSet_i$)
  X_i = ExplicitFeatureExtraction(G_i, r, d)
  store the solution of the SVM problem on Xk_i with relational regularization using ranking measure m in $results[predicateSet_i]$
end for
$topkPredicateSets$ = CalculateTopKPredicateSets($results$, k)
for each $topkPredicateSet_i$ **in** $topkPredicateSets$ **do**
  Xk_i = ExplicitFeatureExtraction(Gk_i, r, d) ▷ Gk_i is the graphicalization of $topkPredicateSet_i$ computed before
  let w^* be the solution of the SVM problem on Xk_i with relational regularization
  predicateImportance[$topkPredicateSet_i$] =ComputePredicateImportance(w^*)
end for
return ComputeOverallPredicateImportance($predicateImportance$)

7.4.2 Wrapper Strategy

Our wrapper method (see Algorithm 2) is a derivation of the Random Sets approach (Nikulin, 2008). The method receives as input the set of predicates to be ranked and the number of random subsets of predicates to try out. The idea is to induce a predictive model on each random set of predicates, retain only the top performing models, and select the high-level relational features that are ranked highest among these models.

When working in a propositional setting, there are generally no constraints among the features and one can sample sets of features uniformly at random. In a *relational* setting, however, given the logical dependencies between predicates, there are consistency issues. As an example, in the definition of our domain knowledge, we could have predicate A defined in terms of predicate B; in this case removing B would necessarily imply the removal of A. To deal with this issue we make use of a dependency graph, built automatically from the background knowledge, to guarantee the generation of consistent E/R models.

A predictive model equipped with the relational regularizer is learned on the graphs resulting from applying the graphicalization procedure on each reduced E/R model. A user-specified number of the best predictive linear models (according to a specified measure, e.g., F1-score) are selected. Subsequently, the importance score of the high-level features is computed for each of the selected models. These scores serve as input to calculate the overall importance scores for the high-level features.

7.5 Evaluation

We validated our techniques on two tasks with structural input, i.e., tasks where the instances are represented using relational information, for which relational learning, and kLog in particular, has proven to be a promising approach. The first task is *hedge cue detection*, as discussed in Chapter 4. The second, *molecular toxicity prediction*, is a classification task in bioinformatics.

7.5.1 Natural Language Processing: Hedge Cue Detection

As explained in Chapter 4, the instances for this task are sentences and consist of a number of words `w`, for which the order is represented by the `next` relation. The dependency relations represent the structure of syntactic relations between the words. This is modeled by the `depHead` relations, where the `depRel` property specifies the type of the dependency (e.g., subject, object, etc.). Other properties of the word that are taken into account as features are the word string itself, its lemma, the part-of-speech tag, the chunk tag and a binary feature that represents whether the word is part of a predefined list of speculative strings. `weaselSentence` represents the target relation.

As outlined in Section 4.3.2, we also used three additional predicates to encode additional background knowledge; `CW` retains only the words that appear in a predefined list of weasel words compiled from the training data, accompanied by their lemma and POS-tag. `LeftOf` and `RightOf` represent the two surrounding words of each `CW` word, again with their respective lemmas and POS-tags. Since these additional predicates have shown to improve performance on previous results for this task, it is to be expected that these are the main discriminative predicates, while the propositional lexico-syntactic features should be less informative.

In a first step, the high-level feature importance was calculated by both the embedded and the wrapper approach, for which the results are listed in Table 7.3. As can be seen, both approaches give a similar ranking, except for `RightOf` and `CW`, which are switched, and the basic lexico-syntactic features, which get an equal weight in the embedded method, and slightly different scores in the wrapper method.

Table 7.4 contains the ranking of the possible triplets of predicates of type entity - relation - entity. It can be observed that pairs of consecutive hedged words or hedged words that are linked in the dependency tree are very informative for the task at hand.

As for the artificial example, we also compared the performance of relational regularization to standard regularization on a propositionalized representation,

Embedded approach			Wrapper approach		
#	Feature	Score	#	Feature	Score
1	CW	27.20	1	RightOf	15.99
2	RightOf	6.69	2	CW	13.72
3	LeftOf	4.30	3	LeftOf	9.68
4	Next	4.02	4	Next	5.81
5	DH	3.42	4	DH	2.97
5	WordString	-2.35	5	WordString	-2.06
5	InList	-2.35	6	InList	-2.56
5	Chunk	-2.35	7	Chunk	-2.71
5	Lemma	-2.35	8	Lemma	-2.72
5	PoS	-2.35	9	PoS	-2.90

Table 7.3: High-level feature ranking (Hedge cue detection)

Triplet	Score
cw - next - cw	49.79
cw - dh - cw	49.73
cw - dh - word	37.29
cw - next - word	31.79
word - dh - word	12.62
word - next - word	-5.76

Table 7.4: Triplet ranking (Hedge cue detection)

obtained by considering a bag-of-words representation of the entity and relation nodes (referred to as non-relational regularization in Table 7.5). As a baseline, we compared to the performance of classification without regularization, again both in the relational and the propositional case (referred to as relational and non-relational baseline respectively). The results can be found in Table 7.5. In the case of hedge cue detection, it can be observed that relational regularization improves over the non-relational case. When comparing to the respective results when no regularization is used, we observe similar results[4]. This indicates that the right features are selected during the regularization process, as learning

[4]The difference in results for the relational baseline as compared to the numbers outlined in Chapter 4 is due to a smaller grid search parameter range during optimization of the SVM, which was used as a statistical learner at the end of the kLog workflow. This was done in order to have a fair comparison with the other methods, which required a smaller parameter grid due to greater runtimes. However, the same optimal kernel hyperparameters as obtained in Chapter 4 were used.

	Relational regularization	Non-relational regularization	Relational baseline	Non-relational baseline
Hedge Cue Detection				
PRECISION	59.98	33.19	60.61	33.19
RECALL	50.76	96.42	49.28	96.42
F1	54.99	49.38	54.36	49.38
Molecular Toxicity				
PRECISION	79.69	55.81	82.37	57.14
RECALL	74.13	54.21	70.02	44.35
F1	76.81	55.00	75.69	49.94

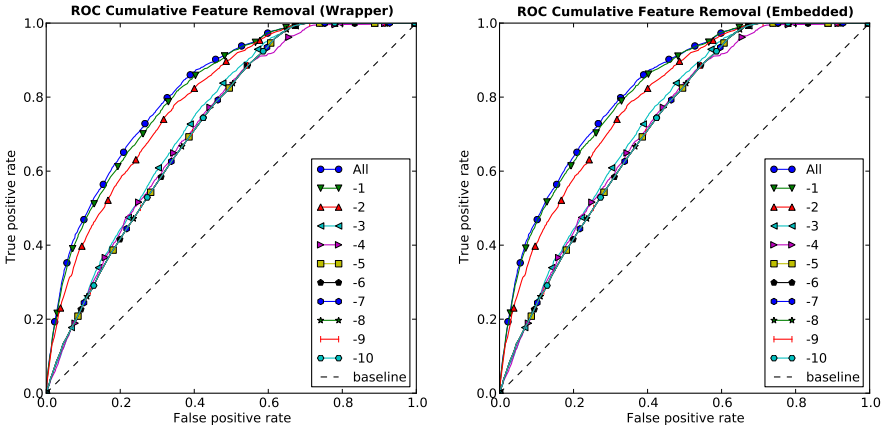
Table 7.5: Relational vs. non-relational regularization

with a reduced number of features does not come at the cost of a reduced performance.

Cumulative and Individual Feature Removal

In order to show the quality of the ranking, we also examined the ROC curves, where we cumulatively and individually remove the predicates (Figures 7.2 and 7.3). For the former, we removed the predicates cumulatively from the background knowledge in their ranked order according to the respective methods. As can be seen, the removal of the first 4 high-level relation features has the biggest influence on the performance. The results after the removal of all 10 high-level relation features while there are only 10 features ranked can be explained by the fact that the predicate to indicate the word ID needs to remain present since it forms the primary key in the relational database representation of the domain.

To show that the effect on the performance is not only due to a reduced number of features, we also removed the features individually (Figure 7.3). The ROC curves and their corresponding scores indicate that the first 4 ranked predicates (CW, RightOf, LeftOf, and Next) reduce the performance, resulting in a lower score then when all predicates are used. Removal of the other predicates results in ROC curves (and scores) that are above the all-feature setting, which indicates that their presence has a negative influence on the performance. This ultimately is the case for DH, which generates a large number of low-level graph features, since every word in the sentence has a dependency head.



(a) Wrapper approach (b) Embedded approach

Figure 7.2: Cumulative feature removal (hedge cue detection)

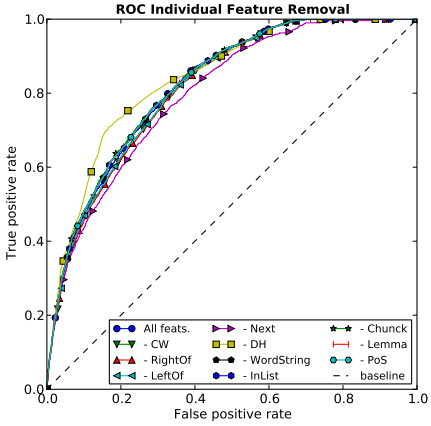


Figure 7.3: Individual feature removal (hedge cue detection)

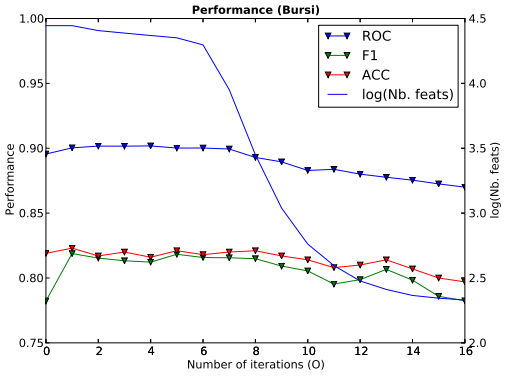


Figure 7.4: Performance with decreasing number of features.

7.5.2 Chemoinformatics: Molecular Toxicity

In Kazius et al. (2005) a dataset of 4337 molecular structures is constructed. The dataset is available with corresponding Ames target data, i.e., each molecule has been tested in a short-term in vitro assay designed to detect genetic damage caused by chemicals and has become the standard test to determine mutagenicity. The class distribution is rather balanced with 2401 mutagens and 1936 nonmutagens. The natural relational representation in terms of atoms and bonds is enriched with a chemistry knowledge base offered by DMax Chemistry Assistant, which allows one to determine the aromaticity perception, the annotation of functional groups, and the relations between them.

The dataset was split in a train and test partition of 3337 versus 1000 molecular structures respectively. In Figure 7.4, the performance in terms of ROC, F1 and accuracy is plotted against the number of iterations, along with the number of retained features (right y axis, in logarithmic scale). We report a substantial stability of the predictive performance w.r.t. the number of features, an indication that relational regularization is effective in selecting the important high-level features.

The analysis of the high-level relational feature ranking (Table 7.6) reveals that the toxicity concept is related to the way functional groups and atoms are connected (fg_connected and bond), rather than to the actual type of atoms and type of functional group (fg_member and atom). It is in fact known that one component in the mutagenicity capacity of a substance depends on the presence of aromatic rings (bond type) forming rigid planes (connected and fused) that can "cut" into DNA/RNA filaments and break them. The results of

High-level feat.	Score
fg_connected	33.29
bond	22.42
fgroup	21.96
fg_fused	6.98
atom	-4.44
fg_member	-21.69
fg_linked	-22.78

Table 7.6: High-level feature ranking (Molecular toxicity)

Triplet	Score
fgroup - fg_connected - fgroup	63.86
fgroup - fg_fused - fgroup	38.72
fgroup - fg_linked - fgroup	28.32
atm - bnd - atm	17.31
fg_member - atm - fgroup	-4.84

Table 7.7: Triplet ranking (Molecular toxicity)

the performance comparison of relational versus non-relational regularization can again be found in Table 7.5. Similar observations as in the case of hedge cue detection can be made.

7.6 Conclusions

In this chapter we have lifted regularization and feature selection to a *relational* level in order to achieve higher interpretability. These techniques take into account the relational structure and topology of a domain and are based on a notion of locality that ties together relevant features in the E/R model. Finally, we have presented two *relational feature selection* approaches; a wrapper technique, which manipulates the declarations in the language bias used by the learner; and an embedded method that incorporates the feature selection as part of the training phase.

In Chapters 4 and 5 we illustrated the advantages of the declarative domain specification and the inclusion of additional background knowledge via declarative feature construction for natural language processing tasks. The techniques developed in this chapter offer a way to assess the importance of these declarative, relational features. This functionality is valuable in the context of natural language processing, to offer new insights into which structural and relational features contribute to a better performance, as illustrated on the hedge cue detection task.

Part III

Probabilistic Rule Learning

Chapter 8

Probabilistic Rule Learning for Knowledge Base Expansion

In many applications the observations are uncertain, and hence, cannot be modelled in a purely logical or deterministic manner. Indeed, consider any context involving the interpretation of sensory information. This situation occurs naturally when performing medical tests, when analyzing images, and when interpreting actions and activities. This also occurs when analyzing text, e.g., during information extraction from structured and unstructured documents. Recently, this task has attracted a lot of attention in the context of Machine Reading, i.e., the automatic extraction of knowledge from text. In this respect, several systems have been developed for extracting relations from words or phrases from text. These relations often have a probability attached that represents the confidence of the method in the extracted relation.

It is then convenient to work with probabilistic examples, that is, examples whose properties and relations can be expressed using probabilistic facts. Furthermore, not only the descriptions of the examples but also their classification can be uncertain. For instance, in a medical context, the diagnosis of a patient on the basis of the existing tests can be uncertain, in image recognition, the identity of a person, or in machine reading, the truth of a sentence. Still in such settings, it is interesting to induce general principles in the form of rules from the available data. This is precisely the topic of this chapter.

We contribute an integrated approach to learning probabilistic relational rules from probabilistic examples and background knowledge that directly upgrades the standard rule learning setting of learning from entailment. It builds on

ProbLog (De Raedt et al., 2007), a simple probabilistic extension of Prolog, and FOIL (Quinlan, 1990). In addition, we generalize the well-known principles and heuristics of rule learning (Fürnkranz and Flach, 2005; Lavrac et al., 2012) to the probabilistic setting. This is a significant extension of the preliminary work by De Raedt and Thon (2010), who introduced the ProbFOIL algorithm for probabilistic rule learning, in that it includes a novel method to probabilistically weigh the induced rules; and we also report on a different and more extensive set of experiments.

The remainder of this chapter is organized as follows. In Section 8.1, some background on probabilistic rule learning is given, as well as a specification of the problem setting. An overview of related work can be found in Section 8.2. Section 8.3 describes ProbFOIL$^+$, our algorithm for learning probabilistic rules from probabilistic data. In Section 8.4 our approach is first evaluated for the propositional case, before turning to the evaluation for the relational case with a case study on NELL, the Never-Ending Language Learner. Finally, Section 8.5 concludes.

This chapter is based on

> Luc De Raedt, Anton Dries, Ingo Thon, Guy Van den Broeck, and Mathias Verbeke. Inducing probabilistic relational rules from probabilistic examples. Submitted.

8.1 Problem Specification: Probabilistic Rule Learning

The field of inductive logic programming has focussed on the learning of rules from examples and background knowledge in a *deterministic setting*. Today, it is realized that purely logical theories do not suffice in many applications and that uncertainty needs to be taken into account and modeled. This is the prime motivation underlying probabilistic logic learning (De Raedt et al., 2008), probabilistic programming (De Raedt and Kimmig, 2013), and statistical relational learning (Getoor and Taskar, 2007). Numerous formalisms, inference mechanisms and results on probabilistic logic learning and statistical relational learning have been reported over the past decade. Nevertheless, we are still lacking a simple upgrade of the basic rule learning setting (as pursued in inductive logic programming), in which the rules, the background knowledge and the examples are all *probabilistic* rather than deterministic. Such an upgraded setting should have the property that, when all probabilities are set to 0 and 1, one obtains the standard rule learning setting pursued in inductive logic programming (learning from entailment (De Raedt, 2008)), and traditional

inductive logic programming principles and systems, such as FOIL (Quinlan, 1990) and Progol (Muggleton, 1995), can be readily applied.

Existing approaches to statistical relational learning do not really satisfy this requirement as they often do not employ Prolog as the underlying representation but rather a representation derived from graphical models (such as Markov Logic, e.g., (Schoenmackers et al., 2010), or Bayesian Logic Programs, e.g., (Raghavan et al., 2012)). Some approaches do learn both the global structure and the parameters of the statistical relational model and do search the space of hypotheses ordering the search space using a form of θ-subsumption. However, their scoring mechanisms differ significantly from those employed by classical rule-learning systems and are typically based on expectation maximisation (EM), e.g., (Kok and Domingos, 2005; Huynh and Mooney, 2008; Bellodi and Riguzzi, 2012). Finally, approaches that target the learning of probabilistic logical rules from examples (Schoenmackers et al., 2010; Raghavan et al., 2012; Doppa et al., 2010) usually take a two-step approach. In the first step, a traditional inductive logic programming system such as FOIL (Quinlan, 1990), Progol (Muggleton, 1995), or Farmer (Nijssen and Kok, 2001) is used to induce the rules and in the second step the probabilities or weights are determined, which leads to a decoupling of the probabilistic and the logical issues rather than an integrated approach.

Based on the logic programming concepts and the specification of ProbLog as introduced in Section 2.2, we are now able to formalize the problem of *inductive probabilistic logic programming* or *probabilistic rule learning* as follows:

Given:

1. $E = \{(x_i, p_i) | x_i$ a ground fact for the unknown target predicate t; $p_i \in [0, 1]$ the target probability of $x_i\}$, the set of examples;
2. a background theory B containing information about the examples in the form of a probabilistic ProbLog program;
3. a loss function $loss(H, B, E)$, measuring the loss of a hypothesis H (that is, a set of clauses) w.r.t. B and E;
4. a space of possible clauses $\mathcal{L}_h$ specified using a declarative bias;
5. the success probability P_s of a query (as defined in Section 2.2.4);

Find: The hypothesis $H \subseteq \mathcal{L}_h$ for which:

$$\arg\min_H loss(H, B, E) = \arg\min_H \sum_{e_i \in E} |P_s(B \cup H \models e_i) - p_i|$$

This loss function aims at minimizing the standard error of the predictions. The reason for this choice is, on the one hand, that it is the simplest possible choice and, on the other hand, that well-known concepts and notions from rule learning and classification carry over to the probabilistic case when this loss function is used as we shall show. Furthermore, this loss function is also used in Kearns and Schapire's probabilistic concept-learning framework (Kearns and Schapire, 1994), a generalization of Valiant's probably approximate correct learning framework to predicting the probability of examples rather than their class. The above definition generalizes that framework further by assuming that also the descriptions of the examples themselves are probabilistic, not just their classes.

There are several interesting observations about this problem setting. First, it generalizes both traditional rule learning and inductive logic programming to a probabilistic setting. The propositional case illustrated in the windsurfing problem from Section 2.2.4 is an instance of probabilistic rule learning. Furthermore, when the background theory also contains relations and possibly clauses defining further predicates, we obtain an inductive probabilistic logic programming setting. In both cases, the original setting is obtained by assuming that the background theory is purely logical and having as only values for the examples 1 and 0; 1 corresponding to the positive examples and 0 to the negative ones. This is in line with the theory of probabilistic logic learning (De Raedt, 2008) and the inductive logic programming setting obtained would be that of learning from entailment because examples are facts that are probabilistically entailed by the theory.

Second, it is also interesting to position this problem in the context of the literature on uncertainty in artificial intelligence. There, one typically makes a distinction between parameter learning and structure learning, the latter being an extension of the former in that in structure learning also the parameters have to be estimated. The probabilistic rule learning problem introduced above is in a sense dual to the parameter estimation problem. Indeed, when estimating parameters, the structure of the model is assumed to be given and fixed, while in parameter learning, the structure is fixed and the parameters are to be learned. Here, only part of the parameters (the probability values) are fixed, namely the probabilities of the examples, whereas the structure, that is, the rules, and their parameters are to be learned.

8.2 Related Work

We start with discussing the relation to traditional propositional rule learners and then continue with related work in machine reading and statistical relational learning.

Early approaches focussed on learning *propositional* rules from data that was automatically extracted from text by information extraction. For example, Nahm and Mooney (2000) presented a technique to learn propositional rules that combines methods from data mining and information extraction. However, it neither involved logic nor probabilistic data.

Many of the more recent rule learning algorithms also attach weights to the learned rules. These weights are sometimes estimated as $Pr(H|B)$, i.e., the probability that the head H of the rule is true given its body B. The resulting rules are often referred to as *probabilistic rules* but, unlike in ProbFOIL$^+$, the example descriptions are deterministic.

In the propositional setting, ProbFOIL$^+$ is also related to the work of Černoch and Železný (2011), who propose a bottom-up approach for tackling ProbFOIL$^+$'s setting in the propositional case. The authors study the task of estimating the probability of a logical rule and solve this problem using a regression-based formulation, expressed as a mixed linear programming problem. The propositional ProbFOIL$^+$ setting is also related to the work on mining frequent itemsets from uncertain data in that the items also hold with a particular probability; cf. Chui et al. (2007). The work on mining uncertain frequent itemsets has been upgraded towards the relational case. For example, Kimmig and De Raedt (2009) study local pattern mining in the context of ProbLog. To this end, a correlated query mining algorithm is introduced. However, instead of learning rules, they mine for patterns whose expected frequency is high.

A large body of related research originates from the machine reading domain, where the goal is to learn inference rules from automatically extracted data and to use the learned rules to expand the knowledge base. With the introduction of NELL (Carlson et al., 2010), a first-order relational learning algorithm similar to FOIL, N-FOIL, was used to learn probabilistic Horn clauses. However, the conditional probability attached to the rules $\hat{Pr}$ is only an estimation, calculated using a Dirichlet prior similar to the m-estimate as $\hat{Pr} = (P + m \times prior)/(P + N + m)$, in which m and $prior$ are fixed constants (Lao et al., 2011). In order to make this approach more scalable and efficient, and to obtain confidence values that take into account the knowledge base, an inference and learning scheme based on random walks in graphs was proposed (Lao et al., 2011). It is based on the path ranking algorithm (PRA) (Lao and Cohen, 2010; Lao et al., 2012). In contrast to N-FOIL, the evidence is based on many existing paths in the knowledge base, and the inference method yields more moderately-confident inferences. ProPPR (Wang et al., 2014c,b) extends PRA to allow learning recursive programs and programs with predicates of arity larger than two and provide an interesting parameter estimation algorithm. Furthermore, also Wang et al. (2014a) sketch an approach to learn rules (using an abductive

template-based approach inspired by the work of Muggleton and Lin (2013)). However, the key difference between these approaches and ours lies in the underlying probabilistic framework. While ProbLog uses a standard possible world semantics that associates degrees of beliefs to individual facts and queries, PRA and ProPPR are based on random walks, which are more like stochastic logic programs (Muggleton, 1996), and which induce a probability distribution over all ground atoms for a particular predicate. While such a distribution is very well suited for sampling ground atoms (or sentences in a grammar), it is well-known that this type of distribution represents a different type of knowledge than the possible worlds one in ProbLog, cf. Cussens (1999), which is arguably more natural in our probabilistic rule learing setting.

A number of other extensions of FOIL exists. The nFOIL extension (Landwehr et al., 2007) integrates FOIL with the Naïve Bayes learning scheme, such that Naïve Bayes is used to guide the search. The kFOIL extension (Landwehr et al., 2006) is a propositionalization technique that uses a combination of FOIL's rule-learning algorithm and kernel methods to derive a set of features from a relational representation. To this end, FOIL searches relevant clauses that can be used as features in kernel methods. As discussed in Section 2.2.3, kLog upgrades this propositionalization technique to graphicalization.

The work of Raghavan et al. (2012) is most related to ProbFOIL$^+$. The authors propose a method to learn probabilistic first-order rules from a large knowledge base of automatically extracted facts. The technique is based on Bayesian Logic Programs (BLPs). Both the structure and the parameters of the BLP are automatically learned, only using the extracted facts. The learned rules in turn are used to derive additional knowledge using BLP inference. Raghavan and Mooney (2013) present a more efficient extension of this work, that allows online rule learning, and takes lexical information from WordNet into account to do the weighting of the rules. In contrast to ProbFOIL$^+$, the knowledge base from which the rules are learnt consists of deterministic facts. Furthermore, the probabilities of the inferred rules are estimated using sample search, an approximate sampling algorithm for Bayesian networks.

Both Doppa et al. (2011) and Sorower et al. (2011) study the problem of learning domain knowledge from texts in the presence of missing data. To this end, the Gricean conversational maxims, a set of principles that enable effective communication, are compiled into Markov Logic. Subsequently, these are inverted via probabilistic reasoning and tailored to learn a set of probabilistic Horn clause rules from facts that are extracted from texts. EM is used to learn the weights of the resulting rules. This work differs from our approach in the same way as explained above for Raghavan et al. (2012).

Jiang et al. (2012) use Markov Logic to clean NELL's knowledge base using the

confidence values of the facts and the ontological constraints already present in the system. However, they do not explicitly learn rules. To overcome some of the limitations of MLNs (e.g., the difficulty to incorporate the confidence values, as all logical predicates must take Boolean truth values), the authors use a number of approximations. To overcome these representational and scalability limitations, Pujara et al. (2013), use Probabilistic Soft Logic (PSL) to extend the approach of Jiang et al. (2012) by including multiple extractors and reasoning about coreferent entities.

(Statistical) Predicate Invention is the discovery of new concepts, properties and relations from data, expressed in terms of the observable ones (Kok and Domingos, 2007). It is related to rule learning in the sense that the learned clauses also represent logic programs that are learned by examples. However, whereas the rules in rule learning are based on the predicates already present in the database, predicate invention adds new ones. Recent work by Muggleton et al. (2014) focussed on Meta-Interpretive Learning, using abduction with respect to a meta-interpreter, and was applied to inductive inference of grammars. Muggleton and Lin (2013) extended this approach for learning higher-order dyadic Datalog fragments. It was tested in the context of NELL, to inductively infer a clause and abduce new facts, and learning higher-order concepts, such as the symmetry of a predicate. Remark though that both approaches are concerned with predicate invention from logic programs.

As for the propositional case, some related work can also be found in the data mining community. Dylla et al. (2013) propose a temporal probabilistic database model for cleaning uncertain temporal facts obtained from information extraction methods. Temporal-probabilistic databases contain data that is valid during a specific time with a given probability. The presented method is based on a combination of temporal deduction rules, temporal consistency constraints an probabilistic inference. Although the method learns from probabilistic facts, the rules and constraints are deterministic. The AMIE system by Galárraga et al. (2013) is related in that it uses inductive logic programming for association rule mining supporting the open word assumption, i.e., statements that are not contained in the knowledge base are not necessarily false. Although the system does not deal with probabilistic data, it thus offers an approach to deal with incomplete evidence for deterministic knowledge bases.

8.3 ProbFOIL+

We now present our algorithm for learning probabilistic clauses, which is a generalization of the mFOIL rule learning algorithm. The outline of the algorithm is shown as Algorithm 3. It follows the typical separate-and-conquer approach (Fürnkranz, 1999) (also known as sequential covering) that is

Algorithm 3 The ProbFOIL$^+$ learning algorithm.

```
1: function ProbFOIL+(target)                          ▷ target is the target predicate
2:     H := ∅
3:     while true do
4:         clause := LearnRule(H, target)
5:         if GlobalScore(H) < GlobalScore(H ∪ {clause}) then
6:             H := H ∪ {clause}
7:         else
8:             return H
9:         end if
10:    end while
11: end function
12: function LearnRule(H, target)
13:    candidates := {x :: target ← true}          ▷ Start with an empty (probabilistic) body
14:    bestrule := (x :: target ← true)
15:    while candidates ≠ ∅ do                                          ▷ Grow rule
16:        nextcandidates := ∅
17:        for each x :: target ← body ∈ candidates do
18:            for each literal ∈ ρ(target ← body) do                ▷ Generate all refinements
19:                if not RejectRefinement(H, bestrule, x :: target ← body) then   ▷ Reject
   unsuited refinements
20:                    nextcandidates := nextcandidates ∪ {x :: target ← body ∧ l}
21:                    if LocalScore (H, x :: target ← body ∧ literal) > LocalScore(H, bestrule)
   then
22:                        bestrule := (x :: target ← body ∧ literal)          ▷ Update best rule
23:                    end if
24:                end if
25:            end for
26:        end for
27:        candidates := nextcandidates
28:    end while
29:    return bestrule
30: end function
```

commonly used in rule learning algorithms. The outer loop of the algorithm, labeled ProbFOIL$^+$, starts from an empty set of clauses and repeatedly adds clauses to the hypothesis until no more improvement is observed with respect to a *global scoring function*. The clause to be added is obtained by the function LearnRule, which greedily searches for the clause that maximizes a *local scoring function*. The resulting algorithm is very much like the standard rule-learning algorithm known from the literature (cf., Fürnkranz and Flach (2005); Mitchell (1997)).

A key difference with the original ProbFOIL algorithm by De Raedt and Thon (2010) is that the hypothesis space $\mathcal{L}_h$ now consists of *probabilistic* rules. While ProbLog and Prolog assume that the rules are definite clauses, in ProbFOIL$^+$ we use probabilistic rules of the form $x :: target \leftarrow body$. Such a rule is actually a short hand notation for the deterministic rule $target \leftarrow body \wedge prob(id)$ and the probabilistic fact $x :: prob(id)$, where id is an identifier that refers to this particular rule. Notice that all facts for such rules are independent of one another, and also that the probability x will have to be determined by the rule learning algorithm. Each call to LocalScore returns the best score that can

be achieved for any value of x. Finally, when returning the best found rule in line 29, the value of x is fixed to the probability that yields the highest local score.

As the *global scoring function*, which determines the stopping criterion of the outer loop, we use *accuracy* which is defined as

$$accuracy_H = \frac{TP_H + TN_H}{M}, \tag{8.1}$$

where M is the size of the dataset.

The *local scoring function* is based on the m-estimate. This is a variant of precision that is more robust against noise in the training data. It is defined as

$$m\text{-}estimate_H = \frac{TP_H + m\frac{P}{N+P}}{TP_H + FP_H + m}, \tag{8.2}$$

where m is a parameter of the algorithm, and P and N indicate the number of positive and negative examples in the dataset, respectively. Increasing the parameter m causes ProbFOIL$^+$ to learn rules that cover more examples. The final hypotheses are more concise.

Both these metrics are based on the number of examples correctly classified as positive (true positives, TP) and the number of examples incorrectly classified as positive (false positives, FP).

These values are part of the contingency table of the hypothesis H

prediction ↓	+	−	← **target**
+	TP_H	FP_H	
−	FN_H	TN_H	
total	P	N	

However, the common definitions for the values in this table only apply to a deterministic case. We now show how we can generalize these concepts to a probabilistic setting where both the examples and the predictions are probabilistic.

8.3.1 Probabilistic contingency table

A key difference between the probabilistic and the deterministic settings is that in the probabilistic setting, each example e_i has a target probability p_i as opposed to a 1/0 value. This means that every example contributes p_i to the positive part of the dataset and $1 - p_i$ to the negative part of the dataset.

This is a generalization of the deterministic setting where $p_i = 1$ for positive examples, and $p_i = 0$ for negative examples. In general, we can define the positive and negative parts of the dataset as

$$P = \sum_{i=0}^{M} p_i, \qquad N = \sum_{i=0}^{M} (1 - p_i) = M - P$$

where M is the number of examples in the dataset.

We can use the same approach to generalize the predictions of a model to a probabilistic setting where a hypothesis H will predict a value $p_{H,i} \in [0, 1]$ for example e_i instead of 0 or 1. In this way we can define a probabilistic version of the true positive and false positive rates of the predictive model as

$$TP_H = \sum_{i=0}^{M} tp_{H,i}, \text{ where } tp_{H,i} = \min(p_i, p_{H,i}),$$

$$FP_H = \sum_{i=0}^{M} fp_{H,i}, \text{ where } fp_{H,i} = \max(0, p_{H,i} - p_i).$$

For completeness we note that $TN_H = N - FP_H$ and $FN_H = P - TP_H$, as was the case in the deterministic setting.

Figure 8.1 illustrates these concepts. If a hypothesis H overestimates the target value of e_i, that is, $p_{H,i} > p_i$ then the true positive part tp_i will be maximal, that is, equal to p_i. The remaining part, $p_{H,i} - p_i$, is part of the false positives. If H underestimates the target value of e_i then the true positive part is only $p_{H,i}$ and the remaining part, $p_i - p_{H,i}$ contributes to the false negative part of the prediction.

The different notions are graphically displayed in Figure 8.2, in which the x-axis contains the examples and the y-axis their probability and all the examples are ordered according to increasing target probability.[1] The areas then denote the respective rates. The deterministic case is illustrated in the Figure 8.2 (right), which shows that in this case the examples take on 1/0 values. Figure 8.2 (left) illustrates this for the probabilistic case. From this picture, it may be clear that the notions of TP, TN, FP and FN correspond to the usual notions of true/false positive/negative rates from the literature in classification, yielding a probabilistic contingency table as shown in Figure 8.1 (right).

[1] The predicted probability is not necessarily monotone.

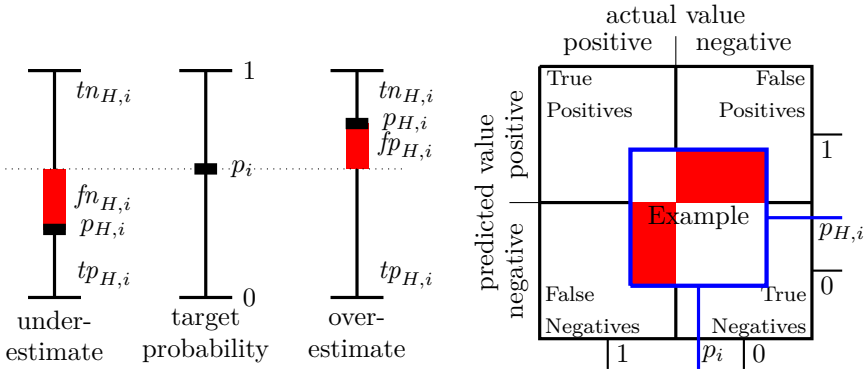


Figure 8.1: The true and false positive and negative part of a single example (left) and the probabilistic contingency table (right).

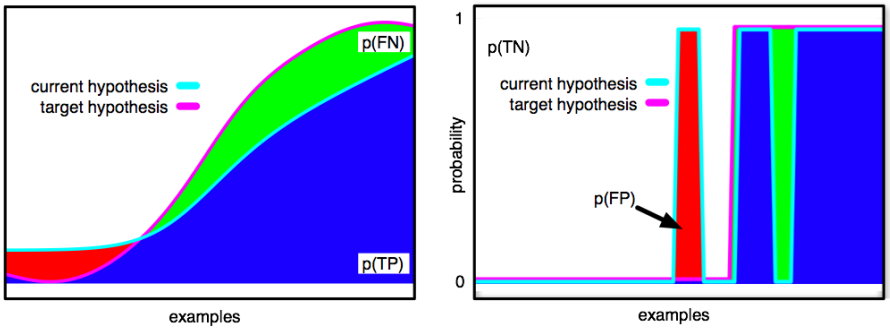


Figure 8.2: The true and false positive and negative part of an entire dataset for the probabilistic case (left), and for the deterministic case (right).

8.3.2 Calculating x

Algorithm 3 builds a set of clauses incrementally, that is, given a set of clauses H, it will search for the clause $c(x) = (x :: c)$ that maximizes the local scoring function, where $x \in [0, 1]$ is a multiplier indicating the clause probability of clause c. The local score of the clause c is obtained by selecting the best possible value for x, that is, we want to find

$$prob_c = \arg\max_x \frac{TP_{H \cup c(x)} + m\frac{P}{N+P}}{TP_{H \cup c(x)} + FP_{H \cup c(x)} + m} \tag{8.3}$$

In order to find this optimal value, we need to be able to express the contingency table of $H \cup c(x)$ as a function of x. As before, we use p_i to indicate the target

value of example e_i. We see that $p_{H\cup c(x),i}$ is a monotone function in x, that is, for each example e_i and each value of x, $p_{H\cup c(x),i} \geq p_{H,i}$ and for each x_1 and x_2, such that $x_1 \leq x_2$, it holds that $p_{H\cup c(x_1),i} \leq p_{H\cup c(x_2),i}$. We can thus define the minimal and maximal prediction of $H \cup c(x)$ for the example e_i as

$$l_i = p_{H,i} \qquad\qquad u_i = p_{H\cup c(1),i}.$$

Note that u_i is the prediction that would be made by the original ProbFOIL algorithm.

For each example e_i, we can decompose $tp_{H\cup c(x),i}$ and $fp_{H\cup c(x),i}$ in

$$tp_{H\cup c(x)} = tp_{H,i} + tp_{c(x),i} \qquad\qquad fp_{H\cup c(x),i} = fp_H + fp_{c(x),i},$$

where $tp_{c(x),i}$ and $fp_{c(x),i}$ indicate the additional contribution of clause $c(x)$ to the true and false positive rates.

As illustrated in Figure 8.3, we can divide the examples in three categories:

E_1 : $p_i \leq l_i$, i.e., the clause overestimates the target value for this example, irrespective of the value of x. For such an example $tp_{c(x),i} = 0$ and $fp_{c(x),i} = x(u_i - l_i)$.

E_2 : $p \geq u$, i.e., the clause underestimates the target value for this example, irrespective of the value of x. For such an example $tp_{c(x),i} = x(u_i - l_i)$ and $fp_{c(x),i} = 0$.

E_3 : $l_i < p_i < u_i$, i.e., there exists a value of x for which the clause predicts the target value for this example perfectly. We call this value x_i and it can be computed as

$$x_i = \frac{p_i - l_i}{u_i - l_i}.$$

Figure 8.4 shows the values of $tp_{c(x),i}$ and $fp_{c(x),i}$ in function of x. The formulae for these functions are

$$tp_{c(x),i} = \begin{cases} x(u_i - l_i) & \text{if } x \leq x_i, \\ p_i - l_i & \text{if } x > x_i \end{cases} \quad \text{and} \quad fp_{c(x),i} = \begin{cases} 0 & \text{if } x \leq x_i, \\ x(u_i - l_i) - (p_i - l_i) & \text{if } x > x_i \end{cases}.$$

We can now determine the contribution to $TP_{c(x)}$ and $FP_{c(x)}$ of the examples in each of these categories. For the examples in E_1, the contributions to $TP_{c(x)}$ and $FP_{c(x)}$ are

$$TP_1(x) = 0 \text{ and } FP_1(x) = x \sum_{i}^{E_1} (u_i - l_i) = xU_1.$$

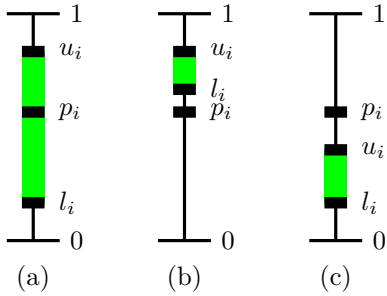


Figure 8.3: Values for l_i, u_i and p_i where (a) it is still possible to perfectly predict p_i with the right value of x, or where p_i will always be (b) overestimated or (c) underestimated.

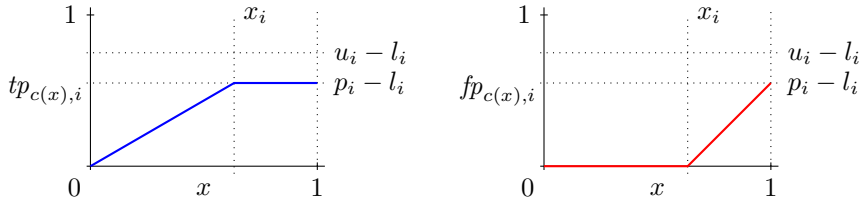


Figure 8.4: True and false positive rate for a single example $e_i \in E_3$ where $l_i < p_i < u_i$.

For the examples in E_2, the contribution to $TP_{c(x)}$ and $FP_{c(x)}$ are

$$TP_2(x) = x \sum_{i}^{E_2} (u_i - l_i) = xU_2 \text{ and } FP_2(x) = 0.$$

For the examples in E_3, the contributions to $TP_{c(x)}$ and $FP_{c(x)}$ are

$$TP_3(x) = x \sum_{i:x \leq x_i}^{E_3} (u_i - l_i) + \sum_{i:x > x_i}^{E_3} (p_i - l_i) = xU_3^{\leq x_i} + P_3^{> x_i},$$

$$FP_3(x) = x \sum_{i:x > x_i}^{E_3} (u_i - l_i) - \sum_{i:x > x_i}^{E_3} (p_i - l_i) = xU_3^{> x_i} - P_3^{> x_i}.$$

By using the fact that $TP_{H \cup c(x)} = TP_H + TP_1(x) + TP_2(x) + TP_3(x)$ and $FP_{H \cup c(x)} = FP_H + FP_1(x) + FP_2(x) + FP_3(x)$ and by reordering terms we

can reformulate the definition of the m-estimate (Equation 8.2) as

$$M(x) = \frac{TP_{H\cup c(x)} + m\frac{P}{N+P}}{TP_{H\cup c(x)} + FP_{H\cup c(x)} + m} = \frac{(U_2 + U_3^{\leq x_i})x + TP_H + P_3^{>x_i} + m\frac{P}{N+P}}{(U_1 + U_2 + U_3)x + TP_H + FP_H + m}. \tag{8.4}$$

In the last step we replaced $FP_3(x) + TP_3(x) = x\sum_i^{E_3}(u_i - l_i) = xU_3$.

By observing that $U_3^{\leq x_i}$ and $P_3^{>x_i}$ are constant on the interval between two consecutive values of x_i, we see that this function is a piecewise non-linear function where each segment is of the form

$$\frac{Ax + B}{Cx + D}$$

where A, B, C and D are constants. The derivative of such a function is

$$\frac{dM(x)}{dx} = \frac{AD - BC}{(CX + D)^2},$$

which is non-zero everywhere or zero everywhere. This means that the maximum of $M(x)$ will occur at one of the endpoints of the segments, that is, in one of the points x_i. By incrementally computing the values of $U_3^{\leq x_i} = \sum_{i:x\leq x_i}^{E_3}(p_i - l_i)$ and $P_3^{>x_i} = \sum_{i:x>x_i}^{E_3}(p_i - l_i)$ in Equation 8.4 for the x_i in increasing order, we can efficiently find the value of x that maximizes the local scoring function.

8.3.3 Significance

In order to avoid learning large hypotheses with many clauses that only have limited contributions, we use a significance test. This test was also used in the mFOIL algorithm (Džeroski, 1993). It is a variant of the likelihood ratio statistic and is defined as

$$LhR(H, c) = 2(TP_{H,c}+FP_{H,c})\left(prec_{H,c}\log\frac{prec_{H,c}}{prec_{true}} + (1 - prec_{H,c})\log\frac{1 - prec_{H,c}}{1 - prec_{true}}\right), \tag{8.5}$$

where

$$TP_{H,c} = TP_{H\cup c} - TP_H, \qquad FP_{H,c} = FP_{H\cup c} - FP_H$$

and

$$prec_{H,c} = \frac{TP_{H,c}}{TP_{H,c} + FP_{H,c}}, \qquad prec_{true} = \frac{P}{P + N}.$$

This statistic is distributed according to χ^2 with one degree of freedom. Note that we use a relative likelihood, which is based on the additional prediction made by adding clause c to hypothesis H. As a result, later clauses will automatically achieve a lower likelihood.

8.3.4 Local stopping criteria

When we analyze the algorithm above, we notice that in the outer loop, the number of positive predictions increases. This means that the values in the first row of the contingency table can only increase (and the values in the second row will decrease). We can formalize this as follows.

Property 1. For all hypotheses H_1, H_2: $H_1 \subset H_2 \rightarrow TP_{H_1} \leq TP_{H_2}$ and $FP_{H_1} \leq FP_{H_2}$.

Additionally, in the inner loop, we start from the most general clause (i.e., the one that always predicts 1), and we add literals to reduce the coverage of negative examples. As a result, the positive predictions will decrease.

Property 2. For all hypotheses H and clauses $h \leftarrow l_1, ..., l_n$ and literals l: $TP_{H \cup \{h \leftarrow l_1,...,l_n\}} \leq TP_{H \cup \{h \leftarrow l_1,...,l_n,l\}}$ and $FP_{H \cup \{h \leftarrow l_1,...,l_n\}} \leq FP_{H \cup \{h \leftarrow l_1,...,l_n,l\}}$

We can use these properties to determine when a refinement can be rejected (line 19 of Algorithm 3). In order for a clause to be a viable candidate it has to have a refinement that

- has a higher local score than the current best rule,
- has a significance that is high enough (according to a preset threshold),
- has a better global score than the current rule set without the additional clause.

In order to determine whether such a refinement exists, we need to be able to calculate these scores for the best possible refinement of a clause. In the best case, such a refinement would eliminate all false positive predictions while maintaining the true positive predictions. This means that $TP^{best}_{H \cup c} = TP_{H \cup c}$ and $FP^{best}_{H \cup c} = FP_H$. From this we can define the best achievable local score for a clause c given theory H as

$$LocalScoreMax(H \cup c) = \frac{TP_{H \cup c} + m\frac{P}{N+P}}{TP_{H \cup c} + FP_H + m}$$

and the maximally achievable significance as

$$LikelihoodRatioMax(c) = -2\,TP_{H\cup c} \log \frac{P}{P+N}$$

Using the same principles we can derive that the third condition above can be reduced to $TP_{H\cup c} > TP_H$.

8.3.5 Additional characteristics

Refinement operator and syntactic restrictions In ProbFOIL, the available variables have different types. These argument types are specified in advance by base declarations. This ensures that arguments are only instantiated with variables of the appropriate type, and in this sense forms a syntactic restriction on the possible groundings. Furthermore, a refinement operator based on modes (Shapiro, 1981) is used. A mode specifies, for each literal argument, whether it can introduce a new variable in the clause or whether it can only use already existing variables.

Beam search The LearnRule function of the ProbFOIL algorithm is based on FOIL and uses hill-climbing search. We replaced this with the rule learning function from mFOIL (Džeroski, 1993), which uses a beam search strategy. This enables to escape from local maxima, which is a typical pitfall of greedy hill-climbing search. Furthermore it uses search heuristics and stopping criteria that improve noise-handling. Also some additional information, such as the symmetry and different variable constraints are used to reduce the search space.

The algorithm is shown in Algorithm 3 (page 142). At the start of the LearnRule function, the clause is initialized with an empty body (lines 13 and 14). Hereafter, the search is started, in which a beam of promising clauses is maintained, as well as the best significant clause found so far (*best rule*).

Relational path finding Relational path finding (Richards and Mooney, 1992) generates clauses by considering the connections between the variables in the example literals, and it is a useful technique to direct the search in first-order rule learning. ProbFOIL uses the approach by Ong et al. (2005) that extends the original relational path finding algorithm such that it uses the base declarations during pathfinding. The use of relational path finding during the rule learning process in ProbFOIL is provided as an option.

8.3.6 Implementation

ProbFOIL$^+$'s evaluation is based on the ProbLog2 implementation[2]. The ProbLog2 inference engine computes the probability of queries in four phases: it grounds out the relevant part of the probabilistic program, converts this to a CNF form, performs knowledge compilation into d-DNNF form and, finally, computes the probability from the obtained d-DNNF structure. This process is described in detail by Fierens et al. (2014).

Due to the specific combinations and structure of ProbFOIL$^+$'s queries, we can apply multiple optimizations[3]:

Incremental grounding While the standard ProbLog2 would perform grounding for each query, ProbFOIL$^+$ uses incremental grounding techniques and builds on the grounding from the previous iteration instead of starting from scratch. This is possible as the rules are constructed and evaluated one literal at-a-time.

Simplified CNF conversion ProbFOIL$^+$'s rulesets require only a subset of the functionality that ProbLog supports. We can therefore use a simplified CNF conversion technique that can exploit the specific structure of ProbFOIL$^+$ evaluations (that is, without support for probabilistic evidence, and only limited support for cycles).

Direct calculation of probabilities Because of the incremental nature of ProbFOIL$^+$'s evaluation, we can often directly compute probabilities without having to resort to (costly) knowledge compilation, for example when we add a literal whose grounding does not share facts with the grounding of the rest of the theory (for which we computed the probability in a previous iteration). This can also significantly reduce the size of the theories that need to be compiled.

Propositional data When propositional data is used, all examples have the same structural component. This means we can construct a d-DNNF for a single example and reuse it to evaluate all other examples.

Range-restricted rules Since in a number of cases, it is desired that the result in rules are *range-restricted*, i.e., that all variables appearing in the head of a clause also appear in its body, ProbFOIL$^+$ offers an option to output only range-restricted rules.

[2] http://dtai.cs.kuleuven.be/problog/

[3] The remainder of the section can be skipped by the reader less familiar with probabilistic programming

8.4 Experiments

In a first set of experiments, we have studied the performance for propositional probabilistic rule learning. To this end, we compared ProbFOIL and ProbFOIL$^+$ against a set of regression learners on Bayesian networks, both for dependent and independent attributes, and partial and full observability. In order to test relational probabilistic rule learning, we considered rule learning in the knowledge base of NELL (Carlson et al., 2010), the never-ending language learner, as a case study.

8.4.1 Propositional probabilistic rule learning: Bayesian Networks

These experiments are motivated by the observation that – in the propositional case – the task can be viewed as that of predicting the probability of the target example from a set of probabilistic attributes. This propositional task can be solved by applying standard regression tasks. Of course, one then obtains regression models, which do not take the form of a set of logical rules that are easy to interpret. While regression can in principle be applied to the propositional case, it is hard to see which regression systems would apply to the relational case. The reason is that essentially *all* predicates are probabilistic (and hence, numeric), a situation that is – to the best of our knowledge – unprecedented in relational learning. Standard relational regression algorithms are able to predict numeric values starting from a relational description, a set of true and false ground facts.

So, in this first experiment, we want to answer the following question:

Q1: How do ProbFOIL and ProbFOIL$^+$ compare to standard regression learners in the propositional case?

Dataset

Bayesian networks naturally lend themselves to test this learning setting. A Bayesian network (BN) is a probabilistic graphical model representing a set of random variables and the conditional dependency relations between them via a directed acyclic graph, i.e., the edges between nodes are directed and there are no loops in the graph. A BN represents a joint probability distribution over the variables in the network, i.e., each variable takes on a particular value, given the value of the other variables. Variables can either be dependent or independent. Two variables are called *dependent* if knowledge of one of the variables can affect the value of the other. Variable *independence* refers to the opposite, i.e., the value of one variable does not affect the value of the other. A

target variable is a variable that one wants to predict, based on evidence, i.e., the values of other variables in the network.

For the experiments, we used BNGenerator[4] to randomly generate Bayesian network structures. The conditional probability tables (CPT) and marginal distributions were left unspecified. Network A has 30 nodes, 40 edges, a maximal degree of 4 and an induced width of 3. Network B has 45 nodes, 70 edges, a maximal degree of 6 and an induced width of 5.

Subsequently, for each of the networks, different instances of the Bayesian network were generated by sampling its CPTs from a beta distribution $\text{Beta}(\alpha, \beta)$. Lower values for α and β make the network more "deterministic" and less "probabilistic". To generate training and test examples for a single network instance, we sampled marginal probabilities for the root nodes from a uniform distribution. These values, together with the inferred probability of the target, make up a single example. Each combination of target attribute and beta distribution is a different learning problem. For each of these, we trained ProbFOIL, ProbFOIL$^+$ and standard regression learners from the Weka suite[5] on 500 training examples. The learned models were evaluated on 500 test examples using the mean absolute error[6], which is 1 minus the accuracy in the rule learning setting.

Learning from Independent Attributes

In the simplest setting – learning from independent attributes – the observed attributes are all root nodes of the Bayesian network, i.e., a node with no parent nodes in the graph.

Table 8.1 and Table 8.2 show the mean absolute error averaged over all target attributes for different combinations of learners and beta functions for network A and network B respectively. In almost all cases, ProbFOIL and ProbFOIL$^+$ are able to outperform the regression learners[7]. When inspecting the results per learner, note that the regression learners have a better performance on the probabilistic networks, whereas ProbFOIL performs well on the deterministic networks. ProbFOIL$^+$ performs well on both.

[4] http://www.pmr.poli.usp.br/ltd/Software/BNGenerator/

[5] For all regression learners, the default parameter settings were used.

[6] As the output of the regression learners can strictly speaking not be seen as probabilities, we used mean absolute error to evaluate the methods instead of the measures based on the probabilistic contingency table.

[7] Note that in all tables, the arguments of ProbFOIL and ProbFOIL$^+$ indicate the values used for m in the m-estimate, the beam width b and the rule significance p respectively.

Table 8.1: Mean absolute error on network A with CPTs $\sim$ Beta(α, β), averaged over all target attributes. Observed nodes are independent.

α, β	1.0000	0.1000	0.0100	0.0010	0.0001	0.00001
ZeroR	0.054	0.11	0.11	0.13	0.12	0.12
LinearRegression	7.7×10^{-3}	0.027	0.024	0.025	0.024	0.024
MultilayerPerceptron	1.8×10^{-3}	8.5×10^{-3}	6.3×10^{-3}	5.8×10^{-3}	5.7×10^{-3}	5.7×10^{-3}
M5P	1.7×10^{-3}	6.7×10^{-3}	4.2×10^{-3}	4.2×10^{-3}	4.0×10^{-3}	4.0×10^{-3}
M5P -R -M 4.0	0.013	0.031	0.026	0.029	0.028	0.028
SMOreg	7.7×10^{-3}	0.027	0.024	0.026	0.025	0.025
ProbFOIL(1,15,0.0)	0.069	0.051	5.9×10^{-4}	1.6×10^{-7}	1.6×10^{-7}	1.6×10^{-7}
ProbFOIL$^+$(1,1,0.0)	1.8×10^{-3}	3.0×10^{-3}	10.0×10^{-5}	1.6×10^{-7}	1.6×10^{-7}	1.6×10^{-7}

Table 8.2: Mean absolute error on network B with CPTs $\sim$ Beta(α, β), averaged over all target attributes. Observed nodes are independent.

α, β	1.0000	0.1000	0.0100	0.0010	0.0001	0.00001
ZeroR	0.023	0.077	0.085	0.093	0.096	0.096
LinearRegression	4.4×10^{-3}	0.021	0.022	0.023	0.020	0.020
MultilayerPerceptron	9×10^{-4}	5.4×10^{-3}	7.2×10^{-3}	6.7×10^{-3}	5.9×10^{-3}	5.9×10^{-3}
M5P	1.5×10^{-3}	6.7×10^{-3}	8.8×10^{-3}	8.7×10^{-3}	7.3×10^{-3}	7.3×10^{-3}
M5P -R -M 4.0	5.8×10^{-3}	0.024	0.025	0.028	0.027	0.027
SMOreg	4.4×10^{-3}	0.021	0.022	0.023	0.020	0.020
ProbFOIL(1,15,0.0)	0.077	0.029	5.2×10^{-3}	9.4×10^{-8}	4.1×10^{-8}	4.1×10^{-8}
ProbFOIL$^+$(1,5,0.0)	2.3×10^{-3}	3.0×10^{-3}	2.9×10^{-4}	9.4×10^{-8}	4.1×10^{-8}	4.1×10^{-8}

Learning from Dependent Attributes, Full Observability

When learning from dependent attributes with full observability, the observed nodes are no longer the root nodes. Consequently, their probabilities are not independent anymore. There is, however, an observed node on every path from a root node to a target node, allowing for full observability and the possibility of rediscovering the model the data was drawn from. The results for network B are listed in Table 8.3. For the more probabilistic networks (higher α and β), the regression learners outperform ProbFOIL, whereas ProbFOIL$^+$ performs on par. Both ProbFOIL and ProbFOIL$^+$ outperform the regression learners on the deterministic networks.

Learning from Dependent Attributes, Partial Observability

By dropping the full observability, we can no longer learn the perfect model. As can be seen from Table 8.4 both ProbFOIL and ProbFOIL$^+$ still outperform the regression learners for most cases, but the advantage is far less clear than

Table 8.3: Mean absolute error on network B with CPTs $\sim$ Beta(α, β), averaged over all target attributes. Observed nodes are **dependent**. There is **full** observability.

α, β	1.0000	0.1000	0.0100	0.0010	0.0001	0.00001
ZeroR	0.023	0.077	0.085	0.093	0.096	0.096
LinearRegression	2.6×10^{-3}	0.018	0.021	0.020	0.018	0.018
MultilayerPerceptron	4×10^{-4}	3.1×10^{-3}	5.7×10^{-3}	3.9×10^{-3}	3.3×10^{-3}	3.3×10^{-3}
M5P	7×10^{-4}	4.9×10^{-3}	6.5×10^{-3}	5.4×10^{-3}	4.4×10^{-3}	4.4×10^{-3}
M5P -R -M 4.0	5.2×10^{-3}	0.021	0.023	0.025	0.024	0.024
SMOreg	2.6×10^{-3}	0.017	0.021	0.020	0.017	0.017
ProbFOIL(1,10,0.0)	0.015	0.012	1.9×10^{-3}	9.4×10^{-8}	4.2×10^{-8}	4.2×10^{-8}
ProbFOIL$^+$(1,5,0.0)	3.9×10^{-3}	3.9×10^{-3}	5.3×10^{-4}	2.8×10^{-7}	4.2×10^{-8}	4.2×10^{-8}

in the previous cases, as is to be expected given the partial observability.

Table 8.4: Mean absolute error on network B with CPTs $\sim$ Beta(α, β), averaged over all target attributes. Observed nodes are **dependent**. There is **partial** observability.

α, β	1.0000	0.1000	0.0100	0.0010	0.0001	0.00001
ZeroR	0.023	0.077	0.085	0.093	0.096	0.096
LinearRegression	4.8×10^{-3}	0.019	0.026	0.032	0.034	0.034
MultilayerPerceptron	1.2×10^{-3}	2.9×10^{-3}	9.7×10^{-3}	0.020	0.031	0.032
M5P	1.6×10^{-3}	6.1×10^{-3}	0.022	0.027	0.027	0.027
M5P -R -M 4.0	6.2×10^{-3}	0.022	0.033	0.040	0.040	0.040
SMOreg	4.5×10^{-3}	0.018	0.022	0.029	0.034	0.034
ProbFOIL(1,15,0.0)	0.020	0.023	0.012	0.015	0.015	0.015
ProbFOIL$^+$(1,10,0.0)	9.5×10^{-3}	0.011	0.011	0.013	0.013	0.013

A1: In almost all cases both ProbFOIL and ProbFOIL$^+$ perform on par or outperform the standard regression learners, which demonstrates their advantage for propositional probabilistic rule learning. Furthermore, in all cases, similar or better results are obtained by ProbFOIL$^+$ when compared to ProbFOIL, illustrating the added value of learning probabilistic rules.

8.4.2 Relational probabilistic rule learning: NELL

The task to extract information from unstructured or semi-structured documents has recently attracted an increased amount of attention in the context of Machine Reading. Due to the huge body of information, this is especially interesting

concerning Web data. Several systems, such as OLLIE (Open Language Learning for Information Extraction) (Mausam et al., 2012) have recently been developed. Here we focused on NELL[8], the Never-Ending Language Learner (Carlson et al., 2010).

NELL's goal is to extract structured information from unstructured web pages, which is referred to as the *reading* task. As a machine learning system, NELL also wants to learn to perform this task better day after day (i.e., the *learning* task). Started in January 2010, and running 24 hours a day 7 days a week, the goal is to build a knowledge base of structured information that reflects the contents of the Web. So far, NELL has accumulated over 50 million candidate beliefs with different levels of confidence, and it has high confidence in about 2 million of these.

Some of the facts that are stated in the text do not need information extraction, as they can be inferred from the already extracted facts. For other facts, plain text-based information extraction techniques are insufficient, since they are implicit and thus can only be inferred by means of reasoning. Although extracting these implicit facts and reasoning about them is obvious for the human reader, it is not for a computer system. Since hand-crafting such inference rules would be a tedious task, several rule learning approaches have been used for knowledge base expansion, as discussed in Section 8.2.

Furthermore, reasoning about the facts in the knowledge base is impeded by the fact that the extracted relations are often noisy. A plain logical rule learning approach in this setting can even increase the noise. Suppose that the knowledge base contains a number of facts that were extracted with a low confidence by the information extraction system. These examples can cause the addition of a new rule to the reasoning engine, that adds inconsistent facts to the knowledge base.

In order to remedy this, with ProbFOIL$^+$, the confidence scores that are attached to the extracted facts and relationships can be used in the rule learning process, and utilized to obtain a probability for the learned rules.

In this case study, we answered the following question:

Q2: How does ProbFOIL$^+$ perform for relational probabilistic rule learning in the context of a probabilistic knowledge base?

[8] http://rtw.ml.cmu.edu

8.4.3 Dataset

In order to test probabilistic rule learning for facts extracted by NELL, we extracted the facts for all predicates related to the sports domain from iteration 850 of the NELL knowledge base.[9] A similar dataset has already been used in the context of meta-interpretive learning of higher-order dyadic Datalog (Muggleton and Lin, 2013). Our dataset contains 10567 facts. The number of facts per predicate is listed in Table 8.5. Each predicate has a probability value attached.

Table 8.5: Number of facts per predicate (NELL sports dataset).

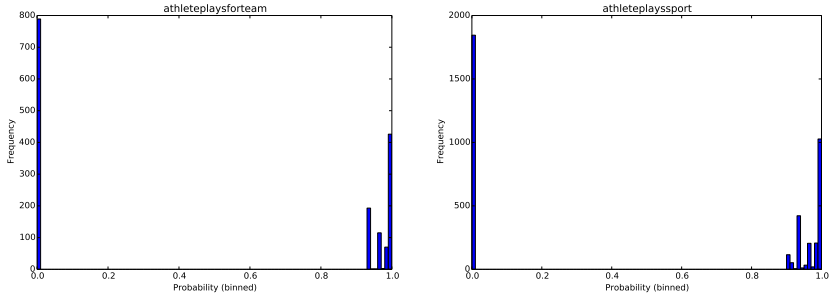
athletealsoknownas(athlete,athlete)	7	coachalsoknownas(coach,coach)	5
athletebeatathlete(athlete,athlete)	130	coachesinleague(coach,league)	69
athletecoach(athlete,coach)	20	coachesteam(coach,team)	150
athletehomestadium(athlete,stadium)	21	coachwontrophy(coach,tournament)	28
athleteinjuredhisbodypart(athlete,bodypart)	157	teamalsoknownas(team,team)	273
athleteledsportsteam(athlete,team)	246	teamhomestadium(team,stadium)	135
athleteplaysforteam(athlete,team)	808	teammate(athlete,athlete)	36
athleteplaysinleague(athlete,league)	1197	teamplaysagainstteam(team,team)	2848
athleteplayssport(athlete,sport)	1899	teamplaysincity(team,city)	384
athleteplayssportsteamposition(athlete,sportsteamposition)	196	teamplayssport(team,sport)	340
athletessuchasathletes(athlete,athlete)	46	teamplaysinleague(team,league)	1229
athletewinsawardtrophytournament(athlete,tournament)	126	teamwontrophy(team,trophy)	217

Table 8.5 also shows the types that were used for the variables in the base declarations for the predicates. As indicated in Section 8.3.5, this typing of the variables forms a syntactic restriction on the possible groundings and ensures that arguments are only instantiated with variables of the appropriate type. Furthermore, the LearnRule function of the ProbFOIL algorithm is based on mFOIL and allows one to incorporate a number of variable constraints using the refinement operator based on modes. As outlined in Section 8.3.5, a mode specifies whether a literal argument can introduce a new variable in the clause or whether it can only use already existing variables. To reduce the search space, we imposed the constraint that each clause can introduce at most one new variable.

8.4.4 Evaluation

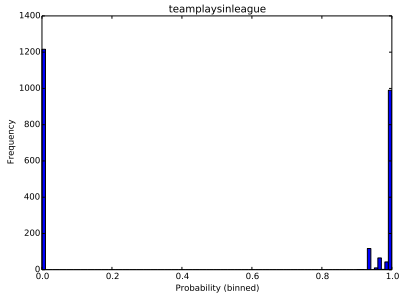
In order to illustrate relational probabilistic rule learning with ProbFOIL$^+$ in the context of NELL, we learned rules for each binary predicate with more than five hundred facts using 3-fold cross-validation. In order to create the folds, for each target predicate, the facts were randomly split into 3 parts. Each fold consists of

[9] Obtained from `http://rtw.ml.cmu.edu/rtw/resources`.

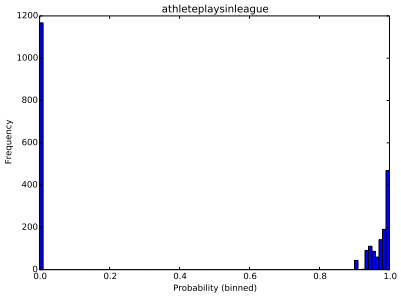


(a)

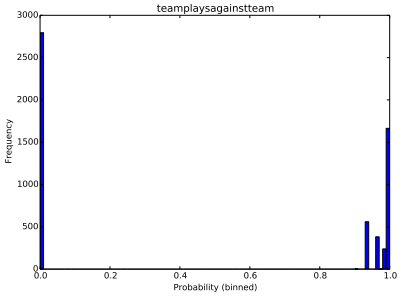
(b)



(c)



(d)



(e)

Figure 8.5: Histogram of probabilities for each of the binary predicates with more than 500 facts: (a) athleteplaysforteam; (b) athleteplayssport; (c) teamplaysinleague; (d) athleteplaysinleague; and, (e) teamplaysagainstteam.

all non-target predicates and a part of the target predicates. We only reported the rules that are learned on the first fold. Similar rules were obtained on the other folds. For each predicate, we also reported the probabilistic contingency table. We compared ProbFOIL$^+$ (referred to as **setting 4**) against three other settings:

Setting 1 All predicates in the training set and all non-target predicates in the test set are deterministic. To this end, as usual in NELL, we interpret each respective example with a probability higher than 0.75 as a positive one, and each example with a lower probability as a negative one.

Setting 2 Similar to setting 1, but probabilistic predicates in the test set are used.

Setting 3 Probabilistic predicates in both train and test set are used, and rules are learned using ProbFOIL, the deterministic rule learning version of ProbFOIL$^+$.

Similar to previous related work (Carlson et al., 2010; Schoenmackers et al., 2010; Raghavan et al., 2012; Raghavan and Mooney, 2013), we used precision as our primary evaluation measure. It measures the fraction of the probabilistic inferences that are deemed correct. Measuring the true recall is impossible in this context, since it would require *all* correct facts for a given target predicate. For example, it is possible that correct facts are inferred using the obtained rules, which are not (yet) present in the knowledge base, and consequently are not reflected in the recall score. We also plotted the predicted against the target probabilities in Figure 8.6.

For all predicates, the m-estimate's m value was set to 1 and the beam width to 5. The value of p for rule significance was set to 0.99.[10] The option to only retain range-restricted rules was used. Furthermore, to avoid a bias towards the majority class, both the train and test examples were balanced, i.e., a part of the negative examples was removed to balance the number of positives. Figure 8.5 shows histograms of the distribution of the probabilities after class balancing for each of the tested predicates. For all settings, we also used relational path finding.

[10] As discussed before, remind that increasing the parameter m causes ProbFOIL$^+$ to learn rules that cover more examples, resulting in more concise hypotheses. The rule significance parameter p controls the contribution of additional clauses when they are added to the learned hypothesis, in order to avoid learning large hypotheses with many clauses that only have limited contributions.

athleteplaysforteam(person,team)

```
0.9375::athleteplaysforteam(A,B) ← athleteledsportsteam(A,B).
0.9675::athleteplaysforteam(A,B) ← athleteledsportsteam(A,V_1),
     teamplaysagainstteam(B,V_1).
0.9375::athleteplaysforteam(A,B) ← athleteplayssport(A,V_1),
     teamplayssport(B,V_1).
0.5109::athleteplaysforteam(A,B) ← athleteplaysinleague(A,V_1),
     teamplaysinleague(B,V_1).
```

Listing 8.1: Learned rules for the `athleteplaysforteam` predicate (fold 1).

Table 8.6: Probabilistic contingency table (`athleteplaysforteam`).

prediction↓	+	−	← **target**
+	344.20	94.04	
−	446.15	712.61	

Table 8.7: Scores per setting for $m = 1$ and $p = 0.99$ (`athleteplaysforteam`).

Setting	Precision
1	76.01
2	72.92
3	77.77
4	78.54

Table 8.8: Scores per setting for $m = 1000$ and $p = 0.9$ (`athleteplaysforteam`).

Setting	Precision
1	70.36
2	70.75
3	73.46
4	73.77

athleteplayssport(person,sport)

```
0.9070::athleteplayssport(A,B) ← athleteledsportsteam(A,V_2),
     teamalsoknownas(V_2,V_1), teamplayssport(V_1,B),
     teamplayssport(V_2,B).
0.9070::athleteplayssport(A,B) ← athleteplaysforteam(A,V_2),
     teamalsoknownas(V_2,V_1),teamplayssport(V_1,B),
     teamplayssport(V_2,B),teamalsoknownas(V_1,V_2).
0.9070::athleteplayssport(A,B) ← athleteplaysforteam(A,V_1),
     teamplayssport(V_1,B).
```

Listing 8.2: Learned rules for the `athleteplayssport` predicate (fold 1).

Table 8.9: Probabilistic contingency table (`athleteplayssport`).

prediction↓	+	−	← **target**
+	191.34	9.69	
−	1655.00	1887.96	

Table 8.10: Scores per setting (`athleteplayssport`).

Setting	Precision
1	93.99
2	94.77
3	92.76
4	95.18

Table 8.11: Scores per setting for $m = 1000$ and $p = 0.9$ (`athleteplayssport`).

Setting	Precision
1	93.46
2	94.96
3	93.12
4	95.15

teamplaysinleague(team,league)

```
0.9585::teamplaysinleague(A,B) ← athleteplaysforteam(V_1,V_2),
     coachesinleague(V_1,B),teamplaysagainstteam(A,V_2).
0.9294::teamplaysinleague(A,B) ← athleteplaysinleague(V_1,B),
     coachesteam(V_1,V_2),teamplaysagainstteam(V_2,A),
     athleteplaysforteam(V_1,V_2),athleteledsportsteam(V_1,V_2).
0.96878::teamplaysinleague(A,B) ← coachesinleague(V_1,B),
     coachesteam(V_1,V_2),teamplaysagainstteam(V_2,A).
```

Listing 8.3: Learned rules for the `teamplaysinleague` predicate (fold 1).

Note that the scores for the `teamplaysinleague` predicate are not available for setting 2, as the learned rules resulted in too many groundings, due to which the evaluation did not halt in a reasonable time. This can be explained by the fact that in this setting a (large) number of very specific rules are learned during training, to approximate the probabilities of the target predicate facts. Consequently, these rules are applicable to a large number of combinations of variables, which results in a large amount of groundings. Note however that for the other setting of m and p, the evaluation succeeded.

Table 8.12: Probabilistic contingency table (`teamplaysinleague`).

prediction↓	+	−	← **target**
+	96.14	3.84	
−	1121.58	1223.43	

Table 8.13: Scores per setting (`teamplaysinleague`).

Setting	Precision
1	94.36
2	94.55
3	93.82
4	96.16

Table 8.14: Scores per setting for $m = 1000$ and $p = 0.9$ (`teamplaysinleague`).

Setting	Precision
1	78.46
2	N/A
3	88.82
4	89.43

athleteplaysinleague(person,league)

```
0.9286::athleteplaysinleague(A,B) ← athleteledsportsteam(A,V_1),
     teamplaysinleague(V_1,B).
0.7868::athleteplaysinleague(A,B) ← athleteplaysforteam(A,V_2),
     teamalsoknownas(V_2,V_1),teamplaysinleague(V_1,B).
0.9384::athleteplaysinleague(A,B) ← athleteplayssport(A,V_2),
     athleteplayssport(V_1,V_2), teamplaysinleague(V_1,B).
0.9024::athleteplaysinleague(A,B) ← athleteplaysforteam(A,V_1),
     teamplaysinleague(V_1,B).
```

Listing 8.4: Learned rules for the `athleteplaysinleague` predicate (fold 1).

teamplaysagainstteam(team,team)

```
0.9375::teamplaysagainstteam(A,B) ← teamalsoknownas(A,V_1),
     teamplaysinleague(B,V_2),teamplaysinleague(V_1,V_2),
     teamplaysinleague(A,V_2).
0.5009::teamplaysagainstteam(A,B) ← athleteplaysforteam(V_2,A),
     athleteplaysinleague(V_2,V_1),teamplaysinleague(B,V_1).
```

Listing 8.5: Learned rules for the `teamplaysagainstteam` predicate (fold 1).

Note that the scores for the `teamplaysagainstteam` predicate are only available for settings 3 and 4, as the rules learned in setting 1 and 2 resulted in too many groundings, due to which the evaluation did not halt in a reasonable time.

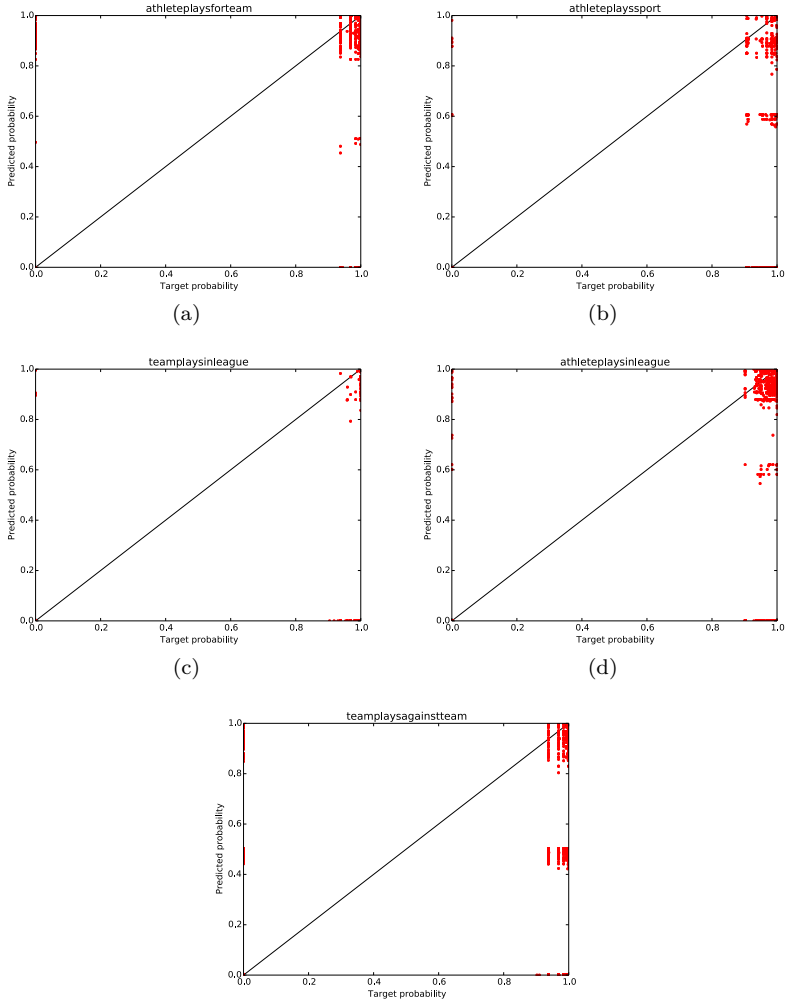


Figure 8.6: Predicted versus target probabilities for 3-fold cross-validation for each of the binary predicates with more than 500 facts: (a) athleteplaysforteam; (b) athleteplayssport; (c) teamplaysinleague; (d) athleteplaysinleague; and, (e) teamplaysagainstteam.

Table 8.15: Probabilistic contingency table (`athleteplaysinleague`).

prediction↓	+	−	← **target**
+	734.51	24.20	
−	433.56	1171.73	

Table 8.16: Scores per setting (`athleteplaysinleague`).

Setting	Precision
1	78.48
2	89.53
3	79.41
4	96.81

Table 8.17: Scores per setting for $m = 1000$ and $p = 0.9$ (`athleteplaysinleague`).

Setting	Precision
1	73.82
2	74.75
3	79.41
4	79.60

Table 8.18: Probabilistic contingency table (`teamplaysagainstteam`).

prediction↓	+	−	← **target**
+	1633.31	252.30	
−	1161.59	2594.80	

Table 8.19: Scores per setting (`teamplaysagainstteam`).

Setting	Precision
1	N/A
2	N/A
3	80.80
4	86.62

Table 8.20: Scores per setting for $m = 1000$ and $p = 0.9$ (`teamplaysagainstteam`).

Setting	Precision
1	73.88
2	73.38
3	80.48
4	81.87

8.4.5 Discussion

As can be observed, for all predicates, interpretable and meaningful rules are obtained. ProbFOIL$^+$ is able to outperform both the two baseline settings that used deterministic training data, and ProbFOIL. In order to avoid overfitting due to too specific rules, we also tested all settings with a high m-value (1000), and a rule significance p of 0.9. Also in this case, one can observe that ProbFOIL$^+$

is able to outperform the other settings. With these settings, ProbFOIL$^+$ learns more deterministic rules. This can also be seen from results obtained with ProbFOIL (Setting 3), which are now more similar to the ones obtained with ProbFOIL$^+$. Furthermore, when testing the obtained rule sets on their respective training sets, similar results as on the test sets were obtained, indicating the generalizability of the learned rules.

The evaluation setting for the machine reading setting is also limited by the available data, which should be taken into account when interpreting these results. First of all, as illustrated by the histograms in Figure 8.5, the distribution of the probabilities in the NELL dataset is very skewed. Furthermore, the dataset also contains a number of predicates for which only a small amount of facts are available in the knowledge base. Finally, the confidence scores that are currently attached to the facts in NELL are a combination of the probability output by the learning algorithm and a manual evaluation. Recent work on improving the quality of the estimated accuracy in NELL was performed by Platanios et al. (2014), but, to the best of our knowledge, is not yet integrated in NELL. Even under these circumstances, ProbFOIL$^+$ performs better than a pure deterministic approach.

A2: ProbFOIL$^+$ obtains promising results for relational probabilistic rule learning. Its use can be valuable for expanding a probabilistic knowledge base, as illustrated in the context of NELL.

8.5 Conclusions

We have introduced a novel setting for probabilistic rule learning, in which probabilistic rules are learned from probabilistic examples. The developed ProbFOIL$^+$ algorithm for solving it combines the principles of the rule learner FOIL with the probabilistic Prolog, ProbLog. The result is a natural probabilistic extension of inductive logic programming and rule learning. We evaluated the approach against regression learners, and showed results on both propositional and relational probabilistic rule learning. Furthermore, we explored its use for knowledge base expansion in the context of NELL, the Never-Ending Language Learner.

8.6 Software and Datasets

ProbFOIL$^+$ and the datasets used in this chapter in ProbFOIL$^+$ format can be downloaded from `dtai.cs.kuleuven.be/probfoil`.

Part IV

Finale

Chapter 9

Conclusions and Future Work

9.1 Summary and Conclusions

While understanding natural language is easy for humans, it is complex for computers. The main reasons for this can be found in the structural nature and the inherent ambiguity of natural language. Correctly interpreting language therefore requires one to take into account the necessary context. In order to perform language understanding by means of machine learning techniques, an appropriate representation is required that takes into account this relational information and integrates the necessary background knowledge into the learning process.

Statistical relational learning is well-suited to represent this relational information, and to incorporate the necessary context and background knowledge for natural language understanding. Furthermore, because of the probabilistic nature of statistical relational learning, it becomes possible to deal with linguistic ambiguity. The goal of this thesis was to investigate the promise of statistical relational learning for natural language processing and to provide evidence for the utility of this approach.

To this end, we focused on four research questions, which were presented in the introduction. We will now review these questions and their respective answers provided in the thesis.

Q1 *What are the advantages of the relational representation and the declarative approach, as offered by SRL, for natural language learning?*

Statistical relational learning has a number of properties that are especially advantageous for natural language learning. First of all, its relational representation proves to be well-suited to represent the structural and relational features that are important for current (semantic) natural language processing tasks. It allows one to represent both the lower-level features (e.g., the lexico-syntactic features at word level) and higher-level (syntactical) structure and (semantic) relations. When looking back at the introductory example at the start of this thesis, we have shown that the relational representation offered by statistical relational learning, and in particular by the graph kernel–based learning approach of kLog, is able to integrate all the necessary contextual information in a flexible way. We have illustrated this for binary sentence classification on the hedge cue detection task (Chapter 4).

Second, the high-level declarative specification of the domain offers an increased expressivity of the domain model and interpretability of the results, especially when contrasted with traditional propositional approaches. We have used the hedge cue detection task to evaluate several types of machine learning systems along two dimensions; the relational representation was contrasted with propositional approaches for both lazy and eager learning. The results show that relational representations are useful, especially for dealing with sentences in which complex, long-distance dependencies amongst constituents need to be captured. The relational representation also enables one-step classification, without the need for an additional thresholding parameter to go from word-level to sentence-level predictions. As part of this investigation, a novel lazy relational learning technique was proposed, which shows that the relational feature construction approach of kLog can be used in an eager type of learner as well as in a memory-based learning setting.

Third, the declarative approach also offers the opportunity to encode additional background knowledge using logical predicates. This allows the learner to exploit additional structural regularities and improve its generalization power. We have illustrated this on the automatic identification of PICO categories in medical abstracts, a multiclass, multilabel sentence classification task, in which we used declarative feature construction to incorporate document context (Chapter 5). A distinguishing property of kLog is that it allows one to use the relational high-dimensional feature space generated by the graph kernel as input for any statistical learner.

Q2 *Can an SRL system be tailored for use in natural language learning, in order to achieve a full relational learning framework for NLP?*

In order to come to a routine application of SRL for natural language learning, a first step is to develop SRL frameworks that abstract away from the underlying details and offer a fully declarative specification of the learning problem. In natural language learning, a lot of attention needs to be given to the modeling of the task. The process of modeling a task in a relational representation that can be used as input to an SRL framework is often seen as an obstacle to applying SRL methods, with the consequence that one often resorts to the traditional propositional methods. To this end, we presented kLogNLP, a natural language processing module for kLog (Chapter 6). It enriches kLog with NLP-specific preprocessors, resulting in a full relational learning pipeline for NLP tasks, embedded in an elegant and powerful declarative machine learning framework. The presented model is a generalization of the models that were used for the tasks discussed in the context of research question **Q1** and is fit to be used as a basis for most NLP tasks. The model can be tailored to meet the needs of the task at hand, and is subsequently used to dynamically transform the dataset into the graph-based relational format of kLog by interfacing with different existing libraries and toolkits. For example, the features used in Figure 1.1 for the introductory example can easily be generated and represented using kLogNLP. The resulting relational representation serves as input for the rest of the kLog workflow and enables one to exploit the advantages of its distinguishing characteristics.

Q3 *How can the importance of declarative, relational features be assessed in the case of statistical relational learning?*

From a linguistic perspective, getting new insights into which structural and relational features contribute to a better performance is valuable for natural language processing. This formed our motivation for the techniques developed in Chapter 7, which offer a way to assess the importance of declarative, relational features. To this end, we have lifted regularization and feature selection to a relational level. These techniques take into account the relational structure and topology of a domain and are based on a notion of locality that ties together relevant features in the E/R model. Furthermore, we have presented two relational feature selection approaches; a wrapper technique, which manipulates the declarations in the language bias used by the learner; and an embedded method that incorporates feature selection as part of the training phase. The functionality of these techniques for natural language processing was illustrated on the hedge cue detection task.

Q4 *How can statistical relational learning be leveraged to reason about and extend probabilistic relational information in the context of Machine Reading?*

In the context of Machine Reading, several systems have been developed to extract relations from text. These relations often have a probability attached that represents the confidence of the method in an extracted relation. These relations and their attached probabilities can be used to extract useful knowledge. In Chapter 8, we have introduced a novel setting for probabilistic rule learning, in which probabilistic rules are learned from probabilistic examples. The resulting algorithm, ProbFOIL$^+$, combines the principles of the rule learner FOIL with ProbLog, a probabilistic Prolog. The result is a natural probabilistic extension of inductive logic programming and rule learning. We evaluated the approach against regression learners, and showed results on both propositional and relational probabilistic rule learning. Furthermore, we explored its use for knowledge base expansion in the context of NELL, the Never-Ending Language Learner. The obtained rules can be used to expand the knowledge base, taking into account the probabilistic nature of the relations from which they were induced.

In general, we can conclude that the distinguishing characteristics of statistical relational learning, namely its relational representation, the declarative approach, and its probabilistic nature, offer several advantages that prove particularly useful for natural language learning. The main goal of this thesis was to strengthen this observation and offer additional insights in these advantages. However, many opportunities for future research remain.

9.2 Discussion and Future Work

We end this thesis with a discussion of four promising directions for future work, namely 1) structured output learning, 2) declarative kernel design, 3) domain-specific and continuous features, and 4) machine reading.

Structured output learning In the hedge cue detection task presented in Chapter 4, the goal was to predict a single output, i.e., the sentence was either hedged or not hedged. The structure was thus only present in the input domain to represent the relations in the sentences to classify. However, for a number of natural language learning tasks, one also wants to predict structure in the output domain. This is referred to as *structured output learning* (as defined in Section 2.1.2). It requires the learner to classify several properties simultaneously. In its simplest form, this task consists of predicting a set of classes per example, as we illustrated for the identification of evidence-based medicine categories (Chapter 5). However, for some tasks, the output is more

complex, for example in the case of parse trees or coreference chains. Although a number of approaches using statistical relational learning for structured output exist (e.g., max-margin Markov Logic Networks, which combine MLNs with structural SVMs (Huynh and Mooney, 2009)), these still lack an intuitive representation of the structural constraints between the input and output space and inside the output space. Statistical relational learning can offer new opportunities in this respect. Although this could be of great benefit to natural language learning tasks, this also brings a number of challenges which come with the complexity caused by the large number of dependencies between the input and output domains.

Declarative kernel design To represent linguistic information, one needs to be able to deal with several types of structures, e.g., sequences to represent the order of words in a sentence or sentences in a document, trees to represent the parse structure, or graphs to represent coreferent entities or ontological information. This also requires methods to calculate the similarity between these structures. To this end, kLog, the graph–kernel based relational learning framework used in this thesis, used the NSPDK graph kernel. This kernel uses pairs of subgraphs which are characterized by an entity or relation which is the kernel point, and a radius and distance to limit the context around this point that is taken into account. While this kernel proved very effective for natural language learning tasks, one could also consider (combinations of) kernels that explicitly take into account these specific structures during the similarity calculation. To this end, an extension of the logical language could be considered that allows the user to construct the overall kernel in a declarative way. This could result in decomposition kernels that are tailored to the natural language learning task under consideration. Initial work on declarative kernels (Cumby and Roth, 2003; Frasconi et al., 2005) has proven particularly useful in the context of natural language processing (Costa et al., 2006), and formed the inspiration for kLog.

In Chapter 7, we presented methods for relational regularization and feature ranking. We limited the high-level relational features to be ranked to single entities or paths of the type entity-relation-entity. To increase the flexibility of the approach, the methods should offer the possibility to consider more complex patterns. In addition to the specification of the kernel, the declaration of these patterns could also form a part of the language bias.

Domain-specific and continuous features For the natural language learning tasks presented in Chapters 4 and 5, we focused on the integration of a diverse set of linguistic features. These features also formed the basis for the kLogNLP model presented in Chapter 6. For a number of these features, domain-specific preprocessing tools were used (e.g., the use of biomedical databases to tag

named entities and the use of a tailored dependency tagger trained to parse biomedical text). These domain-specific preprocessors can also be included in kLogNLP (e.g., the BLLIP parser with the self-trained biomedical parsing model (McClosky, 2010)).

Besides using domain-dependent preprocessing tools, the relational representation makes it possible to include additional domain-specific information. For example, connecting each biomedical named entity to a relevant part of the SNOMED biomedical ontology[1] or linking person names to the DBPedia ontology[2] that contains the relations between named entities in Wikipedia, could improve the learning performance. As the relational representation is not limited to linguistic structure, one could even go one step further, e.g., by linking named entities of chemical compounds in a sentence to their molecular structure.

Furthermore, with the introduction of continuous bag-of-words and skip-gram architectures for computing vector representations of words (Mikolov et al., 2013), the inclusion of continuous features is a promising direction for future work. This word vector representation can capture many linguistic regularities at once, due to which it forms the basis for an improved similarity metric between individual words. Integrating this continuous vector representation into the relational model and the kernel calculation could lead to an improved performance of the relational learner.

Machine reading The probabilistic rule learning as presented in Chapter 8 offers a number of new opportunities in the context of machine reading. In order for a probabilistic rule learner to learn qualitative rules, it needs ground truth facts with a well-calibrated probability attached. Currently, the confidence scores attached to the facts in NELL are a combination of the probability output by the learning algorithm and a manual evaluation. Recent work on improving the quality of the estimated accuracy in NELL was performed by Platanios et al. (2014), who consider combining multiple approximations in order to estimate the true accuracy. To improve upon this, additional research into the field of information extraction is required.

We validated our approach on a subset of the NELL knowledge base. Integrating this approach into the pipeline of an online machine reading algorithm poses new challenges concerning scalability, and a tradeoff between the accuracy of the predictions versus the performance of the inference algorithm. On a related note, probabilistic rule learning also offers new opportunities for learning probabilistic ontologies.

[1] http://www.ihtsdo.org/snomed-ct/
[2] http://wiki.dbpedia.org/Ontology

Appendices

Appendix A

Example Abstract NICTA-PIBOSO Corpus

Abstract ID	Sent. nb.	Label	Sentence
10070506	1	other	BACKGROUND
10070506	2	background	Subfoveal choroidal neovascular membranes (CNV) are a cause of significant visual impairment.
10070506	3	background	Laser treatment of such lesions results in visual loss.
10070506	4	background	Surgical excision of CNV may allow stabilisation or improvement of vision.
10070506	5	background	A series of results of surgical excision are presented.
10070506	6	other	METHODS
10070506	7	intervention	The records for 43 eyes of 40 consecutive patients undergoing surgical excision of CNV not associated with age-related macular degeneration (AMD) were reviewed retrospectively.
10070506	7	population	The records for 43 eyes of 40 consecutive patients undergoing surgical excision of CNV not associated with age-related macular degeneration (AMD) were reviewed retrospectively.
10070506	8	other	Statistical analyses of the relationship between pre-operative factors and post-operative visual results were made.
10070506	9	other	Improvement or worsening of visual acuity was defined as a change of more than 2 lines of Snellen acuity.
10070506	10	other	RESULTS
10070506	11	outcome	In 79.1% of patients visual acuity was improved or unchanged following surgery, and in 20.9% there was a reduction of Snellen acuity.
10070506	12	outcome	There was no statistically significant association between visual outcome and age, gender, duration of visual symptoms, cause of CNV, presence of subretinal haemorrhage, elevation of retina by subretinal fluid, prior laser surgery, or the presence of pre-operative or intraoperative subretinal haemorrhage.
10070506	13	outcome	There was a possible association between the non-use of gas tamponade and an increased chance of reduced vision.
10070506	14	outcome	Visual loss was more likely in those eyes with good pre-operative visual acuity.
10070506	15	outcome	Recurrence of CNV was noted in 10 (23%) eyes; repeat surgery was not associated with a worse visual outcome.
10070506	16	other	CONCLUSIONS
10070506	17	outcome	Surgical excision of CNV not related to AMD is a promising technique.
10070506	18	outcome	More meaningful assessment of visual function in these patients will allow refinement of case selection.

Table A.1: Example of an annotated (structured) abstract from the NICTA-PIBOSO corpus

Appendix B

Other Work

Some of the work done during the course of the Ph.D. was left out of the thesis text in order to make the story as coherent as possible. In this appendix, a brief overview of this work will be given. All methods and applications presented here also concern text analysis. However, whereas the previous chapters primarily focus on machine learning methods for natural language processing, the following techniques are rooted in *data mining*, more specifically *text mining*, i.e., the computational process of discovering patterns in large amounts of data and text respectively.

B.1 Structure Recommendation

Search is one of the main applications on today's Web and other repositories, and it is supported by more and more advanced techniques. However, these techniques largely ignore an important component of search: the further processing of search-result sets that humans invariably undertake when dealing seriously with a list of documents provided by a search engine – and the diversity in which this is done by different people. An important form of such further processing is the grouping of result sets into subsets, which can be seen as investing an undifferentiated list of results with "semantics": a structuring into sets of documents each instantiating a concept (which is a subconcept of the overall query). On the Web, several search engines provide an automatic clustering of search results. However, regardless of how good the employed clustering algorithm is, a "globally optimal" clustering solution is generally impossible: There is no single best way of structuring a list of items; instead, the "optimal" grouping will depend on the context, tasks, previous knowledge,

etc. of the searcher(s).

On a truly Social Web, users should be empowered to see and work with a diversity of possible groupings and their associated sense-makings. A prerequisite for such diversity-aware tools is that users are able to perform individual sense-making, structuring search result sets according to their needs and knowledge. In (Verbeke et al., 2009), we studied how to make users aware of alternative ways to group a set of documents. To this end, we developed a tool to cluster documents resulting from a query. In this tool, *Damilicious* (DAta MIning in LIterature Search engines), the user can, starting from an automatically generated clustering, group a search result document set into meaningful groups, and she can learn about alternative groupings determined by other users. To *transfer* a clustering of one set of documents to another set of documents, the tool learns a *model* for this clustering, which can be applied to cluster alternative sets of documents. We refer to this model as the clustering's *intension*, as opposed to its *extension*, which is the original, unannotated grouping of the documents. This approach supports various measures of diversity, which measure to what extent two groupings differ. Such measures can be used to make recommendations and present new, possibly interesting viewpoints of structuring the result set. The domain of literature search was chosen to ensure a high level of user interest in reflection and meaningful results, although the methods can easily be transferred to other domains.

A good understanding of the dynamics of these structuring activities is a key prerequisite for theories of individual behaviour in collaborative settings, and it is necessary to improve the design of current and next generation (social) recommender systems. Furthermore, it can leverage the design of mechanisms that rely on implicit user interactions such as social search. To this end, we developed a dynamic conceptual clustering technique that simulates this intellectual structuring process, and which is able to identify these structuring dynamics (Verbeke et al., 2012, 2014b). The evolution of the grouping of an individual user is influenced by dynamically changing and collectively determined "guiding grouping(s)", which we will refer to as *guides*. We investigated two types of guides. The first one is motivated by a "narrow" view of the structurer's prior grouping behaviour and experience, while the second one starts from a "wider" perspective that also takes peer experience into account in grouping decisions.

The process in which we want to transfer the grouping of one set of items to another set of items during the guided grouping process is a combination of two data mining tasks. The first task starts with an extension of a structuring, and learns its intension as a classification model. The second task is to use the intensions of the peer groupings and apply their classifiers for prediction, to structure a new item. It starts from defining the k nearest peer groupings for a

user. To decide on the k nearest peer groupings in a situation where peers group different items, we defined a novel measure of divergence between groupings that may have a different number and identities of items. Once we obtain the k nearest peers, the task is to decide on item-to-group assignment using the peers' groupings. Based on the presence of an item in peer groupings, this decision is based either on the extension of the peer grouping (when a peer already grouped the item) or on its intension (when a peer has not grouped the item yet). By comparing the possible end groupings with the actual end grouping, we see which guide selection is more likely to have determined the actual process. The method can be used to identify structuring dynamics, which simulates an intellectual structuring process operating over an extended period of time.

Besides structuring, involving the user into the sense-making process can also be useful in the context of summarization. In (Berendt et al., 2013), we provided an overview of interactive methods and interfaces for multi-document summarization of news articles. We combined a number of these summarization formats into a ubiquitous learning setting, in which the user interacts with textual content, visualizations and summarization tools in different and complementing ways (Chongtay et al., 2013).

B.2 Question Answering for Machine Reading Evaluation

Question answering for machine reading evaluation (QA4MRE) aims at evaluating the performance of machine reading systems through question answering tests. This formed the topic of a series of shared tasks (Peñas et al., 2013). Given a single document on a particular topic, the goal of these tasks is to identify the answers to a set of multiple-choice questions about information that appears explicitly or implicitly in the text. The systems can also make use of a collection of documents that contain background information on the topic and could be useful in acquiring additional knowledge to bridge the gap between the text, the questions and the answers.

For the 2012 edition of the task (Peñas et al., 2012), we developed a text mining approach (Verbeke and Davis, 2012) for answering questions on biomedical texts about the Alzheimer disease (Morante et al., 2012). Question answering in the scientific domain poses additional challenges for machine reading, because domain knowledge is essential to achieve a deep understanding in this context. Consider the following example, taken from the corpus:

Example B.1. Text: [...] *Additionally, no estrogen could be detected in the APP23/Ar / mice (data not shown), suggesting that aromatase gene knock-out prevented the conversion of endogenous testosterone into estrogen.* [...]

Question: *What experimental approach is useful to create an in vivo system where conversion of testosterone into estrogen is blocked?*

Candidate Answers:

1. *ELISA analysis*
2. *hole-board memory task*
3. *NEP activity assay*
4. *Western blot*
5. *knock-out of the aromatase gene*

In order to identify the correct answer, namely *knock-out of the aromatase gene*, the fact that an experiment with mice is an in vivo system is essential information for a machine reading system. The main purpose of our research was to investigate how far-reaching the influence of this background knowledge is. To this end, we developed a system using basic text mining techniques, without considering any external resources.

We proposed two approaches, one based on question similarity and another based on answer containment. The former approach computes the similarity between each question in the reading test and every sentence in the input document and selects the top k most similar sentences. Subsequently, for each of these sentences, the system checks if the sentence contains (part of) an answer. If it does, the respective answer's vote is incremented by either 1 or the normalized similarity value. The answer with the highest number of votes, i.e., the highest weight, is selected. The latter approach reverses the procedure of the question similarity approach, and first checks if the answer appears in the sentence. If it does, the similarity between the sentence and the question is calculated. These similarities are summed and normalized for each of the sentences' respective answers. The answer with the highest similarity is selected.

The system can be seen as a baseline for the task. Based on its outcome, we investigated which types of questions can be answered based solely on the input text and the question string, and for which ones more advanced techniques are needed that also consider the previously acquired background knowledge from the reference document collection. As the fraction of questions of the former type was relatively large, the best submitted run of our system was ranked third out of seven systems participating in the competition. Despite this rank, the results still leave a large margin for improvement, for which considering the background knowledge from the reference document collection is the way forward.

For QA4MRE 2013 (Sutcliffe et al., 2013), we extended this system with additional preprocessing steps and coreference resolution, and explored its

use for question answering on more general (non-scientific) topics (de Oliveira Costa Marques and Verbeke, 2013). The best run of the extended system was ranked second out of 77 runs from 19 participants, and all submitted runs performed above average. When compared to the previous results for question answering on biomedical texts, a slight improvement could be observed. This can be explained by the integration of the additional preprocessing steps and coreference resolution. Due to the generality of the topics, also the lack of background knowledge from the reference document collection exercised a smaller influence on the performance.

B.3 Data Mining for Communication Science

The big-data practices of social media are transforming the way many people interact with one another, show public engagement, and receive and produce information. The endless media options and processes that help form public opinion are reshaped. News media that have traditionally played a key role in the formation of public opinion now share this role with other sources of information. In the fragmentized news media climate, many news media are now themselves actors in social media, establishing their own practices as well as developing practices of interaction and engagement with non-traditional sources such as citizens who can opt to become creators or distributors of news as "citizen journalists".

In (Verbeke et al., 2014a), we investigated how the traditional activities of news production, publication and dissemination unfold in this new ecosystem, and how the different practices interact to shape public opinion. We focused on the microblogging system Twitter and on Twitter messages (tweets) relating to mainstream media. We did not investigate the contents of messages or of public opinion, but focused on the communication structure as such, in particular on who is active in distributing content and/or enriching content. We regard this as a proxy measure of how people read and process news, to what extent they engage in critical news reading, and thus ultimately of how they form and shape opinions.

Methodologically, we investigated how the big-data data that are generated by the big-data practices on social media, can give us insights about those practices, when analyzed through the lens of data mining, as a big-data research method. Furthermore, we used this analysis to demonstrate how questions that arise in this process of interdisciplinary collaboration challenge and enrich the assumptions, questions and interpretations of both disciplines that were involved in this study: computer science and communication science.

To this end, we presented a model of news reading on Twitter, and

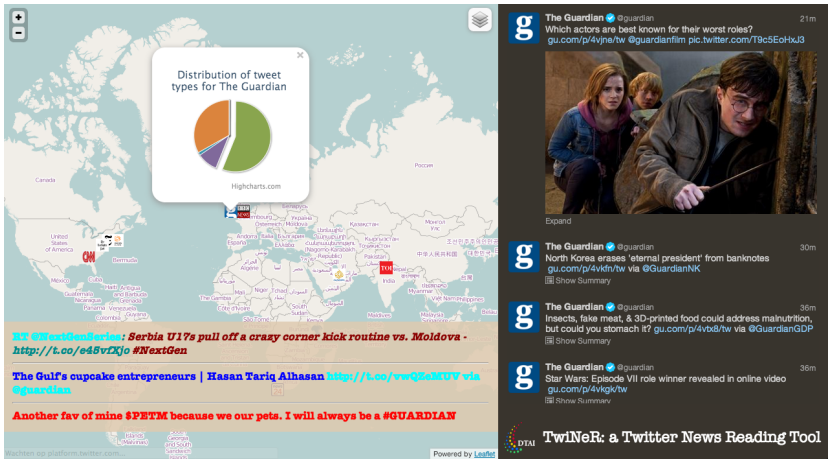


Figure B.1: Screenshot of TwiNeR, illustrating the distribution of tweet types for The Guardian (center left) and analysed tweets (bottom left).

TwiNeR (Twitter News Reading), an interactive tool which allows analysts to automatically gather and analyse tweets in relation to the model of news reading, as well as to read these tweets in detail (see Figure B.1). We demonstrated the usefulness of this tool for answering our research questions by reporting on a study of 18 news media from across all continents.

The results of our study suggest that social-media practices have evolved towards both mainstream news media and "other" users engaging in the distribution, reading and discussion of content originally produced by the media's journalists. In this sense, we observe both a democratization and a still-important role of mainstream media in shaping public opinion. However, when looking more closely, we also observed that discussion often stops after the first comment has been made - after that, tweets tend to be simply spread to further audiences. This indicates that Twitter is seldom used as a forum for critically discussing mainstream news content, and it hints at the possibility that some Twitter users may become the new gatekeepers of the news universe.

Bibliography

Pieter Adriaans and Erik de Haas. Grammar induction as substructural inductive logic programming. In James Cussens and Sašo Džeroski, editors, *Learning Language in Logic*, volume 1925 of *Lecture Notes in Computer Science*, pages 127–142. Springer Berlin Heidelberg, 2000. ISBN 978-3-540-41145-1. doi: 10.1007/3-540-40030-3_8. URL http://dx.doi.org/10.1007/3-540-40030-3_8.

Shashank Agarwal and Hong Yu. Automatically classifying sentences in full-text biomedical articles into introduction, methods, results and discussion. *Bioinformatics*, 25(23):3174–3180, December 2009. ISSN 1367-4803. doi: 10.1093/bioinformatics/btp548. URL http://dx.doi.org/10.1093/bioinformatics/btp548.

David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991. ISSN 0885-6125. doi: 10.1007/BF00153759. URL http://dx.doi.org/10.1007/BF00153759.

James S. Aitken. Learning information extraction rules: An inductive logic programming approach. In *Proceedings of the 15th European Conference on Artificial Intelligence*, pages 355–359, 2002.

Zoltán Alexin, Péter Leipold, János Csirik, Károly Bibok, and Tibor Gyimóthy. A rule-based tagger development framework. In Luboš Popelínský and Miloslav Nepil, editors, *Learning Language in Logic*, pages 1–10, Strasbourg, France, September 2001.

James Allen. *Natural Language Understanding*. Benjamin-Cummings Publishing Co., Inc., Redwood City, CA, USA, 1988. ISBN 0-8053-0330-8.

Érick Alphonse and Stan Matwin. Feature subset selection and inductive logic programming. In Claude Sammut and Achim G. Hoffmann, editors, *ICML*, pages 11–18. Morgan Kaufmann, 2002. ISBN 1-55860-873-7.

Yasemin Altun, Ioannis Tsochantaridis, and Thomas Hofmann. Hidden Markov support vector machines. In Tom Fawcett and Nina Mishra, editors, *ICML*, pages 3–10. AAAI Press, 2003. ISBN 1-57735-189-4.

Edoardo Amaldi and Viggo Kann. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209(1-2):237–260, 1998.

Iman Amini, David Martinez, and Diego Molla. Overview of the ALTA 2012 shared task. In *Proceedings of the Australasian Language Technology Association Workshop 2012*, pages 124–129, 2012. URL http://aclweb.org/anthology/U12-1017.

Laura Antanas, Paolo Frasconi, Fabrizio Costa, Tinne Tuytelaars, and Luc De Raedt. A relational kernel-based framework for hierarchical image understanding. In Georgy Gimel'farb, Edwin Hancock, Atsushi Imiya, Arjan Kuijper, Mineichi Kudo, Shinichiro Omachi, Terry Windeatt, and Keiji Yamada, editors, *Structural, Syntactic, and Statistical Pattern Recognition*, volume 7626 of *Lecture Notes in Computer Science*, pages 171–180. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-34165-6. doi: 10.1007/978-3-642-34166-3_19. URL http://dx.doi.ossssrg/10.1007/978-3-642-34166-3_19.

Laura Antanas, McElory Hoffmann, Paolo Frasconi, Tinne Tuytelaars, and Luc De Raedt. A relational kernel-based approach to scene classification. *IEEE Workshop on Applications of Computer Vision*, 0:133–139, 2013. ISSN 1550-5790. doi: http://doi.ieeecomputersociety.org/10.1109/WACV.2013.6475010.

Eva Armengol and Enric Plaza. Similarity assessment for relational CBR. In *Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR '01, pages 44–58, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-42358-3. URL http://dl.acm.org/citation.cfm?id=646268.684000.

Eva Armengol and Enric Plaza. Relational case-based reasoning for carcinogenic activity prediction. *Artificial Intelligence Review*, 20(1-2):121–141, 2003. ISSN 0269-2821. doi: 10.1023/A:1026076312419.

E. C. Armstrong. The well-built clinical question: the key to finding the best evidence efficiently. *WMJ*, 98(2):25–28, 1999. URL http://www.ncbi.nlm.nih.gov/pubmed/10235058.

Elena Bellodi and Fabrizio Riguzzi. Learning the structure of probabilistic logic programs. In *Inductive Logic Programming*, pages 61–75. Springer, 2012.

Islam Beltagy, Katrin Erk, and Raymond Mooney. Probabilistic soft logic for semantic textual similarity. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1210–1219, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P14/P14-1114.

Bettina Berendt, Mark Last, Ilija Subasic, and Mathias Verbeke. New formats and interfaces for multi-document news summarization and its evaluation. In Alessandro Fiori, editor, *Innovative Document Summarization Techniques: Revolutionizing Knowledge Understanding*. IGI Global, 2013. doi: 10.4018/978-1-4666-5019-0. URL https://lirias.kuleuven.be/handle/123456789/423917.

Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1st edition, 2009. ISBN 0596516495, 9780596516499.

Hendrik Blockeel and Luc De Raedt. Top-down induction of first-order logical decision trees. *Artificial Intelligence*, 101(1-2):285–297, May 1998. ISSN 0004-3702. doi: 10.1016/S0004-3702(98)00034-4. URL http://dx.doi.org/10.1016/S0004-3702(98)00034-4.

Olivier Bodenreider. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32(Database-Issue):267–270, 2004.

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM. ISBN 0-89791-497-X. doi: 10.1145/130385.130401. URL http://doi.acm.org/10.1145/130385.130401.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010)*, pages 177–187, Paris, France, August 2010. Springer. URL http://leon.bottou.org/papers/bottou-2010.

Sabine Buchholz and Erwin Marsi. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 149–164, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1596276.1596305.

Razvan Bunescu and Raymond J. Mooney. Collective information extraction with relational Markov networks. In *Proceedings of the 42nd Annual Meeting*

on Association for Computational Linguistics, ACL '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. doi: 10.3115/1218955.1219011. URL http://dx.doi.org/10.3115/1218955.1219011.

Razvan Bunescu and Raymond J. Mooney. Statistical relational learning for natural language information extraction. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*, pages 535–552. MIT Press, Cambridge, MA, 2007. URL http://www.cs.utexas.edu/users/ai-lab/?bunescu:bkchapter07b.

Mary Elaine Califf and Raymond J. Mooney. Bottom-up relational learning of pattern matching rules for information extraction. *Journal of Machine Learning Research*, 4:177–210, 2003.

Peter Carbonetto, Jacek Kisynski, Nando de Freitas, and David Poole. Nonparametric Bayesian logic. In *UAI*, pages 85–93. AUAI Press, 2005. ISBN 0-9749039-1-4.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. Toward an architecture for never-ending language learning. In *In AAAI*, 2010.

Radomír Černoch and Filip Železný. Probabilistic rule learning through integer linear programming. In Radim Jiroušek and Vojtěch Svátek, editors, *Sborník příspěvků 10. ročníku konference ZNALOSTI*, 2011.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3): 27:1–27:27, May 2011. ISSN 2157-6904. doi: 10.1145/1961189.1961199. URL http://doi.acm.org/10.1145/1961189.1961199.

NiteshV. Chawla. Data mining for imbalanced datasets: An overview. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 853–867. Springer US, 2005. ISBN 978-0-387-24435-8. doi: 10.1007/0-387-25465-X_40. URL http://dx.doi.org/10.1007/0-387-25465-X_40.

Wanxiang Che and Ting Liu. Jointly modeling WSD and SRL with Markov logic. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 161–169, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1873781.1873800.

Lin Chen and Barbara Di Eugenio. A Lucene and maximum entropy model based hedge detection system. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 114–119, Uppsala, Sweden,

July 2010. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W10-3016.

Peter Pin-Shan Chen. The entity-relationship model - toward a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, March 1976. ISSN 0362-5915. doi: 10.1145/320434.320440. URL http://doi.acm.org/10.1145/320434.320440.

Noam Chomsky. *Syntactic Structures*. Mouton, The Hague, 1957.

Noam Chomsky. A review of B.F. Skinner's Verbal Behavior. *Language*, 35(1): 26–58, 1959.

Rocio Chongtay, Mark Last, Mathias Verbeke, and Bettina Berendt. Summarize to learn: Summarization and visualization of text for ubiquitous learning. In *Proceedings of the 3rd IEEE Workshop on Interactive Visual Text Analytics at VIS 2013, The 3rd IEEE Workshop on Interactive Visual Text Analytics, Atlanta, GA, 14 October 2013*, pages 1–4, 2013. URL https://lirias.kuleuven.be/handle/123456789/423905.

Chun Kit Chui, Ben Kao, and Edward Hung. Mining frequent itemsets from uncertain data. In Zhi-Hua Zhou, Hang Li, and Qiang Yang, editors, *PAKDD*, volume 4426 of *Lecture Notes in Computer Science*, pages 47–58. Springer, 2007. ISBN 978-3-540-71700-3.

Grace Chung. Sentence retrieval for abstracts of randomized controlled trials. *BMC Medical Informatics and Decision Making*, 9(1):10, 2009. ISSN 1472-6947. doi: 10.1186/1472-6947-9-10. URL http://www.biomedcentral.com/1472-6947/9/10.

Philipp Cimiano, Helena Hartfiel, and Sebastian Rudolph. Intensional question answering using ILP: What does an answer mean? In Epaminondas Kapetanios, Vijayan Sugumaran, and Myra Spiliopoulou, editors, *Natural Language and Information Systems*, volume 5039 of *Lecture Notes in Computer Science*, pages 151–162. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-69857-9. doi: 10.1007/978-3-540-69858-6_16. URL http://dx.doi.org/10.1007/978-3-540-69858-6_16.

Vincent Claveau, Pascale Sébillot, Cécile Fabre, and Pierrette Bouillon. Learning semantic lexicons from a part-of-speech and semantically tagged corpus using inductive logic programming. *Journal of Machine Learning Research*, 4:493–525, December 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=945365.945393.

William W Cohen. Learning to classify English text with ILP methods. *Advances in inductive logic programming*, 32:124–143, 1995.

Michael Collins and Nigel Duffy. Convolution kernels for natural language. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 625–632. MIT Press, 2002. URL http://papers.nips.cc/paper/2089-convolution-kernels-for-natural-language.pdf.

Alain Colmerauer. Metamorphosis grammars. In Leonard Bolc, editor, *Natural Language Communication with Computers*, volume 63 of *Lecture Notes in Computer Science*, pages 133–189. Springer, 1978. ISBN 3-540-08911-X.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995. ISSN 0885-6125. doi: 10.1023/A:1022627411411. URL http://dx.doi.org/10.1023/A:1022627411411.

Scott Cost and Steven Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10(1):57–78, January 1993. ISSN 0885-6125. doi: 10.1023/A:1022664626993. URL http://dx.doi.org/10.1023/A:1022664626993.

Fabrizio Costa and Kurt De Grave. Fast neighborhood subgraph pairwise distance kernel. In Johannes Fürnkranz and Thorsten Joachims, editors, *ICML*, pages 255–262. Omnipress, 2010. ISBN 978-1-60558-907-7.

Fabrizio Costa, Sauro Menchetti, Alessio Ceroni, Andrea Passerini, and Paolo Frasconi. *Proceedings of the Workshop on Learning Structured Information in Natural Language Applications*, chapter Decomposition Kernels for Natural Language Processing. Association for Computational Linguistics, 2006. URL http://aclweb.org/anthology/W06-2603.

Michael A. Covington. *Natural Language Processing for Prolog Programmers*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 1993. ISBN 0136292135.

Aron Culotta, Michael Wick, and Andrew McCallum. First-order probabilistic models for coreference resolution. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 81–88, Rochester, New York, April 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N07-1011.

Chad M. Cumby and Dan Roth. On kernel methods for relational learning. In Tom Fawcett and Nina Mishra, editors, *ICML*, pages 107–114. AAAI Press, 2003. ISBN 1-57735-189-4.

James Cussens. Part-of-speech tagging using Progol. In Sašo Džeroski and Nada Lavrač, editors, *Proceedings of the 7th International Workshop on Inductive*

Logic Programming, volume 1297 of *LNAI*, pages 93–108. Springer Berlin Heidelberg, 1997.

James Cussens. Loglinear models for first-order probabilistic reasoning. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, UAI'99, pages 126–133, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1-55860-614-9. URL http://dl.acm.org/citation.cfm?id=2073796.2073811.

James Cussens and Sašo Džeroski, editors. *Learning Language in Logic*, volume 1925 of *Lecture Notes in Computer Science*, 2000. Springer. ISBN 3-540-41145-3.

James Cussens and Stephen G. Pulman. Experiments in inductive chart parsing. In James Cussens and Sašo Džeroski, editors, *Learning Language in Logic*, volume 1925 of *LNCS*, pages 143–156. Springer Berlin Heidelberg, June 2000. ISBN 3-540-41145-3. URL http://www.springer.de/cgi-bin/search_book.pl?isbn=3-540-41145-3.

James Cussens, David Page, Stephen Muggleton, and Ashwin Srinivasan. Using inductive logic programming for natural language processing. In Walter Daelemans, Ton Weijters, and Antal van den Bosch, editors, *ECML97 Workshop notes on empirical learning of natural language processing tasks*, pages 25–34, Prague, 1997. Laboratory of Intelligent Systems. URL http://www.cnts.ua.ac.be/papers/1997/dwb97.pdf.

Walter Daelemans and Antal van den Bosch. *Memory-Based Language Processing*. Cambridge University Press, New York, NY, USA, 1st edition, 2009. ISBN 0521114454, 9780521114455.

Walter Daelemans, Jakub Zavrel, Antal van den Bosch, and Ko van der Sloot. MBT: Memory based tagger, version 3.2, reference guide. Technical Report ILK Research Group and CLiPs Technical Report Series 10-04, ILK (Tilburg University), CLiPs (Antwerp University), 2010.

Verónica Dahl. Natural language processing and logic programming. *Journal of Logic Programming*, 19/20:681–714, 1994.

Hal Daumé, III and Daniel Marcu. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pages 169–176, New York, NY, USA, 2005. ACM. ISBN 1-59593-180-5. doi: 10.1145/1102351.1102373. URL http://doi.acm.org/10.1145/1102351.1102373.

P. Davis-Desmond and D. Mollá. Detection of evidence in clinical research papers. In K. Butler-Henderson and K. Gray, editors, *Australasian Workshop*

on Health Informatics and Knowledge Management (HIKM 2012), volume 129 of *CRPIT*, pages 13–20, Melbourne, Australia, 2012. ACS. URL http://crpit.com/confpapers/CRPITV129Davis-Desmond.pdf.

Guilerme de Oliveira Costa Marques and Mathias Verbeke. Combining text mining techniques for QA4MRE 2013. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Valencia, Spain, September 23-26, 2013*, September 2013. URL https://lirias.kuleuven.be/handle/123456789/416992. Conference and Labs of the Evaluation Forum (CLEF), Valencia, Spain, 23-26 September 2013.

Luc De Raedt. Logical settings for concept-learning. *Artificial Intelligence*, 95(1):187 – 201, 1997. ISSN 0004-3702. doi: http://dx.doi.org/10.1016/S0004-3702(97)00041-6. URL http://www.sciencedirect.com/science/article/pii/S0004370297000416.

Luc De Raedt. *Logical and relational learning*. Cognitive Technologies. Springer, 2008. ISBN 978-3-540-20040-6.

Luc De Raedt and Sašo Džeroski. First-order jk-clausal theories are PAC-learnable. *Artificial Intelligence*, 70(1-2):375–392, October 1994. doi: 10.1016/0004-3702(94)90112-0. URL https://lirias.kuleuven.be/handle/123456789/124997.

Luc De Raedt and Kristian Kersting. Probabilistic logic learning. *SIGKDD Explorations*, 5(1):31–48, 2003.

Luc De Raedt and Kristian Kersting. Probabilistic inductive logic programming. In Shai Ben-David, John Case, and Akira Maruoka, editors, *ALT*, volume 3244 of *Lecture Notes in Computer Science*, pages 19–36. Springer, 2004. ISBN 3-540-23356-3.

Luc De Raedt and Angelika Kimmig. Probabilistic programming concepts. *CoRR*, abs/1312.4328, 2013.

Luc De Raedt and Ingo Thon. Probabilistic rule learning. In Frasconi Paolo and Francesa Alessandra Lisi, editors, *Lecture Notes in Computer Science, Inductive Logic Programming - 20th International Conference, ILP 2010,*, pages 47–58, 2010. URL https://lirias.kuleuven.be/handle/123456789/296011.

Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. Problog: A probabilistic Prolog and its application in link discovery. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2468–2473, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1625275.1625673.

Luc De Raedt, Paolo Frasconi, Kristian Kersting, and Stephen Muggleton, editors. *Probabilistic Inductive Logic Programming - Theory and Applications*, volume 4911 of *Lecture Notes in Computer Science*, 2008. Springer. ISBN 978-3-540-78651-1.

Luc Dehaspe, Hendrik Blockeel, and Luc De Raedt. Induction, logic and natural language processing. In *Proceedings of the Joint ELSNET/COMPULOG-NET/EAGLES Workshop on Computational Logic for Natural Language Processing (CLNLP-95)*, 1995. URL https://lirias.kuleuven.be/handle/123456789/134526.

Dina Demner-Fushman and Jimmy Lin. Answering clinical questions with knowledge-based and statistical techniques. *Computational Linguistics*, 33(1): 63–103, March 2007. ISSN 0891-2017. doi: 10.1162/coli.2007.33.1.63. URL http://dx.doi.org/10.1162/coli.2007.33.1.63.

Dina Demner-Fushman, Barbara Few, Susan E. Hauser, and George R. Thoma. Automatically identifying health outcome information in MEDLINE records. *JAMIA*, 13(1):52–60, 2006.

Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, January 1997. ISSN 0004-3702. doi: 10.1016/S0004-3702(96)00034-3. URL http://dx.doi.org/10.1016/S0004-3702(96)00034-3.

Predro Domingos. What's missing in AI: The interface layer. In P. Cohen, editor, *Artificial Intelligence: The First Hundred Years*. AAAI Press, Menlo Park, CA, 2006.

Janardhan Rao Doppa, Mohammad NasrEsfahani, Mohammad S. Sorower, Thomas G. Dietterich, Xiaoli Fern, and Prasad Tadepalli. Towards learning rules from natural texts. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, FAM-LbR '10, pages 70–77, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1866775.1866784.

Janardhan Rao Doppa, Shahed Sorower, Mohammad NasrEsfahani, Walker Orr, Thomas G. Dietterich, Xiaoli Fern, Prasad Tadepalli, and Jed Irvine. Learning rules from incomplete examples via implicit mention models. In Chun-Nan Hsu and Wee Sun Lee, editors, *ACML*, volume 20 of *JMLR Proceedings*, pages 197–212. JMLR.org, 2011.

Leon E. Dostert. The Georgetown-IBM experiment. *Machine translation of languages*, pages 124–135, 1955.

Sahibsingh A. Dudani. The distance-weighted k-nearest-neighbor rule. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-6(4):325–327, 1976. ISSN 0018-9472. doi: 10.1109/TSMC.1976.5408784.

Sašo Džeroski. Handling imperfect data in inductive logic programming. In *Proceedings of the Fourth Scandinavian Conference on Artificial intelligence*, SCAI93, pages 111–125, Amsterdam, The Netherlands, The Netherlands, 1993. IOS Press. ISBN 90-5199-134-7. URL http://dl.acm.org/citation.cfm?id=168459.168478.

Sašo Džeroski and Tomaž Erjavec. Induction of Slovene nominal paradigms. In Nada Lavrač and Sašo Džeroski, editors, *Inductive Logic Programming*, volume 1297 of *Lecture Notes in Computer Science*, pages 141–148. Springer Berlin Heidelberg, 1997. ISBN 978-3-540-63514-7. doi: 10.1007/3540635149_42. URL http://dx.doi.org/10.1007/3540635149_42.

Sašo Džeroski and Tomaž Erjavec. Learning to lemmatise Slovene words. In James Cussens and Sašo Džeroski, editors, *Learning Language in Logic*, volume 1925 of *LNCS*, pages 69–88. Springer Berlin Heidelberg, June 2000. ISBN 3-540-41145-3. URL http://www.springer.de/cgi-bin/search_book.pl?isbn=3-540-41145-3.

Sašo Džeroski, James Cussens, and Suresh Manandhar. An introduction to inductive logic programming and learning language in logic. In Cussens and Džeroski (2000), pages 3–35. ISBN 3-540-41145-3.

Maximilian Dylla, Iris Miliaraki, and Martin Theobald. A temporal-probabilistic database model for information extraction. *Proceedings of the VLDB Endowment*, 6(14):1810–1821, 2013.

Martin Eineborg and Nikolaj Lindberg. ILP in tagging part-of-speech an overview. In James Cussens and Sašo Džeroski, editors, *Learning Language in Logic*, volume 1925 of *Lecture Notes in Computer Science*, pages 157–169. Springer Berlin Heidelberg, 2000. ISBN 978-3-540-41145-1. doi: 10.1007/3-540-40030-3_10. URL http://dx.doi.org/10.1007/3-540-40030-3_10.

Werner Emde and Dietrich Wettschereck. Relational instance-based learning. In Lorenza Saitta, editor, *Proceedings of the 13th International Conference on Machine Learning*, volume 96, pages 122–130, 1996.

F. Esposito, S. Ferilli, N. Fanizzi, and G. Semeraro. Learning from parsed sentences with INTHELEX. In *Proceedings of the 2Nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning - Volume 7*, ConLL '00, pages 194–198, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics. doi: 10.3115/1117601.1117648. URL http://dx.doi.org/10.3115/1117601.1117648.

Richárd Farkas, Veronika Vincze, György Móra, János Csirik, and György Szarvas. The CoNLL-2010 shared task: Learning to detect hedges and their scope in natural language text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning — Shared Task*, CoNLL '10: Shared Task, pages 1–12, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. ISBN 978-1-932432-84-8. URL http://dl.acm.org/citation.cfm?id=1870535.1870536.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the DeepQA project. *AI magazine*, 31(3):59–79, 2010.

Daan Fierens, Guy Van den Broeck, Joris Renkens, Dimitar Shterionov, Bernd Gutmann, Ingo Thon, Gerda Janssens, and Luc De Raedt. Inference and learning in probabilistic logic programs using weighted Boolean formulas. *Theory and Practice of Logic Programming (TPLP)*, FirstView, 2014.

Evelyn Fix and J.L. Hodges, Jr. Discriminatory analysis: Nonparametric discrimination: Consistency properties. Technical Report Project 21-49-004, Report Number 4, USAF School of Aviation Medicine, Randolf Field, TX, USA, 1951.

Peter Flach. *Machine Learning: The Art and Science of Algorithms That Make Sense of Data*. Cambridge University Press, New York, NY, USA, 2012. ISBN 1107422221, 9781107422223.

Paolo Frasconi, Andrea Passerini, Stephen Muggleton, and Huma Lodhi. Declarative kernels. In Stefan Kramer en Bernhard Pfahringer, editor, *Proceedings of the 15th International Conference on Inductive Logic Programming – Late-Breaking Papers*, pages 17–19, 2005.

Paolo Frasconi, Fabrizio Costa, Luc De Raedt, and Kurt De Grave. kLog: A language for logical and relational learning with kernels. *Artificial Intelligence*, 217:117–143, December 2014. doi: 10.1016/j.artint.2014.08.003. URL https://lirias.kuleuven.be/handle/123456789/460323.

Dayne Freitag. Toward general-purpose learning for information extraction. In *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1*, COLING '98, pages 404–408, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics. doi: 10.3115/980451.980914. URL http://dx.doi.org/10.3115/980451.980914.

Johannes Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, February 1999. ISSN 0269-2821. doi: 10.1023/A:1006524209794. URL http://dx.doi.org/10.1023/A:1006524209794.

Johannes Fürnkranz and Peter A. Flach. Roc 'n' rule learning—towards a better understanding of covering algorithms. *Machine Learning*, 58(1):39–77, 2005. ISSN 0885-6125. doi: http://dx.doi.org/10.1007/s10994-005-5011-x.

Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. AMIE: Association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, pages 413–422, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-2035-1. URL http://dl.acm.org/citation.cfm?id=2488388.2488425.

Viola Ganter and Michael Strube. Finding hedges by chasing weasels: Hedge detection using Wikipedia tags and shallow linguistic features. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 173–176, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1667583.1667636.

Thomas Gärtner. A survey of kernels for structured data. *SIGKDD Explor. Newsl.*, 5(1):49–58, July 2003. ISSN 1931-0145. doi: 10.1145/959242.959248. URL http://doi.acm.org/10.1145/959242.959248.

Spandana Gella and Thanh Duong Long. Automatic sentence classifier using sentence ordering features for event based medicine: Shared task system description. In *Proceedings of the Australasian Language Technology Association Workshop 2012*, pages 130–133, 2012. URL http://aclweb.org/anthology/U12-1018.

Maria Georgescul. A hedgehop over a max-margin framework using hedge cues. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning — Shared Task*, CoNLL '10: Shared Task, pages 26–31, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. ISBN 978-1-932432-84-8. URL http://dl.acm.org/citation.cfm?id=1870535.1870539.

Lise Getoor and Ben Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. MIT Press, 2007. ISBN 0262072882.

Mark Goadrich, Louis Oliphant, and Jude Shavlik. Learning ensembles of first-order clauses for recall-precision curves: A case study in biomedical information extraction. In *Proceedings of the Fourteenth International Conference on Inductive Logic Programming*, Porto, Portugal, 2004.

Mark Goadrich, Louis Oliphant, and Jude Shavlik. Learning to extract genic interactions using Gleaner. In James Cussens and Claire Nédellec, editors,

Proceedings of the 4th Learning Language in Logic Workshop (LLL05), pages 62–68, Bonn, August 2005.

Noah D. Goodman, Vikash K. Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: a language for generative models. In David A. McAllester and Petri Myllymäki, editors, *UAI*, pages 220–229. AUAI Press, 2008. ISBN 0-9749039-4-9.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, March 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=944919.944968.

Stevan Harnad. Other bodies, other minds: A machine incarnation of an old philosophical problem. *Minds and Machines*, 1(1):43–54, February 1991. ISSN 0924-6495. URL http://dl.acm.org/citation.cfm?id=103493.103530.

Hamed Hassanzadeh, Tudor Groza, and Jane Hunter. Identifying scientific artefacts in biomedical literature: The evidence based medicine use case. *Journal of Biomedical Informatics*, 49:159–170, 2014.

David Haussler. Convolution kernels on discrete structures. Technical Report UCS-CRL-99-10, University of California at Santa Cruz, Santa Cruz, CA, USA, 1999. URL http://citeseer.ist.psu.edu/haussler99convolution.html.

K. Hirohata, N. Okazaki, S. Ananiadou, and M. Ishizuka. Identifying sections in scientific abstracts using conditional random fields. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP 2008)*, pages 381–388, 2008.

Tamás Horváth, Stefan Wrobel, and Uta Bohnebeck. Relational instance-based learning with lists and terms. *Machine Learning*, 43(1-2):53–80, April 2001. ISSN 0885-6125. doi: 10.1023/A:1007668716498. URL http://dx.doi.org/10.1023/A:1007668716498.

Yufang Hou, Katja Markert, and Michael Strube. Global inference for bridging anaphora resolution. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *HLT-NAACL*, pages 907–917. The Association for Computational Linguistics, 2013.

Tuyen N. Huynh and Raymond J. Mooney. Discriminative structure and parameter learning for Markov logic networks. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *ICML*, volume 307 of *ACM International Conference Proceeding Series*, pages 416–423. ACM, 2008. ISBN 978-1-60558-205-4.

Tuyen N. Huynh and Raymond J. Mooney. Max-margin weight learning for Markov logic networks. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), Part 1*, pages 564–579, Bled, Slovenia, September 2009. URL http://www.cs.utexas.edu/users/ai-lab/?huynh:ecml-pkdd09.

Ken Hyland. *Hedging in Scientific Research Articles*. Pragmatics & Beyond: New Series. John Benjamins Publishing Co., P.O. Box 75577, 1070 AN Amsterdam, The Netherlands, 1998.

Laurent Jacob, Guillaume Obozinski, and Jean-Philippe Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 433–440, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553431. URL http://doi.acm.org/10.1145/1553374.1553431.

F. Jelinek. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4):532–556, April 1976. ISSN 0018-9219. doi: 10.1109/PROC.1976.10159.

Frederick Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA, USA, 1997. ISBN 0-262-10066-5.

Rodolphe Jenatton, Jean-Yves Audibert, and Francis Bach. Structured variable selection with sparsity-inducing norms. *Journal of Machine Learning Research*, 12:2777–2824, November 2011. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1953048.2078194.

Rodolphe Jenatton, Nicolas L. Roux, Antoine Bordes, and Guillaume R. Obozinski. A latent factor model for highly multi-relational data. In P. Bartlett, F.c.n. Pereira, C.j.c. Burges, L. Bottou, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 3176–3184. Advances in Neural Information Processing Systems, 2012. URL http://books.nips.cc/papers/files/nips25/NIPS2012_1455.pdf.

David Jensen and Jennifer Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In Claude Sammut and Achim G. Hoffmann, editors, *ICML*, pages 259–266. Morgan Kaufmann, 2002. ISBN 1-55860-873-7.

Shangpu Jiang, Daniel Lowd, and Dejing Dou. Learning to refine an automatically extracted knowledge base using Markov logic. In *Proceedings of the 12th IEEE International Conference on Data Mining (ICDM)*, pages 912–917, 2012.

Alípio Jorge and Alneu de Andrade Lopes. Iterative part-of-speech tagging. In James Cussens and Sašo Džeroski, editors, *Learning Language in Logic*, volume 1925 of *Lecture Notes in Computer Science*, pages 170–183. Springer Berlin Heidelberg, 2000. ISBN 978-3-540-41145-1. doi: 10.1007/3-540-40030-3_11. URL http://dx.doi.org/10.1007/3-540-40030-3_11.

Markus Junker, Michael Sintek, and Matthias Rinck. Learning for text categorization and information extraction with ILP. In James Cussens, editor, *Learning Language in Logic*, pages 84–93, Bled, Slovenia, June 1999. URL http://www.cs.york.ac.uk/mlg/lll/workshop/proceedings/Junkeretal/Junkeretal.ps.gz.

Dimitar Kazakov. Achievements and prospects of learning word morphology with inductive logic programming. In James Cussens and Sašo Džeroski, editors, *Learning Language in Logic*, volume 1925 of *LNCS*, pages 89–109. Springer Berlin Heidelberg, June 2000. ISBN 3-540-41145-3. URL http://www.springer.de/cgi-bin/search_book.pl?isbn=3-540-41145-3.

Jeroen Kazius, Ross McGuire, and Roberta Bursi. Derivation and validation of toxicophores for mutagenicity prediction. *Journal of Medicinal Chemistry*, 48(1):312–320, 2005. doi: 10.1021/jm040835a. URL http://pubs.acs.org/doi/abs/10.1021/jm040835a.

Michael J. Kearns and Robert E. Schapire. Efficient distribution-free learning of probabilistic concepts. *Journal of Computer and System Sciences*, 48(3): 464–497, 1994.

Kristian Kersting and Luc De Raedt. Bayesian logic programming: Theory and tool. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*, chapter 10. MIT Press, 2007.

Halil Kilicoglu and Sabine Bergler. Recognizing speculative language in biomedical research articles: a linguistically motivated perspective. *BMC Bioinformatics*, 9(Suppl 11):S10, 2008. ISSN 1471-2105. doi: 10.1186/1471-2105-9-S11-S10. URL http://www.biomedcentral.com/1471-2105/9/S11/S10.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, BioNLP '09, pages 1–9, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-44-2. URL http://dl.acm.org/citation.cfm?id=1572340.1572342.

Su Kim, David Martinez, Lawrence Cavedon, and Lars Yencken. Automatic classification of sentences to support evidence based medicine. *BMC*

Bioinformatics, 12(Suppl 2):S5, 2011. ISSN 1471-2105. doi: 10.1186/1471-2105-12-S2-S5. URL http://www.biomedcentral.com/1471-2105/12/S2/S5.

A. Kimmig, V. Santos Costa, R. Rocha, V. Demoen, and L. De Raedt. On the efficient execution of problog programs. In *ICLP*, pages 175–189, 2008.

Angelika Kimmig and Luc De Raedt. Local query mining in a probabilistic Prolog. In Craig Boutilier, editor, *IJCAI*, pages 1095–1100, 2009.

Angelika Kimmig, Stephen H. Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. A short introduction to probabilistic soft logic. In *NIPS Workshop on Probabilistic Programming: Foundations and Applications*, 2012.

Donald E. Knuth. On the translation of languages from left to right. *Information and Control*, 8(6):607–639, 1965.

Stanley Kok and Pedro Domingos. Learning the structure of Markov logic networks. In *Proceedings of the 22nd international conference on Machine learning*, pages 441–448. ACM, 2005.

Stanley Kok and Pedro Domingos. Statistical predicate invention. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 433–440, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273551. URL http://doi.acm.org/10.1145/1273496.1273551.

Stanley Kok and Pedro Domingos. Extracting semantic networks from text via relational clustering. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I*, ECML PKDD '08, pages 624–639, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-87478-2. doi: 10.1007/978-3-540-87479-9_59. URL http://dx.doi.org/10.1007/978-3-540-87479-9_59.

Janet L. Kolodner. An introduction to case-based reasoning. *Artificial Intelligence Review*, 6(1):3–34, 1992. ISSN 0269-2821. doi: 10.1007/BF00155578. URL http://dx.doi.org/10.1007/BF00155578.

Stasinos Konstantopoulos. NP chunking using ILP. In Tanja Gaustad, editor, *CLIN*, volume 47 of *Language and Computers - Studies in Practical Linguistics*, pages 77–91. Rodopi, 2002. ISBN 90-420-1126-2.

Parisa Kordjamshidi, Paolo Frasconi, Martijn van Otterlo, Marie-Francine Moens, and Luc De Raedt. Relational learning for spatial relation extraction from natural language. In *Inductive Logic Programming*, pages 204–220. Springer, 2012. doi: 10.1007/978-3-642-31951-8_{}20. URL https://lirias.kuleuven.be/handle/123456789/337956.

Stefan Kramer, Nada Lavrač, and Peter Flach. Propositionalization approaches to relational data mining. In Saěso Dězeroski, editor, *Relational Data Mining*, pages 262–286. Springer-Verlag New York, Inc., New York, NY, USA, 2000. ISBN 3-540-42289-7. URL http://dl.acm.org/citation.cfm?id=567222.567236.

Kenichi Kurihara and Taisuke Sato. Variational Bayesian grammar induction for natural language. In *Proceedings of the 8th International Conference on Grammatical Inference: Algorithms and Applications*, ICGI'06, pages 84–96, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-45264-8, 978-3-540-45264-5. doi: 10.1007/11872436_8. URL http://dx.doi.org/10.1007/11872436_8.

George Lakoff. Hedges: A study in meaning criteria and the logic of fuzzy concepts. *Journal of Philosophical Logic*, 2:458–508, 1973. doi: 10.1007/BF00262952.

Niels Landwehr, Andrea Passerini, Luc De Raedt, and Paolo Frasconi. kFOIL: Learning simple relational kernels. In *ILP'06, 16th International Conference on Inductive Logic Programming, Short Papers*, pages 125–127, 2006. URL: http://www.cs.kuleuven.ac.be/cgi-bin-dtai/publ_info.pl?id=42697.

Niels Landwehr, Kristian Kersting, and Luc De Raedt. Integrating naive Bayes and FOIL. *Journal of Machine Learning Research*, 8:481–507, May 2007. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1248659.1248677.

Ni Lao and William W. Cohen. Relational retrieval using a combination of path-constrained random walks. *Machine Learning*, 81(1):53–67, October 2010. ISSN 0885-6125. doi: 10.1007/s10994-010-5205-8. URL http://dx.doi.org/10.1007/s10994-010-5205-8.

Ni Lao, Tom Mitchell, and William W. Cohen. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 529–539, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-937284-11-4. URL http://dl.acm.org/citation.cfm?id=2145432.2145494.

Ni Lao, Amarnag Subramanya, Fernando Pereira, and William W. Cohen. Reading the Web with learned syntactic-semantic inference rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1017–1026, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390948.2391061.

Nada Lavrac, Johannes Furnkranz, and Dragan Gamberger. *Foundations of rule learning*. Springer Berlin Heidelberg, 2012.

Nada Lavrač and Sašo Džeroski. *Relational Data Mining*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 2001. ISBN 3540422897.

Maria Liakata and Stephen Pulman. Learning theories from text. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA, 2004a. Association for Computational Linguistics. doi: 10.3115/1220355.1220382. URL http://dx.doi.org/10.3115/1220355.1220382.

Maria Liakata and Stephen Pulman. Learning domain theories. In Nicolas Nicolov, Kalina Botcheva, Galia Angelova, and Ruslan Mitkov, editors, *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*, pages 29–44. Johns Benjamins, Amsterdam/Philadelphia, 2004b.

Marc Light, Xin Y. Qiu, and Padmini Srinivasan. The language of bioscience: Facts, speculations, and statements in between. In Lynette Hirschman and James Pustejovsky, editors, *HLT-NAACL 2004 Workshop: BioLINK 2004, Linking Biological Literature, Ontologies and Databases*, pages 17–24, Boston, Massachusetts, USA, May 2004. Association for Computational Linguistics. URL http://www.aclweb.org/anthology-new/W/W04/W04-3103.bib.

Xiao Ling and Daniel S. Weld. Temporal information extraction. In Maria Fox and David Poole, editors, *AAAI*. AAAI Press, 2010.

Francesca A. Lisi. Learning onto-relational rules with inductive logic programming. *CoRR*, abs/1210.2984, 2012.

Marco Lui. Feature stacking for sentence classification in evidence-based medicine. In *Proceedings of the Australasian Language Technology Association Workshop 2012*, pages 134–138, 2012. URL http://aclweb.org/anthology/U12-1019.

Suresh Manandhar, Sašo Džeroski, and Tomaž Erjavec. Learning multilingual morphology with Clog. In David Page, editor, *Inductive Logic Programming*, volume 1446 of *Lecture Notes in Computer Science*, pages 135–144. Springer Berlin Heidelberg, 1998. ISBN 978-3-540-64738-6. doi: 10.1007/BFb0027317. URL http://dx.doi.org/10.1007/BFb0027317.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-13360-1.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008. ISBN 0521865719, 9780521865715.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, June 1993. ISSN 0891-2017. URL http://dl.acm.org/citation.cfm?id=972470.972475.

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 523–534, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390948.2391009.

Andrew McCallum and Ben Wellner. Conditional models of identity uncertainty with application to noun coreference. In *NIPS*, 2004.

Andrew McCallum, Karl Schultz, and Sameer Singh. FACTORIE: Probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems (NIPS)*, 2009.

John McCarthy. What is artificial intelligence. http://www-formal.stanford.edu/jmc/whatisai/whatisai.html, 2007. [Online; accessed June 2, 2014].

David McClosky. *Any Domain Parsing: Automatic Domain Adaptation for Natural Language Parsing*. PhD thesis, Brown University, Providence, RI, USA, 2010. AAI3430199.

Ben Medlock. Exploring hedge identification in biomedical literature. *Journal of Biomedical Informatics*, 41(4):636 – 654, 2008. ISSN 1532-0464. doi: http://dx.doi.org/10.1016/j.jbi.2008.01.001. URL http://www.sciencedirect.com/science/article/pii/S1532046408000087.

Ben Medlock and Ted Briscoe. Weakly supervised learning for hedge classification in scientific literature. In John A. Carroll, Antal van den Bosch, and Annie Zaenen, editors, *ACL*. The Association for Computational Linguistics, 2007.

Sauro Menchetti, Fabrizio Costa, and Paolo Frasconi. Weighted decomposition kernels. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pages 585–592, New York, NY, USA, 2005. ACM. ISBN 1-59593-180-5. doi: 10.1145/1102351.1102425. URL http://doi.acm.org/10.1145/1102351.1102425.

James Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 209(441-458):415–446, 1909. doi: 10.1098/rsta.1909.0016. URL http://rsta.royalsocietypublishing.org/content/209/441-458/415.short.

Ivan Meza-Ruiz and Sebastian Riedel. Jointly identifying predicates, arguments and senses using Markov logic. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 155–163, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-41-1. URL http://dl.acm.org/citation.cfm?id=1620754.1620777.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013. URL http://papers.nips.cc/book/advances-in-neural-information-processing-systems-26-2013.

Brian Milch, Bhaskara Marthi, Stuart J. Russell, David Sontag, Daniel L. Ong, and Andrey Kolobov. Blog: Probabilistic models with unknown objects. In Luc De Raedt, Thomas G. Dietterich, Lise Getoor, and Stephen Muggleton, editors, *Probabilistic, Logical and Relational Learning*, volume 05051 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2005.

George A. Miller. Wordnet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, November 1995. ISSN 0001-0782. doi: 10.1145/219717.219748. URL http://doi.acm.org/10.1145/219717.219748.

Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997. ISBN 0070428077, 9780070428072.

Ruslan Mitkov. *The Oxford Handbook of Computational Linguistics (Oxford Handbooks in Linguistics S.)*. Oxford University Press, 2003. ISBN 0198238827.

Yoko Mizuta, Anna Korhonen, Tony Mullen, and Nigel Collier. Zone analysis in biology articles as a basis for information extraction. *International Journal of Medical Informatics*, 75(6):468 – 487, 2006. ISSN 1386-5056.

doi: http://dx.doi.org/10.1016/j.ijmedinf.2005.06.013. URL http://www.sciencedirect.com/science/article/pii/S138650560500122X. Recent Advances in Natural Language Processing for Biomedical Applications Special Issue.

Diego Mollá. Experiments with clustering-based features for sentence classification in medical publications: Macquarie test's participation in the ALTA 2012 shared task. In *Proceedings of the Australasian Language Technology Association Workshop 2012*, pages 139–142, 2012. URL http://aclweb.org/anthology/U12-1020.

Diego Mollá and Maria Elena Santiago-Martínez. Development of a corpus for evidence based medicine summarisation. In *Proceedings of the 2011 Australasian Language Technology Workshop (ALTA 2011)*, pages 86–94, Canberra, Australia, 2011.

Raymond J. Mooney. Inductive logic programming for natural language processing. In Stephen Muggleton, editor, *Inductive Logic Programming Workshop*, volume 1314 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 1996. ISBN 3-540-63494-0.

Raymond J. Mooney and Mary Elaine Califf. Induction of first-order decision lists: Results on learning the past tense of English verbs. *Journal of Artificial Intelligence Research (JAIR)*, 3:1–24, 1995.

James Moor, editor. *The Turing Test: the Elusive Standard of Artificial Intelligence*. Kluwer Academic Publishers, Dordrecht, 2003.

Roser Morante, Vincent Van Asch, and Walter Daelemans. A memory-based learning approach to event extraction in biomedical texts. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, BioNLP '09, pages 59–67, Stroudsburg, PA, USA, 2009a. Association for Computational Linguistics. ISBN 978-1-932432-44-2. URL http://dl.acm.org/citation.cfm?id=1572340.1572349.

Roser Morante, Vincent Van Asch, and Antal van den Bosch. Joint memory-based learning of syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, pages 25–30, Stroudsburg, PA, USA, 2009b. Association for Computational Linguistics. ISBN 978-1-932432-29-9. URL http://dl.acm.org/citation.cfm?id=1596409.1596413.

Roser Morante, Vincent Van Asch, and Walter Daelemans. Memory-based resolution of in-sentence scopes of hedge cues. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning — Shared Task*,

CoNLL '10: Shared Task, pages 40–47, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. ISBN 978-1-932432-84-8. URL http://dl.acm.org/citation.cfm?id=1870535.1870541.

Roser Morante, Martin Krallinger, Alfonso Valencia, and Walter Daelemans. Machine reading of biomedical texts about Alzheimer's disease. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF (Online Working Notes/Labs/Workshop)*, 2012. ISBN 978-88-904810-3-1.

Alessandro Moschitti. A study on convolution kernels for shallow semantic parsing. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. doi: 10.3115/1218955.1218998. URL http://dx.doi.org/10.3115/1218955.1218998.

Alessandro Moschitti. State-of-the-art kernels for natural language processing. In *Tutorial Abstracts of ACL 2012*, ACL '12, pages 2–2, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390500.2390502.

Alessandro Moschitti and Fabio Massimo Zanzotto. Fast and effective kernels for relational learning from texts. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 649–656, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273578. URL http://doi.acm.org/10.1145/1273496.1273578.

S. H Muggleton. Stochastic logic programs. In L. De Raedt, editor, *Advances in Inductive Logic Programming*, pages 254–264. IOS Press, 1996.

Stephen Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13(3-4):245–286, 1995. ISSN 0288-3635. doi: 10.1007/BF03037227. URL http://dx.doi.org/10.1007/BF03037227.

Stephen Muggleton and Michael Bain. Analogical prediction. In *Proceedings of the 9th International Workshop on Inductive Logic Programming*, ILP '99, pages 234–244, London, UK, UK, 1999. Springer-Verlag. ISBN 3-540-66109-3. URL http://dl.acm.org/citation.cfm?id=647999.742799.

Stephen Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19/20:629–679, 1994.

Stephen Muggleton, Christopher H. Bryant, and Ashwin Srinivasan. Learning chomsky-like grammars for biological sequence families. In Pat Langley, editor, *ICML*, pages 631–638. Morgan Kaufmann, 2000. ISBN 1-55860-707-2.

Stephen H. Muggleton and Dianhuan Lin. Meta-interpretive learning of higher-order dyadic Datalog: Predicate invention revisited. In Francesca Rossi, editor, *IJCAI*. IJCAI/AAAI, 2013. ISBN 978-1-57735-633-2.

Stephen H. Muggleton, Dianhuan Lin, Niels Pahlavi, and Alireza Tamaddoni-Nezhad. Meta-interpretive learning: application to grammatical inference. *Machine Learning*, 94(1):25–49, 2014. ISSN 0885-6125. doi: 10.1007/s10994-013-5358-3. URL http://dx.doi.org/10.1007/s10994-013-5358-3.

Un Yong Nahm and Raymond J. Mooney. A mutually beneficial integration of data mining and information extraction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, pages 627–632, Austin, TX, July 2000. URL http://www.cs.utexas.edu/users/ai-lab/?nahm:aaai00.

Chieko Nakabasami. Interactive background knowledge acquisition for inducing differences among documents. In Luboš Popelínský and Miloslav Nepil, editors, *Learning Language in Logic*, pages 47–57, Strasbourg, France, September 2001.

Claire Nédellec. Corpus-based learning of semantic relations by the ILP system, Asium. In James Cussens, editor, *Learning Language in Logic*, pages 28–39, Bled, Slovenia, June 1999. URL http://www.cs.york.ac.uk/mlg/lll/workshop/proceedings/Nedellec/Nedellec.ps.gz.

Claire Nédellec. Learning language in logic – genic interaction extraction challenge. In James Cussens and Claire Nédellec, editors, *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 31–37, Bonn, August 2005. URL http://www.cs.york.ac.uk/aig/lll/lll05/lll05-nedellec.pdf.

Miloslav Nepil. Learning to parse from a treebank: Combining TBL and ILP. In *Proceedings of the 11th International Conference on Inductive Logic Programming*, ILP '01, pages 179–192, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-42538-1. URL http://dl.acm.org/citation.cfm?id=648001.757276.

Miloslav Nepil, Luboš Popelínský, and Eva Žáčková. Part-of-speech tagging by means of shallow parsing, ILP and active learning. In Luboš Popelínský and Miloslav Nepil, editors, *Learning Language in Logic*, pages 58–66, Strasbourg, France, September 2001.

Jennifer Neville and David Jensen. Leveraging relational autocorrelation with latent group models. In *ICDM*, pages 322–329. IEEE Computer Society, 2005. ISBN 0-7695-2278-5.

Mathias Niepert, Christian Meilicke, and Heiner Stuckenschmidt. A probabilistic-logical framework for ontology matching. In Maria Fox and David Poole, editors, *AAAI*. AAAI Press, 2010.

Mathias Niepert, Jan Noessner, Christian Meilicke, and Heiner Stuckenschmidt. Probabilistic-logical web data integration. In Axel Polleres, Claudia Amato, Marcelo Arenas, Siegfried Handschuh, Paula Kroner, Sascha Ossowski, and Peter Patel-Schneider, editors, *Reasoning Web. Semantic Technologies for the Web of Data*, volume 6848 of *Lecture Notes in Computer Science*, pages 504–533. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-23031-8. doi: 10.1007/978-3-642-23032-5_11. URL http://dx.doi.org/10.1007/978-3-642-23032-5_11.

S. Nijssen and J. Kok. Faster association rules for multiple relations. In *IJCAI01*, pages 891–896, 2001.

Vladimir Nikulin. Random sets approach and its applications. In Isabelle Guyon, Constantin F. Aliferis, Gregory F. Cooper, André Elisseeff, Jean-Philippe Pellet, Peter Spirtes, and Alexander R. Statnikov, editors, *WCCI Causation and Prediction Challenge*, volume 3 of *JMLR Proceedings*, pages 65–76. JMLR.org, 2008.

Yun Niu, Graeme Hirst, Gregory McArthur, and Patricia Rodriguez-Gianolli. Answering clinical questions with role identification. In *Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine - Volume 13*, BioMed '03, pages 73–80, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1118958.1118968. URL http://dx.doi.org/10.3115/1118958.1118968.

Joakim Nivre. *Inductive Dependency Parsing (Text, Speech and Language Technology)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 1402048882.

Timothy J. O'Donnell, Noah D. Goodman, and Joshua B. Tenenbaum. Fragment grammars: Exploring computation and reuse in language. Technical Report MIT-CSAIL-TR-2009-013, Massachusetts Institute of Technology, 2009. URL http://dspace.mit.edu/handle/1721.1/44963.

Irene M. Ong, Inĕs de Castro Dutra, David Page, and Vítor Santos Costa. Mode directed path finding. In *Machine Learning: ECML 2005*, volume 3720 of *Lecture Notes in Computer Science*, pages 673–681. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-29243-2. doi: 10.1007/11564096_68. URL http://dx.doi.org/10.1007/11564096_68.

Anselmo Peñas, Eduard Hovy, Pamela Forner, Álvaro Rodrigo, Richard Sutcliffe, and Roser Morante. QA4MRE 2011-2013: Overview of question

answering for machine reading evaluation. In Pamela Forner, Henning uller, Roberto Paredes, Paolo Rosso, and Benno Stein, editors, *Information Access Evaluation. Multilinguality, Multimodality, and Visualization*, volume 8138 of *Lecture Notes in Computer Science*, pages 303–320. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-40801-4. doi: 10.1007/978-3-642-40802-1_29. URL http://dx.doi.org/10.1007/978-3-642-40802-1_29.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

Anselmo Peñas, Eduard H. Hovy, Pamela Forner, Álvaro Rodrigo, Richard F. E. Sutcliffe, Caroline Sporleder, Corina Forascu, Yassine Benajiba, and Petya Osenova. Overview of QA4MRE at CLEF 2012: Question answering for machine reading evaluation. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF (Online Working Notes/Labs/Workshop)*, 2012. ISBN 978-88-904810-3-1.

F Pereira and D Warren. Definite clause grammars for language analysis. In Barbara J. Grosz, Karen Sparck-Jones, and Bonnie Lynn Webber, editors, *Readings in Natural Language Processing*, pages 101–124. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1986. ISBN 0-934613-11-7. URL http://dl.acm.org/citation.cfm?id=21922.24330.

Emmanouil Antonios Platanios, Avrim Blum, and Tom M Mitchell. Estimating accuracy from unlabeled data. In *Conference on Uncertainty in Artificial Intelligence*, pages 1–10, 2014.

Gordon D. Plotkin. A note on inductive generalization. *Machine Intelligence*, 5:153–163, 1970.

David Poole. Abducing through negation as failure: stable models within the independent choice logic. *Journal of Logic Programming*, 44(1-3):5–35, 2000.

Hoifung Poon. *Markov Logic for Machine Reading*. PhD thesis, University of Washington, Seattle, WA, USA, 2011. AAI3485537.

Hoifung Poon. Grounded unsupervised semantic parsing. In *ACL (1)*, pages 933–943. The Association for Computer Linguistics, 2013. ISBN 978-1-937284-50-3.

Hoifung Poon and Pedro Domingos. Joint inference in information extraction. In *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 1*, AAAI'07, pages 913–918. AAAI Press, 2007. ISBN 978-1-57735-323-2. URL http://dl.acm.org/citation.cfm?id=1619645.1619792.

Hoifung Poon and Pedro Domingos. Joint unsupervised coreference resolution with Markov logic. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 650–659, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1613715.1613796.

Hoifung Poon and Pedro Domingos. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 1–10, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-59-6. URL http://dl.acm.org/citation.cfm?id=1699510.1699512.

Hoifung Poon and Pedro Domingos. Unsupervised ontology induction from text. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 296–305, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1858681.1858712.

Hoifung Poon and Lucy Vanderwende. Joint inference for knowledge extraction from biomedical literature. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 813–821, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. ISBN 1-932432-65-5. URL http://dl.acm.org/citation.cfm?id=1857999.1858122.

Luboĭ Popelínský and Tomáš Pavelek. Mining lemma disambiguation rules from Czech corpora. In Jan M. Żytkow and Jan Rauch, editors, *Principles of Data Mining and Knowledge Discovery*, volume 1704 of *Lecture Notes in Computer Science*, pages 498–503. Springer Berlin Heidelberg, 1999. ISBN 978-3-540-66490-1. doi: 10.1007/978-3-540-48247-5_64. URL http://dx.doi.org/10.1007/978-3-540-48247-5_64.

Luboš Popelínský and Jan Blaťák. Learning genic interactions without expert domain knowledge: Comparison of different ILP algorithms. In James Cussens and Claire Nédellec, editors, *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 59–61, Bonn, August 2005. URL http://www.cs.york.ac.uk/aig/lll/lll05/lll05-popelinsky.pdf.

Martin F Porter. An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137, 1980.

Jay Pujara, Hui Miao, Lise Getoor, and William Cohen. Knowledge graph identification. In Harith Alani, Lalana Kagal, Achille Fokoue, Paul Groth, Chris Biemann, JosianeXavier Parreira, Lora Aroyo, Natasha Noy, Chris Welty, and Krzysztof Janowicz, editors, *The Semantic Web -*

ISWC 2013, volume 8218 of *Lecture Notes in Computer Science*, pages 542–557. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-41334-6. doi: 10.1007/978-3-642-41335-3_34. URL http://dx.doi.org/10.1007/978-3-642-41335-3_34.

Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Bjorne, Jorma Boberg, Jouni Jarvinen, and Tapio Salakoski. BioInfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8(1):50, 2007. ISSN 1471-2105. doi: 10.1186/1471-2105-8-50. URL http://www.biomedcentral.com/1471-2105/8/50.

Brian Quanz and Jun Huan. Aligned graph classification with regularized logistic regression. In *SDM*, pages 353–364. SIAM, 2009.

J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, March 1986. ISSN 0885-6125. doi: 10.1023/A:1022643204877. URL http://dx.doi.org/10.1023/A:1022643204877.

J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5: 239–266, 1990.

Sindhu Raghavan and Raymond J. Mooney. Online inference-rule learning from natural-language extractions. In *Proceedings of the 3rd Statistical Relational AI (StaRAI-13) workshop at AAAI '13*, July 2013. URL http://www.cs.utexas.edu/users/ai-lab/?raghavan:starai13.

Sindhu Raghavan, Raymond J. Mooney, and Hyeonseo Ku. Learning to "read between the lines" using Bayesian logic programs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 349–358, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390524.2390574.

Jan Ramon. *Conceptuele Clustering en Instance-based Leren in Eerste-orde Logica*. PhD thesis, Informatics Section, Department of Computer Science, Faculty of Engineering Science, October 2002. URL https://lirias.kuleuven.be/handle/123456789/260589. Bruynooghe, Maurice and De Raedt, Luc (supervisors).

Bradley L. Richards and Raymond J. Mooney. Learning relations by pathfinding. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, pages 50–55, San Jose, CA, July 1992. URL http://www.cs.utexas.edu/users/ai-lab/?richards:aaai92.

Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, February 2006. ISSN 0885-6125.

doi: 10.1007/s10994-006-5833-1. URL http://dx.doi.org/10.1007/s10994-006-5833-1.

Sebastian Riedel and Ewan Klein. Genic interaction extraction with semantic and syntactic chains. In James Cussens and Claire Nédellec, editors, *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, pages 69–74, Bonn, August 2005. URL http://www.cs.york.ac.uk/aig/lll/lll05/lll05-riedel.pdf.

Sebastian Riedel and Ivan Meza-Ruiz. Collective semantic role labelling with Markov logic. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, CoNLL '08, pages 193–197, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-48-4. URL http://dl.acm.org/citation.cfm?id=1596324.1596357.

Sebastian Riedel, Hong-Woo Chun, Toshihisa Takagi, and Jun'ichi Tsujii. A Markov logic approach to bio-molecular event extraction. In *Proceedings of the Natural Language Processing in Biomedicine NAACL 2009 Workshop (BioNLP '09)*, pages 41–49, 2009.

Stefan Riezler. Learning log-linear models on constraint-based grammars for disambiguation. In James Cussens and Sašo Džeroski, editors, *Learning Language in Logic*, volume 1925 of *LNCS*, pages 199–217. Springer Berlin Heidelberg, June 2000. ISBN 3-540-41145-3. URL http://www.springer.de/cgi-bin/search_book.pl?isbn=3-540-41145-3.

Ellen Riloff, Janyce Wiebe, and Theresa Wilson. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 25–32, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1119176.1119180. URL http://dx.doi.org/10.3115/1119176.1119180.

Nick Rizzolo and Dan Roth. Learning based Java for rapid development of NLP systems. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *LREC*. European Language Resources Association, 2010. ISBN 2-9517408-6-7.

William Rosenberg and Anna Donald. Evidence based medicine: an approach to clinical problem-solving. *British Medical Journal*, 310(6987):1122–1126, 4 1995. doi: 10.1136/bmj.310.6987.1122.

Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, 2004. URL http://aclweb.org/anthology/W04-2401.

Dan Roth and Wen-tau Yih. Global inference for entity and relation identification via a linear programming formulation. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009. ISBN 0136042597, 9780136042594.

Kenji Sagae and Jun Tsujii. Dependency parsing and domain adaptation with data-driven LR models and parser ensembles. In *Proceedings of the CoNLL 2007 Shared Task. Joint Conferences on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Prague, Czech Republic, 7 2007.

Abeed Sarker, Diego Mollá, and Cecile Paris. An approach for automatic multi-label classification of medical sentences. In *Proceedings of the 4th International Louhi Workshop on Health Document Text Mining and Information Analysis (Louhi 2013)*, 2013.

Taisuke Sato and Yoshitaka Kameya. PRISM: A language for symbolic-statistical modeling. In *Proceedings of the Fifteenth International Joint Conference on Artifical Intelligence - Volume 2*, IJCAI'97, pages 1330–1335, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. ISBN 1-555860-480-4. URL http://dl.acm.org/citation.cfm?id=1622270.1622348.

Taisuke Sato and Yoshitaka Kameya. Parameter learning of logic programs for symbolic-statistical modeling. *Journal of Artificial Intelligence Research*, 15 (1):391–454, December 2001. ISSN 1076-9757. URL http://dl.acm.org/citation.cfm?id=1622845.1622858.

Taisuke Sato, Yoshitaka Kameya, and Kenichi Kurihara. Variational Bayes via propositionalized probability computation in PRISM. *Annals of Mathematics and Artificial Intelligence*, 54(1-3):135–158, November 2008. ISSN 1012-2443. doi: 10.1007/s10472-009-9135-8. URL http://dx.doi.org/10.1007/s10472-009-9135-8.

Sandeepkumar Satpal, Sahely Bhadra, Sundararajan Sellamanickam, Rajeev Rastogi, and Prithviraj Sen. Web information extraction using Markov logic networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1406–1414, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0813-7. doi: 10.1145/2020408.2020615. URL http://doi.acm.org/10.1145/2020408.2020615.

Stefan Schoenmackers, Oren Etzioni, Daniel S. Weld, and Jesse Davis. Learning first-order Horn clauses from Web text. In *Proceedings of the*

2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10, pages 1088–1098, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1870658.1870764.

Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, March 2002. ISSN 0360-0300. doi: 10.1145/505282.505283. URL http://doi.acm.org/10.1145/505282.505283.

Pascale Sébillot, Pierrette Bouillon, and Cécile Fabre. Inductive logic programming for corpus-based acquisition of semantic lexicons. In *Proceedings of the 2Nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning - Volume 7*, CoNLL '00, pages 199–208, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics. doi: 10.3115/1117601.1117649. URL http://dx.doi.org/10.3115/1117601.1117649.

Ehud Y. Shapiro. An algorithm that infers theories from facts. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'81, pages 446–451, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1623156.1623240.

Sameer Singh, Karl Schultz, and Andrew McCallum. Bi-directional joint inference for entity resolution and segmentation using imperatively-defined factor graphs. In Wray Buntine, Marko Grobelnik, Dunja Mladenič, and John Shawe-Taylor, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5782 of *Lecture Notes in Computer Science*, pages 414–429. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-04173-0. doi: 10.1007/978-3-642-04174-7_27. URL http://dx.doi.org/10.1007/978-3-642-04174-7_27.

Parag Singla and Pedro Domingos. Entity resolution with Markov logic. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06, pages 572–582, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2701-9. doi: 10.1109/ICDM.2006.65. URL http://dx.doi.org/10.1109/ICDM.2006.65.

Burrhus Frederic Skinner. *Verbal Behavior*. Appleton-Century-Crofts New York, 1957.

Yang Song, Jing Jiang, Wayne Xin Zhao, Sujian Li, and Houfeng Wang. Joint learning for coreference resolution with Markov logic. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL

'12, pages 1245–1254, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390948.2391089.

Mohammad S Sorower, Janardhan R Doppa, Walker Orr, Prasad Tadepalli, Thomas G Dietterich, and Xiaoli Z Fern. Inverting Grice's maxims to learn rules from natural language extractions. In *Advances in Neural Information Processing Systems*, pages 1053–1061, 2011.

Lucia Specia. A hybrid relational approach for WSD - first results. In Nicoletta Calzolari, Claire Cardie, and Pierre Isabelle, editors, *ACL*. The Association for Computer Linguistics, 2006.

Lucia Specia, Ashwin Srinivasan, Ganesh Ramakrishnan, and Maria das Graças Volpe Nunes. Word sense disambiguation using inductive logic programming. In Stephen Muggleton, Ramón P. Otero, and Alireza Tamaddoni-Nezhad, editors, *Proceedings of the 16th International Conference on Inductive Logic Programming (ILP 2006), Revised Selected Papers*, volume 4455 of *Lecture Notes in Computer Science*, pages 409–423. Springer, 2006. ISBN 978-3-540-73846-6.

Lucia Specia, Maria Graças, Volpe Nunes, Ashwin Srinivasan, and Ganesh Ramakrishnan. USP-IBM-1 and USP-IBM-2: The ILP-based systems for lexical sample WSD in SemEval-2007. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 442–445. Association for Computational Linguistics, 2007. URL http://aclweb.org/anthology/S07-1099.

Craig Stanfill and David Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228, December 1986. ISSN 0001-0782. doi: 10.1145/7902.7906. URL http://doi.acm.org/10.1145/7902.7906.

Richard Sutcliffe, Anselmo Peñas, Eduard Hovy, Pamela Forner, Álvaro Rodrigo, Corina Forascu, Yassine Benajiba, and Petya Osenova. Overview of QA4MRE main task at CLEF 2013. In *Working Notes, CLEF*, 2013.

György Szarvas. Hedge classification in biomedical texts with a weakly supervised selection of keywords. In Kathleen McKeown, Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, editors, *ACL*, pages 281–289. The Association for Computer Linguistics, 2008. ISBN 978-1-932432-04-6.

Songbo Tan. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*, 28(4):667–671, May 2005. ISSN 0957-4174. doi: 10.1016/j.eswa.2004.12.023. URL http://dx.doi.org/10.1016/j.eswa.2004.12.023.

L. R. Tang and Raymond J. Mooney. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proceedings of the 12th European Conference on Machine Learning*, pages 466–477, Freiburg, Germany, 2001.

Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, UAI'02, pages 485–492, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. ISBN 1-55860-897-4. URL http://dl.acm.org/citation.cfm?id=2073876.2073934.

Benjamin Taskar, Carlos Guestrin, and Daphne Koller. Max-margin Markov networks. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *NIPS*. MIT Press, 2003. ISBN 0-262-20152-6.

Cynthia A. Thompson and Mary Elaine Califf. Improving learning by choosing examples intelligently in two natural language tasks. In James Cussens and Sašo Džeroski, editors, *Learning Language in Logic*, volume 1925 of *LNCS*, pages 279–299. Springer Berlin Heidelberg, June 2000. ISBN 3-540-41145-3. URL http://www.springer.de/cgi-bin/search_book.pl?isbn=3-540-41145-3.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58(1):267–288, 1996.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1073445.1073478. URL http://dx.doi.org/10.3115/1073445.1073478.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. A global joint model for semantic role labeling. *Computational Linguistics*, 34(2): 161–191, June 2008. ISSN 0891-2017. doi: 10.1162/coli.2008.34.2.161. URL http://dx.doi.org/10.1162/coli.2008.34.2.161.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6:1453–1484, September 2005.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured

output spaces. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 104–, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5. doi: 10.1145/1015330.1015341. URL http://doi.acm.org/10.1145/1015330.1015341.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis P. Vlahavas. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer, 2010.

A. M. Turing. Computing machinery and intelligence, 1950. URL http://cogprints.org/499/.

Naushad UzZaman and James F. Allen. Trips and trios system for tempeval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 276–283, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1859664.1859726.

Antal van den Bosch. Wrapped progressive sampling search for optimizing learning algorithm parameters. In *Proceedings of the Sixteenth Belgian-Dutch Conference on Artificial Intelligence*, pages 219–226, 2004.

Jason Van Hulse, Taghi M. Khoshgoftaar, and Amri Napolitano. Experimental perspectives on learning from imbalanced data. In Zoubin Ghahramani, editor, *ICML*, volume 227 of *ACM International Conference Proceeding Series*, pages 935–942. ACM, 2007. ISBN 978-1-59593-793-3.

C. J. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979. ISBN 0408709294.

Mathias Verbeke and Jesse Davis. A text mining approach as baseline for QA4MRE'12. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Rome, Italy, September 17-20, 2012*, September 2012. URL https://lirias.kuleuven.be/handle/123456789/356434. Conference and Labs of the Evaluation Forum (CLEF), Rome, Italy, 17-20 September 2012.

Mathias Verbeke, Bettina Berendt, and Siegfried Nijssen. Data mining, interactive semantic structuring, and collaboration: a diversity-aware method for sense-making in search. In Claudia Niederee, editor, *Proceedings of First International Workshop on Living Web, Collocated with the 8th International Semantic Web Conference (ISWC-2009), Washington D.C., USA, October 26, 2009, Living Web Workshop, Washington, 26 Oct 2009*, page 8. CEUR-WS, October 2009. URL https://lirias.kuleuven.be/handle/123456789/253393.

Mathias Verbeke, Ilija Subasic, and Bettina Berendt. Guiding user groupings. In *Proceedings of the 3rd International Workshop on Mining Ubiquitous and Social Environments (MUSE), Mining Ubiquitous and Social Environments (MUSE), Bristol, UK, 24 September 2012*, 2012. URL https://lirias.kuleuven.be/handle/123456789/356404.

Mathias Verbeke, Bettina Berendt, Leen d'Haenens, and Michaël Opgenhaffen. When two disciplines meet, data mining for communication science. Annual Meeting of International Communication Association (ICA), Seattle, U.S., 22-26 May 2014, May 2014a. URL https://lirias.kuleuven.be/handle/123456789/436424.

Mathias Verbeke, Ilija Subasic, and Bettina Berendt. How to carve up the world: Learning and collaboration for structure recommendation. In Martin Atzmueller, Alvin Chin, Denis Helic, and Andreas Hotho, editors, *MSM-MUSE 2012 Postproceedings, MSM-MUSE, 2012*, 2014b. URL https://lirias.kuleuven.be/handle/123456789/424131.

Eva Žáčková, Luboš Popelínský, and Miloslav Nepil. Automatic tagging of compound verb groups in Czech corpora. In Petr Sojka, Ivan Kopeček, and Karel Pala, editors, *Text, Speech and Dialogue*, volume 1902 of *Lecture Notes in Computer Science*, pages 115–120. Springer Berlin Heidelberg, 2000. ISBN 978-3-540-41042-3. doi: 10.1007/3-540-45323-7_20. URL http://dx.doi.org/10.1007/3-540-45323-7_20.

William Y. Wang, Kathryn Mazaitis, and William W. Cohen. Structure learning via parameter learning. In *23rd ACM International Conference on Information and Knowledge Management*, 2014a.

William Yang Wang, Kathryn Mazaitis, and William W Cohen. ProPPR: Efficient first-order probabilistic logic programming for structure discovery, parameter learning, and scalable inference. *Proceedings of the AAAI 2014 Workshop on Statistical Relational AI (StarAI 2014)*, 2014b.

William Yang Wang, Kathryn Mazaitis, Ni Lao, Tom M. Mitchell, and William W. Cohen. Efficient inference and learning in a large knowledge base: Reasoning with extracted information using a locally groundable first-order probabilistic logic. *CoRR*, abs/1404.3301, 2014c.

Jason Weston, André Elisseeff, Bernhard Schölkopf, and Mike Tipping. Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, March 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=944919.944982.

Terry Winograd. Understanding natural language. *Cognitive Psychology*, 3(1): 1–191, 1972.

Rüdiger Wirth. Learning by failure to prove. In *Proceedings of the Third European Working Session on Learning*, pages 237–251. Pitman, 1988.

Rüdiger Wirth. Completing logic programs by inverse resolution. In *Proceedings of the Fourth European Working Session on Learning*, pages 239–250. Pitman, 1989.

Xiaofeng Yang, Jian Su, Jun Lang, Chew Lim Tan, Ting Liu, and Sheng Li. An entity-mention model for coreference resolution with inductive logic programming. In Kathleen McKeown, Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, editors, *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL 2008)*, pages 843–851. The Association for Computer Linguistics, 2008. ISBN 978-1-932432-04-6.

Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. Jointly identifying temporal relations with Markov logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pages 405–413, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-45-9. URL http://dl.acm.org/citation.cfm?id=1687878.1687936.

Katsumasa Yoshikawa, Sebastian Riedel, Tsutomu Hirao, Masayuki Asahara, and Yuji Matsumoto. Coreference based event-argument relation extraction on biomedical text. In *Proceedings of the Fourth International Symposium on Semantic Mining in Biomedicine (SMBM' 10)*, 2010.

Xiaofeng Yu and Wai Lam. An integrated probabilistic and logic approach to encyclopedia relation extraction with multiple features. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 1065–1072, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6. URL http://dl.acm.org/citation.cfm?id=1599081.1599215.

Xiaofeng Yu and Wai Lam. Probabilistic joint models incorporating logic and learning via structured variational approximation for information extraction. *Knowledge and Information Systems*, 32(2):415–444, 2012. ISSN 0219-1377. doi: 10.1007/s10115-011-0455-8. URL http://dx.doi.org/10.1007/s10115-011-0455-8.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006. ISSN 1467-9868. doi: 10.1111/j.1467-9868.2005.00532.x. URL http://dx.doi.org/10.1111/j.1467-9868.2005.00532.x.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106, March 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=944919.944964.

John M. Zelle and Raymond J. Mooney. Learning semantic grammars with constructive inductive logic programming. In Richard Fikes and Wendy G. Lehnert, editors, *AAAI*, pages 817–822. AAAI Press / The MIT Press, 1993. ISBN 0-262-51071-5.

Dengyong Zhou and Bernhard Schölkopf. A regularization framework for learning from graph data. In *ICML Workshop on Statistical Relational Learning*, pages 132–137, 2004.

Jun Zhu, Zaiqing Nie, Xiaojiang Liu, Bo Zhang, and Ji-Rong Wen. StatSnowball: A statistical approach to extracting entity relationships. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 101–110, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-487-4. doi: 10.1145/1526709.1526724. URL http://doi.acm.org/10.1145/1526709.1526724.

Curriculum Vitae

Mathias Verbeke was born in Bruges, Belgium on February 28th, 1986. He went to school at the Sint-Franciscus-Xaveriusinstituut in Bruges and started higher education in 2004 at KU Leuven. In 2007, he obtained a Bachelor of Science in Informatics. Two years later, he finished his Master of Science in Informatics at the same university, and graduated magna cum laude. His Master thesis, supervised by Prof. Luc De Raedt and Prof. Bettina Berendt was titled "Integrating Data Mining in Literature Search Engines". Under the auspices of these same promotors, he started his doctoral studies at the Declarative Languages and Artificial Intelligence (DTAI) lab at KU Leuven. In December 2014, he will defend his doctoral thesis, titled "Statistical Relational Learning of Natural Language".

Publication List

Journal Articles

Luc De Raedt, Anton Dries, Ingo Thon, Guy Van den Broeck, and Mathias Verbeke. Inducing probabilistic relational rules from probabilistic examples. Submitted.

Mathias Verbeke, Bettina Berendt, Leen d'Haenens, and Michaël Opgenhaffen. Critical news reading with Twitter? An investigation into big-data practices and their impact on societal discourse. Submitted.

Book Chapters

Bettina Berendt, Mark Last, Ilija Subasic, and Mathias Verbeke. New formats and interfaces for multi-document news summarization and its evaluation. In Alessandro Fiori, editor, *Innovative Document Summarization Techniques: Revolutionizing Knowledge Understanding*. IGI Global, 2013. URL https://lirias.kuleuven.be/handle/123456789/423917.

Conference Papers

Mathias Verbeke, Vincent Van Asch, Walter Daelemans, and Luc De Raedt. Lazy and eager relational learning using graph-kernels. In Laurent Besacier, Adrian-Horia Dediu, and Carlos Martín-Vide, editors, *Proceedings of the Second International Conference on Statistical Language and Speech Processing, International Conference on Statistical Language and Speech Processing, Grenoble, France, 14-16 October 2014*, pages 171–184. Springer, October 2014b. URL https://lirias.kuleuven.be/handle/123456789/457709.

Mathias Verbeke, Paolo Frasconi, Kurt De Grave, Fabrizio Costa, and Luc De Raedt. kLogNLP: Graph kernel-based relational learning of natural language. In Kalina Bontcheva and Zhu Jingbo, editors, *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Annual Meeting of the Association for Computational Lingusitics (ACL), Baltimore, Maryland, USA, 22-27 June 2014*, pages 85–90. Association for Computational Linguistics, June 2014a. URL https://lirias.kuleuven.be/handle/123456789/451228.

Fabrizio Costa, Mathias Verbeke, and Luc De Raedt. Relational regularization and feature ranking. In *Proceedings of the 14th SIAM International Conference on Data Mining, SIAM International Conference on Data Mining, Philadephia, Pennsylvania, USA, 24-26 April 2014*, pages 650–658, 2014. URL https://lirias.kuleuven.be/handle/123456789/437691.

Mathias Verbeke, Vincent Van Asch, Roser Morante, Paolo Frasconi, Walter Daelemans, and Luc De Raedt. A statistical relational learning approach to identifying evidence based medicine categories. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP - CoNLL), Jeju Island, Korea, 13-14 July 2012*, pages 579–589. Association for Computational Linguistics, 2012b. URL https://lirias.kuleuven.be/handle/123456789/350664.

Mathias Verbeke, Paolo Frasconi, Vincent Van Asch, Roser Morante, Walter Daelemans, and Luc De Raedt. Kernel-based logical and relational learning with kLog for hedge cue detection. In Stephen H. Muggleton, Alireza Tamaddoni-Nezhad, and Francesca A. Lisi, editors, *Proceedings of the 21st International Conference on Inductive Logic Programming, Inductive Logic Programming, Windsor Great Park, UK, 31 July 2011 - 3 August 2011*, pages 347–357. Springer, March 2012a. URL https://lirias.kuleuven.be/handle/123456789/356403.

Mathias Verbeke, Paolo Frasconi, Vincent Van Asch, Roser Morante, Walter Daelemans, and Luc De Raedt. Kernel-based logical and relational learning with kLog for hedge cue detection. In *Preliminary Papers ILP 2011, ILP, Windsor, UK, 31 July - 3 August 2011*, pages 1–6, August 2011. URL https://lirias.kuleuven.be/handle/123456789/317052.

Workshop Papers and Posters

Willem Mattelaer, Mathias Verbeke, and Davide Nitti. KUL-Eval: A combinatory categorial grammar approach for improving semantic parsing of robot commands using spatial context. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 385–390, Dublin, Ireland, August 2014. Association for Computational Linguistics and Dublin City University. URL http://www.aclweb.org/anthology/S14-2066.

Mathias Verbeke, Bettina Berendt, Leen d'Haenens, and Michaël Opgenhaffen. When two disciplines meet, data mining for communication science. Annual Meeting of International Communication Association (ICA), Seattle, U.S., 22-26 May 2014, May 2014a. URL https://lirias.kuleuven.be/handle/123456789/436424.

Mathias Verbeke, Ilija Subasic, and Bettina Berendt. How to carve up the world: Learning and collaboration for structure recommendation. In Martin Atzmueller, Alvin Chin, Denis Helic, and Andreas Hotho, editors, *MSM-MUSE 2012 Postproceedings, MSM-MUSE, 2012*, 2014b. URL https://lirias.kuleuven.be/handle/123456789/424131.

Denis Lukovnikov, Mathias Verbeke, and Bettina Berendt. User interest prediction for tweets using semantic enrichment with DBpedia. In *Proceedings of The 25th Benelux Conference on Artificial Intelligence (BNAIC 2013), BNAIC, Delft, 7-8 November 2013*, pages 1–8, 2013. URL https://lirias.kuleuven.be/handle/123456789/423918.

Rocio Chongtay, Mark Last, Mathias Verbeke, and Bettina Berendt. Summarize to learn: Summarization and visualization of text for ubiquitous learning. In *Proceedings of the 3rd IEEE Workshop on Interactive Visual Text Analytics at VIS 2013, The 3rd IEEE Workshop on Interactive Visual Text Analytics, Atlanta, GA, 14 October 2013*, pages 1–4, 2013. URL https://lirias.kuleuven.be/handle/123456789/423905.

Guilerme de Oliveira Costa Marques and Mathias Verbeke. Combining text mining techniques for QA4MRE 2013. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Valencia, Spain, September 23-26, 2013*, September 2013. URL https://lirias.kuleuven.be/handle/123456789/416992. Conference and Labs of the Evaluation Forum (CLEF), Valencia, Spain, 23-26 September 2013.

Mathias Verbeke, Bettina Berendt, Leen d'Haenens, and Michaël Opgenhaffen. Data mining for computational journalism. European

Journalism Observatory (EJO) workshop, Berlin, Germany, 18-19 September 2013, September 2013. URL https://lirias.kuleuven.be/handle/123456789/423237.

Mathias Verbeke, Ilija Subasic, and Bettina Berendt. Guiding user groupings. In *Proceedings of the 3rd International Workshop on Mining Ubiquitous and Social Environments (MUSE), Mining Ubiquitous and Social Environments (MUSE), Bristol, UK, 24 September 2012*, 2012d. URL https://lirias.kuleuven.be/handle/123456789/356404.

Mathias Verbeke and Jesse Davis. A text mining approach as baseline for QA4MRE'12. In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Rome, Italy, September 17-20, 2012*, September 2012. URL https://lirias.kuleuven.be/handle/123456789/356434. Conference and Labs of the Evaluation Forum (CLEF), Rome, Italy, 17-20 September 2012.

Mathias Verbeke, Roser Morante, Paolo Frasconi, and Luc De Raedt. A statistical relational learning approach to identify sections in scientific abstracts using sentence and document structure. LSD, Leuven, 7-8 June 2012, June 2012c. URL https://lirias.kuleuven.be/handle/123456789/350663.

Mathias Verbeke, Roser Morante, Paolo Frasconi, and Luc De Raedt. A statistical relational learning approach to identify sections in scientific abstracts using sentence and document structure. In *Proceedings of the 21st Belgian-Dutch Conference on Machine Learning*, May 2012b. URL https://lirias.kuleuven.be/handle/123456789/350660. BeNeLearn, Ghent, 24-25 May 2012.

Mathias Verbeke, Paolo Frasconi, Vincent Van Asch, Roser Morante, Walter Daelemans, and Luc De Raedt. Kernel-based logical and relational learning with kLog for hedge cue detection. Meeting of Computational Linguistics in The Netherlands (CLIN), Tilburg, The Netherlands, 20 January 2012, January 2012a. URL https://lirias.kuleuven.be/handle/123456789/335365.

Mathias Verbeke, Bettina Berendt, and Siegfried Nijssen. Data mining, interactive semantic structuring, and collaboration: a diversity-aware method for sense-making in search. In Claudia Niederee, editor, *Proceedings of First International Workshop on Living Web, Collocated with the 8th International Semantic Web Conference (ISWC-2009), Washington D.C., USA, October 26, 2009, Living Web Workshop,*

Washington, 26 Oct 2009, page 8. CEUR-WS, October 2009. URL https://lirias.kuleuven.be/handle/123456789/253393.

FACULTY OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE
DECLARATIVE LANGUAGES AND ARTIFICIAL INTELLIGENCE
Celestijnenlaan 200A box 2402
B-3001 Heverlee
mathias.verbeke@cs.kuleuven.be
http://people.cs.kuleuven.be/~mathias.verbeke/