From Logic to Language:

Natural Language Generation from Logical Forms

Valerio Basile

The work presented here was carried out under the auspices of the Center for Language and Cognition Groningen of the Faculty of Arts of the University of Groningen.

Grodil Dissertations in Linguistics 142

Document prepared with LaTeXand typeset with pdfLaTeX.

Book cover: Sara Barcena

ISBN: 978-90-367-8248-7 (DRM-free PDF)
ISBN: 978-90-367-8249-4 (Printed book)

From Logic to Language

Natural Language Generation from Logical Forms

PhD Thesis

to obtain the degree of PhD at the
University of Groningen
on the authority of the
Rector Magnificus Prof. E. Sterken
and in accordance with
the decision by the College of Deans.

This thesis will be defended in public on
Thursday 10 December 2015 at 09.00 hours

by

Valerio Basile

born on 20 april 1981
in Napoli, Italy

Supervisor
Prof. J. Bos

Assessment Commitee
Prof. G.J.M. van Noord
Prof. Claire Gardent
Prof. Leo Wanner

Contents

Acknowledgments

I am a lucky man. My personal and professional path has been guided by an invisible hand that steered me towards the right direction when opportunities had arisen. I met Johan Bos as a consequence of a coincidence. In a short time, I was lucky enough to become a PhD candidate under his supervision, and to jump on a series of exciting new projects from their inception. The lucky part was not getting the position *per se* (I earned that), but the privilege of working with Johan, who is without a doubt the best supervisor a PhD candidate can hope for. During these years, the more I talked to fellow PhD students about the University, our research, and our supervisors, the more I felt like the guidance, the support, the teachings and just the plain example I have had from Johan Bos are but an exception.

Many more people have contributed to the development of this book, knowingly or otherwise, with inspiring conversations. I would like to thank especially Prof. Claire Gardent, Prof. Leo Wanner, and Prof. Gertjan van Noord for accepting to be members of the reading committee, and for their feedback.

Since the beginning of my appointment, when my peers learned that I was in the Computational Linguistic group their comments were always around the line of "oh, you're in *that* group, you're so lucky...". Indeed, it takes luck to end up in *that* group, because *that* group is a great group of people and professionals. I would like to thank, in no particular order, the members of my research group, for the many interesting conversations, feedback and challenges (and pubquiz): John, Gertjan, Gosse, Gideon, Barbara, Kostadin, Nynke, Martijn, Çağrı, Daniël, Dörte, Peter, Leonie, Erik, Harm, Johannes, Marjoleine, Simon, Gregory, Dieke, Rob. A special mention is surely deserved for two friends and the best fellow PhDs one could ever desire. Noortje and Kilian, my deepest thanks for having been loyal comrades, invaluable colleagues, and perfect paranimfs.

It took me almost a year to make sense of the organizational units around my research groups. With its overlapping layers and intersecting sets, the complete structure of the reality that sorrounded me at the RUG it still partially a mystery to me. However, in all that mess, many people have been supportive and helpful from the moment I stepped in to the moment I said goodbye. In particular, I want to thank the Graduate School of Humanities and the Center for Language and Cognition Groningen, including all the fellow PhDs with whom we shared similar journeys of joy and sorrow (the latter, mostly): Kashmiri, Aynur, Bregtje, Pavel, Ni, Meriel, the PhD committee and all the people I met at their social events. A heartfelt hello goes to Todd and Marcel and the journalism study group.

Before Groningen, I was lucky enough to live and work in the beautiful Bologna. There, I wasn't only exposed to *tigelle* and *crescentine* (although I still occasionaly have dreams about them), but I also became acquainted with Natural Language Processing, thanks to Fabio Tamburini at the University of Bologna. Had I not taken his class (with great pleasure) and later done my master's thesis under his supervision, I would be probably doing something very different now. The Alma Mater University of Bologna has been a positive influence to my career even after graduation: I want to thank Fabio Vitali and Silvio Peroni at CS and Monica Palmirani and the CIRSFID staff for the inspiring conversations and for keeping my radar from focusing on just one topic. And of course, thumbs up to my best buddies Luca and Francesco. Are you still up to change the destiny?

But a man's life (albeit a lucky one) isn't only work, is it? During my Dutch years I had the luck of meeting a great deal of extraordinary people who gave me so much that I cannot thank them enough. Enyo, Maya, Igor, and Piera. Juan, Romeo, Amalia, Rinse, Rosina, Femke, Lautaro, and Sandra. Frida, Dani, Johan, and Malvina. Gabriel, Rafael, Paco, and Tal. Enej, Artyom, Simon, and Olga. Gabriel, Evaluna, Leo, and Antonella. Federico, Mariella, and Frank. Davide and Lia, Ruggero and Saskia, Tollak and Miranda, Marie, Gretel, Giacomo. To me, you are family.

I am a lucky man, because I met wonderful people in several places. So lucky that I can safely claim I have a fantastic, extended family in Bologna and around the globe. Viola, Nicola, and Linda, Andrea and Alice, Andrea and Gabriella, thank you for being close, selfless, and loving the way you are. Let me break the pattern here, for a very special person. Paola, you are irreplaceable in our lives, thank you for everything you've done. Let's not forget Gruppo Supporto Nerd, behind whose name there are the great friends that helped keeping my sanity intact (mostly failing): Nicola, Pasquale, and Valerio, thank you and see you on the other side. And of course there is my first family, Mom, Dad, Ivano, and Walter without whom none of this would have been possible.

Finally, thank you Sara, for standing by my side, making feel safe and loved every day. It's been a wild ride, are you ready for the next round?

Occorre persuadere molta gente che anche lo studio è un mestiere, e molto faticoso, con un suo speciale tirocinio, oltre che intellettuale, anche muscolare-nervoso: è un processo di adattamento, è un abito acquisito con lo sforzo, la noia e anche la sofferenza.

– Antonio Gramsci

Chapter 1

Introduction

In the beginning of the electronic computation era, when computers (then called *supercomputers*) were as big as office rooms, those gigantic machines typically used punched cards to read the program to execute and the data to set the input and the parameters of the computation. Before they were attached to electronic screens, they used to communicate the results of their elaboration by printing long lists of numbers, characters and symbols on rolls of papers. The human *operator* was responsible of encoding the input and decoding the output in order to make sense of the results.

Soon enough, the technological progress brought us keyboards to type in numbers and commands, monitors to read the results of the running processes in real-time and, later, speakers to hear the sounds produced by the machines and microphones to record the user's voice. The dream of the early computer era was clearly that of a more or less intelligent electronic machine that understands its human counterpart and communicates bidirectionally using their language. The history shows that expecting computers to be able to "sit at the table" and converse with humans in a few years was a naïve idea, and that the field of human-computer communication had a long way to go.

Developers of computer applications have always used natural language in their software to communicate to humans, but the opposite, i.e., human communicating to the computer using natural language, is a way less frequent occurrence. This has to do with the fact that for a computer it is not easy to "understand" the fuzzy, potentially ambiguous natural language (e.g., English) that the user may type in. In fact, the greatest portion, by far, of the research work in the field of Natural Language Processing (NLP) revolves around the analysis of language, rather than its production.

An example of computer programs that understand, to a certain extent, natural language typed in by the user is found among the textual adventures that were popular in the eighties. Those were pieces of text-based interactive fiction where the

```
Welcome to ZORK.
Release 12 / Serial number 990623 / Inform v6.14 Library 6/7
WEST OF HOUSE
This is an open field west of a white house, with a boarded
front door.
There is a small mailbox here.
A rubber mat saying 'Welcome to Zork!' lies by the door.

> look at the mat

Welcome to Zork!

> look inside the mailbox

The mailbox is closed.

> open the mailbox

You open the mailbox, revealing a small leaflet.

> take the leaflet

Taken.
```

Figure 1.1: A snippet of the interactive fiction Zork showing the command line inputs and the program's output.

user (reader? player?) reads a description of the environment on the screen and interacts with the story by providing instructions through a command line interface. Such commands can be simple ones like moving in the direction of one cardinal point, or complex instructions involving the entities found through the story. Figure 1.1 shows an example of an interaction taken from the classic textual adventure *Zork* developed at the Massachussets Institute of Technology.

Notice, from the example, how the interactive fiction software must be able to interpret commands more sophisticated than single words or fixed phrases. Commands like “take the leaflet” for instance have to be analyzed as a verb-determiner-noun structure.

There are many more examples of modern technology that relies on natural language understanding, from the automatic assessment of plagiarism to large-scale systems that monitor customer satisfaction, to the virtual assistants provided by modern smartphones. All the while, the *generation* of proper expressions in natural language is trivial as far as many applications go. The text-based adventure, for example, only has to print out pre-written snippets of text when certain events are triggered. Nevertheless, there are actual applications that need to generate text (or speech) on-the-fly based on a variety of possible inputs, and they are neither few not easy to implement. The field of research that studies these kind of problems and the methodologies for their solution is called Natural Language Generation (NLG), and it will be treated to a great deal of detail in this thesis.

The goal of this introductory chapter is to present the **what** and the **why**:

- **What** is Natural Language Generation, how it is done, what are the input and the output, and what are its restrictions.
- **Why** NLG represents a challenge for the research fields of logic, linguistics and computer science, and what are the motivations to undertake such challenge.

The rest of the thesis, instead, is concerned with the **how**, that is, the methodologies to create a working NLG system with particular focus on an input formalism that allows statistical techniques to be employed. It is structured as follow:

- Chapter 2: overview of the state of the art in the field of Natural Language Generation, with special emphasis on the solutions that make use of large datasets and statistics. This chapter also contains a small survey of working NLG software.
- Chapter 3: here I introduce a novel approach to natural language generation based on statistical models. The architecture of the proposed system is explained, and the chapter goes into the details of the formalism for abstract meaning representation that is the input to the system, and an algorithm that leverages this representation to produce surface forms.
- Chapter 4: the system presented in Chapter 3 is based on supervised learning, that is, its components learn from data. In this case, the source of data is a large collection of textual documents paired with their linguistic and semantic analysis. This chapter presents the work done in order to build such resource.
- Chapter 5: one of the two main modules of the NLG system deals with the prediction of the order of words and phrases in the output. This module and the rationale behind its design are described in this chapter.

- Chapter 6: this chapter describes the problem of the choice of words that express given concepts, and presents two alternative approaches to its solution and a pilot study on the prediction of morphological inflections.
- Chapter 7: in this final chapter the reader will find a series of considerations *a posteriori* on several aspects of the work presented throughout the thesis, along with ideas for future directions of research.

The following sections of this chapter will cover in more depth what the generation of natural language implies, its parallels with automatic machine translation, and why it is important as a research area.

1.1 The Generation of Natural Language

To give a definition of Natural Language Generation is at the same time an easy task and a hard one. On the one hand, every algorithmic process that takes some kind of information as input and produces a natural language expression that describes it is an example of an NLG system. For instance, the function in the software of an electronic cashier machine that dictates that one apple should be written as "1 apple" on the receipt, but any number more than one should be written as "apple**s**", is performing an NLG task. On the other hand, a definition that covers the entirety of the generation process is always forced to include vague terms like the above "some kind of information" to describe what is the input, the starting point of the process itself. The definition given on the "What is Text Generation?" section of the website of the the Special Interest Group on Natural Language Generation of the Association for Computational Linguistics[1] leaves the input completely underspecified:

> The objective of natural language generation (NLG) or text generation systems is to produce coherent natural language texts which satisfy a set of one or more communicative goals.

The generation of natural language is tightly tied to the the **representation** of the information that needs to be generated. Different representation formalisms can have very different characteristics. There is a quote in the field of NLG, usually attributed to Yorick Wilks, that goes approximately like this:

> Natural Language Understanding is like counting from one to infinity.
> Natural Language Generation is like counting from infinity to one.

[1] http://www.siggen.org/nlg.html

The quote, based on an anecdote contained in *Remarks on the Foundations of Mathematics* by Ludwig Wittgenstein, hints at the fact that while in Natural Language Understanding the input is always the same (i.e., text or speech) and the output may be of different natures, in Natural Language Generation the opposite is true. While we want a complete NLG system to produce text (or speech), there is no consensus on what its input should look like.

Some ways of representing data are more or less shallow, such as numeric tables, time series and the like. Others are more structured and can integrate rules and constraints on the data, such as databases or computational ontologies. Another dimension of discrimination between representations is the level of abstraction with respect to the language. A sequence of numeric values describing, for instance, the temperature and atmospheric pressure measured in some location arguably does not carry any linguistic material, while a semantic network made of concepts and relations expressed as nouns and verbs is closer to the natural language form. The ideal representation that could serve as the basis for every NLG-related task does not exists, or at least it has not been uncovered yet. Each problem involving NLG has its own definition and thus demands its specific input representation. Nevertheless, methods developed for one problem can be (and in fact are) transferred across multiple domains and applications.

Regardless of the properties of its input, an NLG system performs one or a series of *transformations* of representations. Such operations aim at modifying the language in which a piece of information is expressed while retaining its *meaning*. Some systems are modular and consists of several component each feeding the next, in a pipeline-like architecture performing separate steps towards the final goal, while others are focused on one type of transformation only. Some systems introduce syntactic and lexical information during the process, some do not, and the list of differences may go on. The next chapter provides an overview of several approaches to NLG known found in the literature.

1.2 Semantics and Predicate Logic

The ultimate goal of a system that does Natural Language Generation is to convey a meaning, simple or complex, through natural language. But how do we describe and represent meaning in the first place?

When children start to learn their (parents') language, they develop associations between the sound of *words* and their sensorial experience. For example, the child hears the word "dog" while looking at the family dog and petting the animal and thus the association is made in the brain (this is of course a gross oversimplification of the real brain chemistry processes). As the child grows, the brain develops more and more and it becomes capable of processing the meaning of full *sentences* rather than isolated words.

We do not know much about the representation of meaning in the human brain. Modern science tells us that the brain operates by sending electric impulses across a network of cells, and researchers are making progresses towards mapping the observed activity in certain regions of the brain to the process of elaboration of certain meaning. While this area of study has enormous importance for all kind of applications, from medical to behavioral research, if we want a *computer* to be able to process language and meaning we need to look at the representation of meaning from a different, more abstract perspective.

Since the ancient times of Greek philosophers (Aristotle *in primis*), scholars have tried to *formalize* the meaning of natural language, so to have unambiguous ways of interpreting it. Since then, the field of *semantics*, that is, the study of meaning, has turned to *logic* in order to have a formal language to describe the meaning of natural language. *Predicate logic*, in particular, has been successfully employed to represent abstract meaning, but also to reason about the meaning itself in a formal way, e.g., by means of deduction. There are many formal systems that fall under the categorization of predicate logic, *first-order logic* being probably the most well known. A simple instance of a first-logic formula looks like this:

$$\forall x : GREEK(x) \rightarrow MORTAL(x)$$

It only takes a high-school level of knowledge of logic to see how such a formula can be used to represent a sentence like "all Greeks are mortal". However, there are many pitfalls to be aware of when dealing with a logic-based representation of the meaning of natural language. For instance, the symbols GREEK and MORTAL in the formula are just symbols, that is, they are not words of the English language, and thus they should be treated accordingly. Later in this thesis more specific formal systems will be introduced that have been designed explicitly to represent the meaning of natural language.

1.3 Natural Language Generation and Machine Translation

Machine Translation is the field that studies and develops computer-based methods to obtain translations of natural language across languages. There are similarities between machine translation and NLG, both in principle and in practice. From a certain perspective, NLG can be seen as a translation task, where instead of translating from a natural language to another one, one translates from an abstract, formal language to a natural one. This section gives a brief overview of how Machine Translation works, focusing on the methods that exploit statistics. and point out its relevance with respect to the field of Natural Language Generation and to the work presented in this thesis in particular.

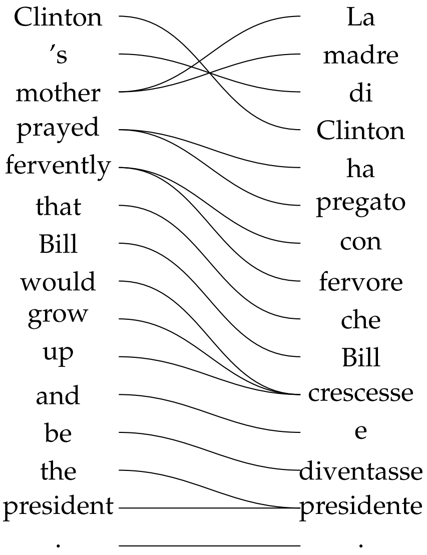
Machine Translation is one of the oldest application of NLP, towards which a gigantic body of work has been done in the past decades. In its beginning, around the middle of the twentieth century, Machine Translation was performed by systems based on translation rules mapping words and phrases from a *source* language to a *target* language. This approach, however, is limited in that the rules have to be compiled by hand, e.g., by expert translators, and a new set of rules is necessary for each new pair of languages/varieties/domains to translate.

Starting from the 1980s, thanks to the increasing computing power available to the researcher, *statistical* methods of Machine Translation started to surface. These methods generally work by automatically extracting the most likely translations of (sequences of) words in context from large amount of *parallel* text, that is, text presented in two or more languages at the same time. In order for parallel text corpora to be exploitable by a Machine Translation algorithm, the parallel versions of the text must be somehow *aligned* at the sentence, phrase or word level. The alignment is actually of crucial importance, being the extra information that a computer system needs in order to learn how to translate the words in different ways, based on the context where they occur. Once a parallel corpus of sufficient size has been aligned, the task of the Statistical Machine Translation algorithm will be that of translating words, phrases and ultimately sentences and documents by looking at various features of the context of the words to translate and leveraging the information learned by the aligned data. Sentence-level alignment of parallel text is the most basic form of alignment, adding a layer that specifies which sentence(s) of the source language text are translated by which sentence(s) in the target language text. Here is an example of sentence alignment of a two-sentence short text in English (on the right) and Italian (on the left):

1. Clinton's mother prayed fervently that Bill would grow up and be president.
2. So far, half of her prayer has been answered.

1. La madre di Clinton ha pregato con fervore che Bill crescesse e diventasse presidente.
2. Finora, metà delle sue preghiere sono state esaurite.

Sentence-level alignment is useful in breaking down the alignment to single, potentially short sentences, but obviously it is too coarse-grained to find an application in learning translation rules. Phrase-level and word-level alignment are a much richer way of annotating a parallel text in order to learn how expressions in the source language translate to the target language based on their context. The first pair of sentences from the example above, for instance, would be word-aligned as follow:



A computer algorithm equipped with linguistic analysis and informed by the alignment would be able to predict that "would grow up" in this context is translated into the Italian inflected verb "crescesse". The context is actually key in this process — a correct translation would not be achieved, in this case, simply by translating word by word "would", "grow" and "up" separately. Aligning word by word a parallel corpus of decent size is a hard and demanding jog though, while sentence alignment, even if it is not always just a one-to-one mapping of sentences, is considerably easier.

But what has Machine Translation to do with Natural Language Generation? If we consider whatever formalism is used as input to a NLG component as a language in itself, then suddenly the generation problem becomes a translation problem, from a formal language expressing some abstract concept into a chosen natural language. As a consequence of this consideration, in the design of a statistical component for generation, some form of alignment between the generation input and the text must be implemented. The problem of this particular kind of alignment is not trivial, because the objects to align are different in nature, and also because the formalism in which the input is specified has to be suitable for a fine-grained word-level alignment.

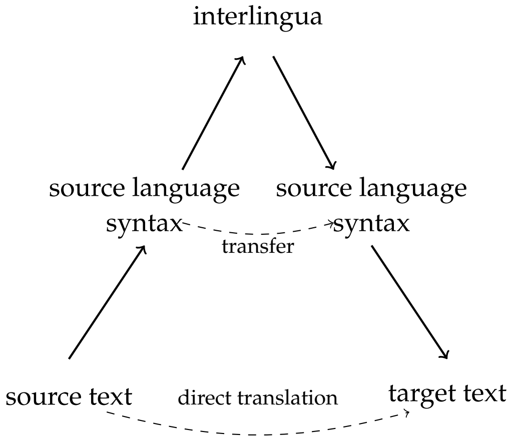


Figure 1.2: Bernard Vauquois' pyramid showing different kind of translations and the relationships between them.

Another way of seeing the relevance of NLG to Machine Translation is to consider the NLG process as part of a bigger process of translation. Figure 1.2 shows the so called "Bernard Vauquois' pyramid", a schematic way to represent different approaches to machine translation.

Following the diagram, *literal* translation is a superficial way to translate a text that relies on words and phrases direct translation from one language to the other. In contrast, *transfer-based* machine translation is the process of analyzing the source text in order to obtain a (language dependent) syntactic representation, then converting it to the analogous representation in the target language, and finally generating the translated text from the syntactic representation. The approach to machine translation that involves the deepest kind of analysis is called *interlingual* Machine Translation and it involves the analysis of the source text to obtain a language-independent, abstract representation that preserves the meaning of the original text. From the interlingua representation then the target text is generated with some NLG technique.

1.4 Why NLG from Logical Forms?

Now that the ideas of natural language generation and logical forms have been introduced, this final section of the introduction has the goal of answering the question: why would one want to use logical forms as a foundation for NLG?

While a straightforward answer is hard to come up with, there are several advantages in the integration of formal logic into an NLG system as its meaning representation formalism. First, logical forms are agnostic with respect to natural languages. As hinted at in the previous section, the same abstract meaning representation can, in principle, be translated into different languages, while two sentences who are one the translation of the other should be represented by the same logical formula.

Logical forms are also *formal* by definition, which means that the reasoning devices of logic can be applied in order to derive new information from existing logical forms. By being formal, logical forms are also directly tractable with *computational* methods. The modular system which is the main topic of this thesis is in fact one example of a computational method (or a set thereof) to generate natural language expressions that would not be possible if its input would have been different in nature.

1.5 Research Questions

This chapter introduced the problem of natural language generation from logical forms, setting the stage for the work that is presented in the rest of the thesis. Before going further, let us make clear the directions that this thesis will take, by formulating a series of research questions that will be answered throughout the thesis itself:

1. What logical formalism can represent the meaning of natural language expressions in a way that facilitates fine-grained alignment with the surface form?

2. How can we produce natural language from a logical form, provided that its alignment with the surface form is known?

3. Given an arbitrary logical form, what methods are the most effective at predicting the alignment with the surface form?

4. In the case of supervised statistical methods for natural language generation, resources like annotated corpora are needed. What characteristics should such a resource have, and how is it possible to build one that is rich enough to be employed to train statistical models?

These research questions drive the entirety of the work presented in this thesis. However, single chapters tend to focus on specific questions. In particular, the questions 1 and 2 are addressed in Chapter 3, question 3 is extensively treated in Chapters 5 and 6, and question 4 is the main topic of Chapter 4.

Chapter 2

Related Work

The field of Natural Language Generation is not a novelty in the landscapes of Computer Science and Information Technology. In fact, the first examples of NLG modules stem from early machine translation projects in the 50s and 60s. Despite the common origin, NLG as a field has not progressed at the same pace as Natural Language Understanding, with the result that the vast majority of work presented at modern conferences and journals in NLP focuses on the latter. Nevertheless, the field of NLG has continued to grow steadily over the years, and has become vastly diverse with its international community working on all kinds of sub-topics. The NLG special interest group of the Association for Computational Linguistics (SIGGEN) brings together the NLG experts worldwide and organizes yearly international meetings and workshops.

By reviewing the literature, three macro-approaches to NLG emerge. First, the *NLG as translation* approach that reduces the NLG problem to finding a suitable map between an abstract language, i.e., some formal meaning representation, and the natural language. This approach may involve defining or learning a bidirectional grammar to form the basis for both language understanding (in one verse) and generation (in the opposite verse). The second approach, or better a family of heterogeneous approaches, is *engineering* one's way through the NLG process. This way of doing NLG is popular in those cases where the problem is application-driven and where specific sub-tasks are considered, i.e., generating referring expressions or surface realization, rather than the full process that goes from, e.g., a logical form to its natural language expression. Finally, a set of alternative approaches cast the NLG problem as a *choice* problem, specifically as a "functionally motivated choice" between the many possible realizations of an abstract meaning representation van Deemter [2009], Bateman [1997a].

The purpose of this chapter is to give an overview of the state of the art of NLG at the moment of this writing, to give an impression of what are the challenges that need to be addressed by a generation method like the one put forward by this thesis. Of course it would be overly ambitious to try to be complete, so a few limitations are needed. Particular attention is given to statistical approaches, mainly because being data-driven is one of the characteristics of the work presented in this thesis. This chapter is also limited to general approaches to NLG and methods that are interesting for the process as a whole or significant components of the NLG pipeline, while work related to more specific topics is reviewed in the respective relevant chapters. Finally, at the end of the chapter, I review actual software packages produced by the NLG scientific community from the user perspective.

2.1 The NLG Pipeline

Many modern articles begin by referring to a Natural Language Generation pipeline, a chain of modules each solving a specific sub-task and passing its output to the next module. This type of architecture, introduced by Reiter and Dale [2000], has the merit of proposing “standard” modules of an ideal NLG system, but its contribution is especially valuable because it identifies the actual problems that needs to be solved for the generation process. This also explains the success of this model, where the slot corresponding with different sub-tasks can be filled by different systems while the general pipeline architecture is retained. In fact, in Reiter and Dale [2000] the authors provide several examples of such modules eg. for the sub-task of Surface Realization.

At a high level of abstraction, the **document planning** module is responsible to decide *what to say*, that is, determining what information is relevant to perform the NLG task at hand. For instance, in a system that answers queries about airplane transport, it is the document planner’s duty to pick the right flights, days, times, etc. that will constitute the material for the generation. The **microplanning** component is roughly responsible for *how to say* the final utterance, by taking care of problems such as lexicalization, aggregation and the generation of referring expressions. The output of the microplanner is an abstract structure containing all the information needed to produce the surface form, including the content words. The final component, is responsible for the **surface realization**, the task of *producing the text* by transforming the abstract representation provided by the microplanner into actual text for the system’s users.

While the naming and scope of individual components may vary, the general structure is fixed, at least in terms of the division of the NLG process into sub-tasks. However, the specifications for the input and output of each module can and do

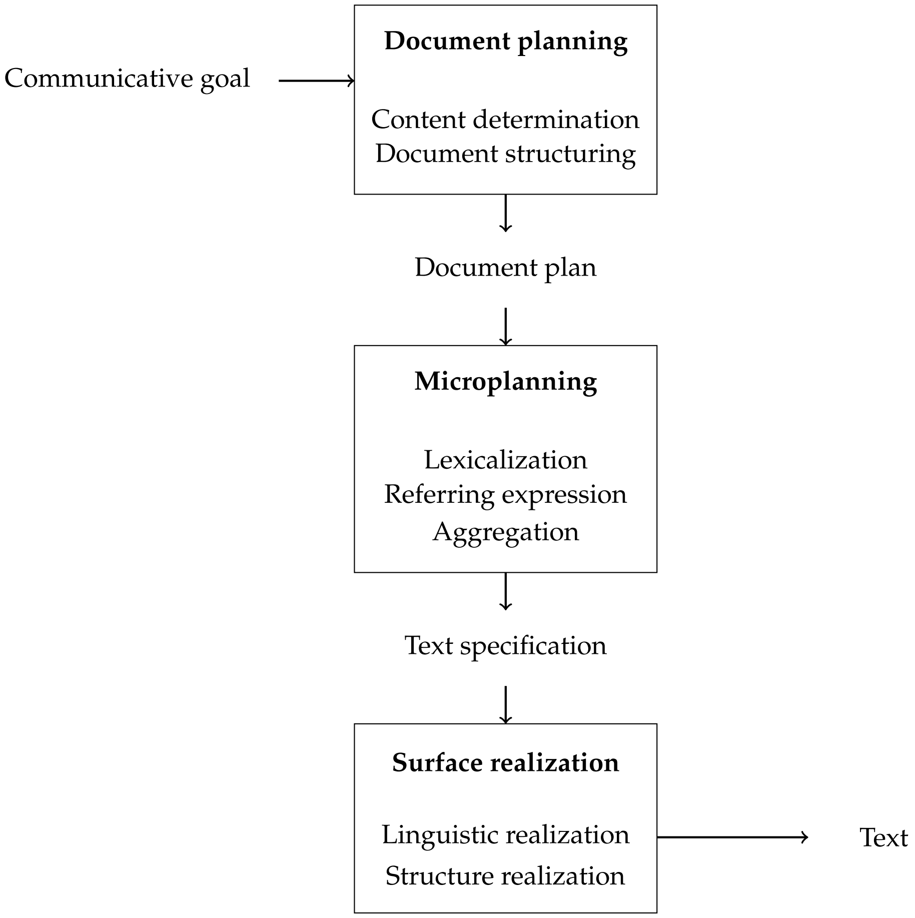


Figure 2.1: The NLG pipeline proposed by Reiter and Dale [2000].

vary from one system to another, a reason why most existing systems are standalone and not easily interoperable. A depiction of the NLG pipeline architecture proposed by Reiter and Dale [2000] is given in Figure 2.1.

2.1.1 Document Planning

The document planner encapsulates the functionalities for retrieving the information that needs to be produced in natural language, and organizing the information in some kind of structured format. The latter function is especially important when the extent of the communication goal exceeds what can be conveyed by simple, short sentences, that is, when some sort of *discourse* information is needed.

Arguably, the document planning module is not necessarily a component of an NLG pipeline *per se*, as apart from general principles, its way of working may be drastically different based on the target application.

The output of the document planner is a *document plan*, that is, a structure that contains the information to be communicated in an organized way, e.g., a tree. The content of the document plan results from the choices made by the module on matters such as isolating the relevant information and deciding the right level of specificity. As an example, a document plan for a system that generates weather reports based on tabular data might contain information items about temperature and rainfall levels in the past 24 hours, while the traffic congestion or the temperature of the past six months should be left out.

The document plan cannot be realized linguistically in its form, because even if such mapping could be made it would result in a very clumsy, computer-like generation. For this reason the plan is fed to the next module on the pipeline, the microplanner.

In this thesis, generation from a certain kind of logical forms is targeted, thus the decision process proper of the document planning is considered as already given to the NLG system. However, while the focus is not on the document planning in itself, it is important to notice that the format of the output of the planner, that serves as input for the next module, is crucial to the rest of the process. In other words, choosing the right formalism to encode a document plan will affect the capabilities of the NLG system.

In Chapter 3 a specific formalism is proposed as the basis for the NLG system object of this thesis, that can be compared in a way to the output of a document planning stage.

2.1.2 Microplanning

To produce a linguistically sound surface form from the document plan, a series of non trivial problems must be solved. In the standard pipeline of Reiter and Dale [2000], three tasks at this level are taken care of by the microplanning module: lexicalization, generating referring expressions, and aggregation.

Lexicalization — sometimes referred to as “lexical choice”, depending on how the problem is approached — is the problem of choosing the (content) words that best express the concepts to communicate, depending on the context of the communication. Even considering just single concepts to be mapped to single expressions, lexicalization is crucial to provide fluent realizations. Consider for instance the difference between the two sentences:

1. Economic activity is limited to providing services to the military.
2. Economic activity is circumscribed to catering services to the war machine.

While the sentences are similar, and both are comprehensible, they certainly show a difference in terms of fluency to the human reader. Lexicalization is difficult because often the choice of words has to be informed by extra-linguistic information such as the identity of the parts involved in the communication process, or pragmatics. Moreover, one needs more than a direct mapping between content words and concepts. Consider for instance the following two realizations of a possessive construction:

1. the car owned by Mary
2. Mary's car

While at a logical level the two expressions have the same meaning, (1) stresses the ownership property more than the neutral (2). In (1) the word "owned" maps to some concept of property while in (2) this concept is expressed by the particle "'s". Arguably, the former is a content word in its own right, while the latter is not.

The system presented in this thesis comprises a module to solve the lexicalization task by choosing from given sets of words associated to the concepts the most appropriate. The lexicalization problem is further explored in Chapter 6, where different methods are proposed that fits into the general architecture for NLG proposed in this thesis.

Another problem solved in the microplanning phase is the *generation of referring expressions*. There are many ways of introducing concepts in a discourse, and not all of them are equally effective, so a system must pay attention to how entities are referred to, both the first time they are introduced and the subsequent times. For instance, the first time the entity *Valerio* is introduced in his autobiography it could be referred to by first name, or full name including titles, or "A PhD candidate", and so on, depending on the nature of the writing and the audience of the message. Likewise, the following times the writer refers to *Valerio*, a personal pronoun will be likely used, unless other entities have been introduced that makes the use of pronouns ambiguous.

Lastly, *aggregation* is the process of transforming the abstract structures to make them closer to natural-sounding expressions, by putting together elements that can be communicated in a single clause. For example, the following two sentences, taken from the output of a weather forecast system, show much repetition: "The month was cooler than average. The month was drier than average." Even though technically correct, it appears immediately clear that this is the output of an automatic system. By means of aggregation, the system could instead express the same meaning with "The month was cooler and dried than average", which sounds more natural. When done effectively, aggregation can significantly enhance the quality of the final surface realization.

The output of the microplanner is a *text specification*, a structured container for the words and phrases to generate (again, this can be a tree), which abstracts away from grammatical and morphological aspects. The text specification, in turn, is fed to the last module of the pipeline, the surface realizer.

2.1.3 Surface Realization

With a rich microplanning module as the one proposed by Reiter and Dale [2000], the final step of the NLG process has only a few tasks left to take care of in order to produce the final output of the system.

The morphological aspects in the text specification could be underspecified. If that is the case, the surface realizer is responsible to predict morphology for the surface form. In some cases, the input to the surface realization module contains more abstract structures such as syntax specifications or lexicalized case frames, which are typically harder to realize.

Other than that, the surface realizer is also responsible for the presentation layer, that is, to produce a usable output that can be consumed by users. This is not trivial when the expected output is not simple raw text but rather has some structure, e.g. HTML pages, PDF files, etc.

The system presented in this thesis is characterized by a fuzzier distinction between the microplanner and the surface realizer. Rather, it tries to map abstract meaning representations to the text *directly*, tackling word order prediction, lexical choice and morphology realization in parallel. The architecture of the NLG pipeline employed by such system is presented in detail later in the thesis, in Chapter 3.

2.2 Statistical Approaches to NLG tasks

Just as in Natural Language Analysis the development of statistical, data-driven methods has led to improvements of the performance of computer systems on a wide variety of tasks, recent developments in the same direction are pushing forward the capabilities of NLG systems. In the following section, a review of statistical approaches to several problems related to NLG is given.

Given the modularity of the pipeline architecture, it is not surprising that many scholars have opted for tackling one NLG-related task at a time, while others have try to solve the problem at once. An example of the latter group is the work of Konstas and Lapata [2012], where content selection and surface realization are performed jointly by means of data-driven methods. Another aspect highlighted by the aforementioned paper is that the authors cast the problem explicitly in terms of learning a PCFG grammar, in their specific case one that maps the structure of a database holding the domain knowledge to the natural language expressions to realize it. Section 2.5, later in this chapter, contains further hints at the parallel between NLG and grammar learning.

Zhou et al. [2002] employ statistical methods to produce natural language expressions from semantic representations that act as interlingua in a transfer-based machine translation system. The authors use a manually annotated corpus to train a maximum-entropy model. The NLG component then applies the statistical model together with a dictionary of phrase translations to produce the output surface form.

Statistical methods have also been used to approach NLG tasks located earlier in the NLG pipeline. Duboue and McKeown [2003], for instance, developed a two-stage approach to content selection based on statistical techniques. Their method employs clustering to derive content selection rules for the purpose of automatic generation of biographies.

Bohnet et al. [2011b] developed a statistical system to generate sentences from semantic structures. Here, the input to the NLG process is a semantic representation based on the CoNNL syntactic annotation, with nodes corresponding to function words removed and other modifications made in order to abstract from the syntactic specificity of the natural language. The system of Bohnet et al. [2011b] is rooted in the framework of the Meaning Text Theory [Mel'cuk, 1988], a linguistic theory that models formally all the steps from semantics to phonetics, and consists of a pipeline of decoders that transform semantic structures into *deep syntactic* structures, then into syntactic parse trees, and finally into a linearized surface string — lexical choice is left out of the process. In a similar line of work, Ballesteros et al.

[2014] propose two approaches based on Support Vector Machine (SVM) classifiers to map semantic and syntactic structures, overcoming the problem of their inherent non-isomorphy. This method also works by constructing a pipeline of processes, essentially the inverse of a typical NLP analysis pipeline, made of subsequent steps each informing the following one.

An alternative approach to the pipeline architecture is that of Mairesse et al. [2010], which aims at learning the mapping between meaning and text directly, from a data set made of text annotated with *stack-based* meaning representations. The Bagel system developed by the authors employs Bayesian networks to learn from text aligned with semantic representations akin to linearized semantic trees.

2.3 Generation from Knowledge Bases

The process of generating text from formally defined databases of information has been studied in the past, although in the majority of the cases with specific applications in mind.

Bouayad-Agha et al. [2012b] propose a layered ontology architecture as a basis for generation made of three RDF/OWL computational ontologies, where the *domain* ontology and the *domain communication ontology* depend on the generation task at hand, while the *communication* ontology provides the general concept needed by the document planner to produce a text plan. While this approach is an improvement over using a single domain-specific knowledge base or a template-based approach, it is still partly dependent on the application of the NLG pipeline, or at least on its domain.

RDF/OWL is an XML based formalism to define computational ontologies. Despite its many interesting properties, such as being modular and promoting interoperability, RDF/OWL is not ubiquitous in the NLG area. Interestingly, there exists a mapping between RDF/OWL and Discourse Representation Structures proposed by Presutti et al. [2012] that helps to fill the gap between broadly used knowledge bases and generation from deep semantic structures.

Recent work from Gyawali and Gardent [2014] combines the generation from a knowledge base with a statistical approach. The authors derive a *tree adjoining grammar* from annotated data, and then use the grammar to generate text for new instances from the knowledge base, with particular measures to account for unseen items to lexicalize. The methods works by learning a pre-made alignment of the knowledge representations with the corresponding surface forms at the event and entity level.

While RDF/OWL ontologies are a popular format to encode domain and general knowledge, there are alternatives. It is debated, for instance, whether one should consider WordNet [Miller, 1995] a computational ontology in the proper sense. Regardless of its definition, WordNet has been used both as a standalone knowledge base and as a mean to augment existing RDF/OWL ontologies [Lin and Sandkuhl, 2008].

WordNet has been rarely been applied to NLG, despite its wide application in other NLP fields, such as, for instance, Word Sense Disambiguation (WSD). A notable exception is the work by Jing [1998] who proposes WordNet-based methods to address specific NLG tasks, in particular lexicalization and paraphrasing.

The author shows that the open-domain nature of WordNet makes it a robust knowledge base to support generation in open-domain scenarios, but also that it can also used in combination with other knowledge bases, i.e., to adapt to a particular domain. Later in this thesis (Chapter 6) this claim is tested with the implementation of a novel method for lexicalization that incorporates WordNet as linguistic knowledge base to provide natural sounding generations.

2.4 Surface Order

If grounding an NLG system to some form of knowledge representation database is fundamental for having a repository of words and phrases to use, the other side of the coin is that at some point this lexical material has to be laid down in a linear for that requires a precise order of its constituents. This task is often referred to as *linearization*.

In many cases, the order of the words in a sentence is determined by constraints defined in the target language's grammar, although there might be other factors. In some languages, for instance, different orderings of the same words are used to convey different communication intents, that is, sentences with different word order each convey its specific semantics — this is the case in some Eastern European languages such as Czech. These considerations bring up the question of what determines the word order of a realization, given the semantic content to be realized. Korbayova et al. [2002] have exploited *systemic ordering*, a language-specific concept imposing an order on the types of possible complements (actor, patient and so on), to control the word order of their proposed NLG pipeline. In this thesis, the problem of word order is tackled at an early point of the generation process, i.e., by the sentence planning module.

Not all languages are on the same level in terms of freedom of word order. German, for instance, is known to be a relatively free word order language, which in turn increase the difficulty of the task of ordering words and constituents in generation. Bohnet [2007] developed a technique to induce grammars for linearization using rule patterns on the *topological models* of several languages, in an effort to derive a language-independent method for predicting word order.

An alternative approach is that of generating all the possible orderings and then let a further elaboration step rank them in order to select the most appropriate. While this approach may work to different extent based on the target language, as mentioned earlier, it is in practice an effective one and potentially straightforward to implement. Most modern systems implement this approach by imposing a statistical language model over a selection of possible realization [Langkilde and Knight, 1998], ranking them by how their word order matches that of observed natural language, thus enforcing syntactic and semantic constraints only in an implicit way.

2.5 Logic-based Representations for NLG

So far we have looked at Natural Language Generation from a rather engineering-oriented point of view, that is, as a collection of techniques and architectures geared towards the solution of a series of sub-problems in order to translate some abstract representation of information into a human-readable text. Looking at the generation of text from a more theoretical perspective, one can see it as the problem of mapping abstract meaning representations to linguistic surface representations. This thesis, in particular, focuses on a formalism for generation that is essentially grounded in formal logic.

Neither the use of logical formulas to encode the meaning of natural language nor the generation of natural language expressions from logical forms are completely new ideas. Already in the eighties, tin Wang [1980] was designing a computer system capable of generating sentences and questions from logical formulas, based on the generation of intermediate template-like representation called *semiotic interpretations*. Other authors, later in the nineties, also propose to employ logical forms as the basis for natural language generation, e.g., Phillips [1993]. More recently, Coppock and Baxter [2010] employed logical forms as the formalism for meaning representation in the NLG component of Cyc [Lenat, 1995], a large knowl-

Table 2.1: Three sentences with equivalent meaning and their respective logical forms. Example from Shieber [1993].

item	String *Canonical Logical Form*
(i)	John threw a large red ball. $\exists x.throw(j, x) \wedge large(x) \wedge red(x) \wedge ball(x)$
(ii)	John threw a red ball that is large. $\exists x.throw(j, x) \wedge red(x) \wedge ball(x) \wedge large(x)$
(iii)	John threw a large ball that is red. $\exists x.throw(j, x) \wedge large(x) \wedge ball(x) \wedge red(x)$

edge base of *common sense*. The component performs a series of transformation operations on the input logical form, such as the extraction of *clausal skeletons*, to facilitate its mapping to natural language expressions. While becoming more refined over time, all the cited systems are based on rules, which makes them inevitably not scalable and language dependent.

When the abstract representations are based on some kind of formal logic, a problem arises called the *problem of logical-form equivalence*. Shieber [1993] formulates the problem using first-order logic, and shows how different, but equivalent, logical forms generate different sentences with the same meaning, while the mappings are not easily interchangeable. For instance, the three logical forms in Table 2.1 are *logically equivalent*, but a generator would never produce, for instance, the second sentence from the first logical form.

Why is logical-form equivalence a problem? Logical equivalence is, in general, non computable. Even though in some logics, including propositional logic, equivalence is decidable,[1] in any formalism sufficiently expressive to represent the meaning of natural language determining whether two formulas are equivalent is an intractable problem. This is a problem only when analyzing a text to produce a logical formula representing its meaning, because it is impossible to just produce all possible analysis of a sentence and then select the most correct one. When the direction of the process is reversed, that is, when performing generation of text from logical formulas, the logical formula is given as input. Still, the ambiguity of the mapping between meaning representations and surface form makes it hard to solve the generation problem by constructing a grammar to treat generation as a translation problem from logic to natural language.

[1] Decidability of propositional logic can be proved by constructing truth tables. Since in propositional logic every argument is finite, a truth table for a given argument is guaranteed to be finite.

There is no straightforward solution to the problem of logical-form equivalence, although methods have been proposed to address some weaker variants of the problem. However, according to Shieber [1993], it is important to choose the right formalism to represent meaning, one in which the variations in the semantics is mirrored by variations in the natural language syntax. Appelt [1987] also hints at a solution in this direction when analyzing the reversible grammar approach to NLG. From such perspective, the NLG process is the application of a bidirectional grammar in one direction, whereas the analysis of the language is the application of the same formalism in the opposite direction. That is, one grammar for both problems. An interesting examples of such formalisms is the Reversible Stochastic Attribute-Value Grammars proposed by Kok et al., a kind of grammar in which a single statistical model is employed for both parsing and fluency ranking. Among the characteristics of a bidirectional grammar suitable for generation, Appelt [1987] mentions the need to be enough fine-grained to catch in the semantic representation all the linguistic variations. Consider for example the sentence "The resistance of R1 is 500 ohms". In some formalism the meaning of this sentence would be written as the following logical formula:

$$ResistanceOf(R1, ohm(500)) \tag{2.1}$$

While this representation is perfectly fine from the logic perspective, its structure is sensibly far from the original text. In other words, it is difficult to *map* the elements of 2.1 to the words of the sentence "The resistance of R1 is 500 ohms" in a natural way. Now consider the following first-order logic formula equivalent to 2.1:

$$\begin{aligned} &\exists x ResistanceOf(R1, x) \ \ \& \\ &x = ohm(500) \end{aligned} \tag{2.2}$$

While 2.1 and 2.2 are *logically* equivalent, the latter is easier to align to the surface form. The equality predicate in 2.2, for instance, maps naturally to the verb *be*. Here is another example:

$$\begin{array}{l} \exists x1x2x3 \ \& \\ resistance(x1) \ \& \\ named(x2, r1, nam) \ \& \\ of(x1, x2) \ \& \\ ohm(x3) \ \& \\ |x3| = 500 \ \& \\ x1 = x3 \end{array} \tag{2.3}$$

The predicates and operators in 2.3 are even more fine-grained, in terms of expressing the linguistic variation — for instance the binary relation *of* directly maps to the homonym preposition in the original sentence.

Order matters too. While in most logics scrambling the relative order of the predicates of a formula results in an equivalent expression, that information can be used to express different phrasing of the same utterance. The paraphrase "R1'resistance is 500 ohms", for instance, is not easily derived from 2.3 (and neither from 2.1 or 2.2, for that matters), but an equivalent formula can express the same meaning while preserving the information relative to the order of the words:

$$\begin{array}{l} \exists x1x2x3 \ \& \\ named(x2, r1, nam) \ \& \\ resistance(x1) \ \& \\ of(x2, x1) \ \& \\ ohm(x3) \ \& \\ |x3| = 500 \ \& \\ x2 = x3 \end{array} \tag{2.4}$$

The take away message from this sequence of examples is that choosing the right formalism to represent the natural language meaning is important from a generation perspective. Even when the NLG problem is not cast as learning a grammar, the consequences of the problem of logical-form equivalence have an impact on the amount of ambiguity a system will have to deal with, especially when the underlying formalism used for meaning representation lacks the capabilities to express all the dynamics of natural language.

In Chapter 4 I introduce a formalism to encode meaning representations that provides several desirable properties from the generation perspective, and I show how it is capable of performing an alignment between its logical formulas and the correspondent surface form in order to automatically learn such mapping from the data.

2.6 NLG Software

The Association for Computational Linguistics has a Web page with a list of NLG systems available for download.[2] The list has been created on 7th February 2009 and to date the last update has been made on 27th November 2013. In this section I evaluate the current state of the NLG software by going through the aforementioned list item by item, installing each software package and testing it. The tests are conducted on a modern desktop PC running a common distribution of the GNU/Linux operating system (Ubuntu 14.04.1 LTS).

- ASTROGEN is a Prolog program comprising sets of rules to translate lexicalized logical forms into English language. Included in the system there are rules for several kind of aggregation and for surface realization. Tested with SWI-prolog it worked out of the box.

- LKB (Linguistic Knowledge Builder) is a framework for grammar engineering built as part of the DELPH-IN project. Installing and running the software is straightforward following the instructions on the project website, and the GUI makes it easy to experiment with example grammars. The package also contains a surface realization component, not accessible through the user interface but only as a Lisp library. LKB is developed with Minimal Recursion Semantics as its focus semantic formalism, although it supports a variety of grammars.

- MUG workbench is a Web-based tool for the development and testing of Multimodal Functional Unification Grammar composed by a Prolog backend and a Web interface to interact with the models. Following simple instructions the system was up and running in short time, however the generation capabilities are not working, throwing exceptions for every example tested. As a tool to explore and develop grammars, knowledge bases, and models, MUG is user-friendly and has lots of useful features.

[2] http://aclweb.org/aclwiki/index.php?title=Downloadable_NLG_systems

- NaturalOWL is a plugin for Protégé[3], a popular editor for OWL Ontologies. With NaturalOWL it is possible to annotate an ontology with a layer of linguistic information such as word forms for concepts, and use the annotations to produce natural language descriptions based on the original ontology. As a Protégé plugin, NaturalOWL is very easy to install although it requires specific versions of Protégé and the Java virtual machine. Once installed, it is straightforward to activate the plugin, annotate existing ontologies, and produce automatically generated text previews.

- SimpleNLG is a Java library for surface realization. Following the provided instruction the library is easy to integrate with the Eclipse IDE[4] and get up and running. The website also contains a tutorial with which the user can get the hang of the basic functionalities in a few minutes. The library provides a large number of features to transform lexical items enriched with their properties into fully realized sentences, including custom lexicons, modifiers, prepositional phrases, multiple clauses, and more.

Some packages could not be tested for a variety of reasons.

- CLINT: template system augmented with a noun-phrase generator; reasons: 1) not available for download (broken FTP link), 2) only available for the MS Windows operating systems.

- CRISP [Koller and Stone*, 2007]: a system for microplanning and sentence realization that treats the problem in the AI planning perspective; reason: unable to build.

- DAYDREAMER: a "computer model of the stream of thought"; reason: broken link.

- FUF, SURGE, SURGE 2.3, SURG-SP, SURG-IT: a large grammar of English (SURGE) built on top of FUF, a CLisp interpreter for a functional unification formalism, and their respective variations for Spanish (SURG-SP) and Italian (SURGIT); reason: broken links.

- GenI: Haskell-based surface realization module that uses Tree Adjoining Grammar; reason: unable to build following the instructions on the website.

[3] http://protege.stanford.edu/

[4] http://eclipse.org/downloads/

- Grammar Explorer: a tool to explore large-scale systemic-functional grammars, including KPML grammars, for coverage and other features; reason: only available for MS Windows operating systems.

- KPML: framework for grammar engineering tailored for NLG; reason: only available for MS Windows operating systems.

- NLGen: component for sentence generation based on probabilistic inference that generates from the structures produced by the RelEx dependency relationship extractor; reason: there were files missing in the software distribution. A video shows a demo of a virtual pet answering few simple questions with generated short sentences.

- NLGen2: like NLGen it uses the RelEx dependencies as input, but it employs a different sentence generation strategy from NLGen; reason: could not run the software, no valid executable was found.

- OpenCCG: library for parsing and realization for Combinatory Categorial Grammar; reason: impossible to run the system following the instructions in the documentation.

- RAGS (Reference Architecture for Generation Systems): collection of packages including a genetic algorithm based text planner and a wrapper for FUF/SURGE; reason: broken link.

- SPUD: software for sentence planning and surface realization based on a lexicalized tree-adjoint grammar; reason: unable to run the software with the provided instructions.

- STANDUP: a tool for generating word-based jokes with the purpose of aiding non-speaking children to communicate. Reason: unable to run the software due to errors. However, the website of the project provides an excellent online interface to the program, complete with the explanation of the generation process.

- Suregen: system for generating text for clinical medicine from an ontology-based formalism. It employs NLG techniques as part of the interactive user interface. Reason: only a demo for MS Windows operating systems is available for download, although a screencast video shows an example of the software in action.

- TG/2: hybrid component for surface realization that supports context-free grammars as well as templates and canned text. Reason: broken link.

Of course, the fact that some of the reviewed software packages were not working properly *out of the box* means nothing with respect to the scientific value of their respective underlying ideas. Ideas from these systems are sometimes implemented in commercial software distributed by companies such as Arria[5] and Yseop[6].

2.7 Conclusion

This chapter have shown that the field of Natural Language Generation is a variegated galaxy of tasks, methods, theoretical frameworks, and applications. While many researchers stick to a *de facto* standard architecture while building single components or entire NLG systems, others have experimented with alternative architectures to map abstract representations of meaning to natural language expressions. Moreover, some are interested in particular applications, while others have looked at the NLG problem from the theoretical perspective of being the image in the mirror of Natural Language Understanding.

Statistical approaches are relatively new in the panorama of NLG, which is unsurprising, considering that such methods typically require language resources that are expensive and hard to obtain. The previous work discussed in 2.2 and the work presented in this thesis indicate that the use of statistical methods is indeed promising for the future developments in NLG.

Sections 2.3 and 2.4 refer to existing work done on specific aspects of NLG, namely the lexicalization of concepts from formal knowledge bases and the prediction of surface order (treated respectively in this thesis in Chapters 6 and 5. For both problems, valid and relevant literature was found. However, in both cases the solutions proposed in literature (and this thesis is no exception) are somewhat tied to the input representation and its level of abstraction.

As underlined by Evans et al. [2002] when answering the question “What is NLG?”, the definition of NLG is asymmetrical. While it is easy to agree on what the output of an NLG component should look like (i.e., natural language text or speech) there is no agreement on the format and the characteristics of its input. With this consideration in mind, it is not surprising that so many different approaches to NLG and its sub-tasks have been proposed over the years, as different types of input require different ways of processing. As such, the formalism used for the representation of the input to an NLG system is just as important as its architecture.

[5] `http://www.arria.com/`

[6] `http://yseop.com/EN`

Finally, the small survey of NLG software presented in Section 2.6 shows that there exist robust free software packages to build NLG systems both for practical applications and experimental work. Unfortunately, a good deal of existing software is not ready for the general public yet (probably some of the packages were not meant for non-academic users at all).

This chapter concludes the introductory part of this thesis, where the information presented to the reader has been distilled from previously existing work. The next chapters, instead, will contain new material, including theoretical considerations and experimental evidence.

Chapter 3

Generation of Text from Aligned Logical Forms

In the past chapter we have seen what is the state of the art of Natural Language Generation, in particular the solutions based on statistical methods. In this chapter I introduce a novel solution to the problem of generating natural language expressions, in particular from formal representations of meaning. This is also the first chapter where new material is presented, in the form of a modular system for NLG based on the idea of aligning meaning representations and text in order to train a supervised statistical model. Such model is then used to predict the text aligned to unlabeled input meaning representations and ultimately construct natural language utterances to express them.

This chapter presents an overview of the system, then goes into the details of the meaning representation, the alignment procedure and how to create a complete text out of partial surface forms corresponding to single entities. The central components of the system, responsible to construct the text alignment, need more space to be described in details, and for this reason they are covered by the Chapters 5 and 6, while Chapter 4 presents a linguistic resource collected in order to leverage statistical methods.

The goal of this chapter is threefold:

- To introduce a logic formalism capable of representing the meaning of natural language expressions. These abstract meaning representations can be aligned to the natural language with a word-based granularity.
- To present a modular system that, given an arbitrary meaning representation written in the formalism mentioned in the previous item, produces its alignment with the surface structure.
- To show how, with such an alignment in place, the final surface form can be

constructed in a deterministic way by composing partial surface forms corresponding to the entities in the meaning representation.

In the rest of this chapter, I will present the plan for a novel system that generates a natural language string to express the meaning of a logical formula. The system is based on a statistical, supervised approach, and is made of several components which are described in detail in the following sections of this chapter.

Before a panoramic of the entire system is given, it is important to grasp all the details of the meaning representation formalism on which the system is based on, in particular its capabilities of being able to represent the meaning of arbitrary open-domain natural language, as well as being aligned to the surface structure word by word. The formalism is introduced and described in the next two sections. [1]

3.1 Introduction

Some of the approaches found in literature work very well for the particular applications they focus on, but they are not intended as general-purpose NLG solutions. This is not meant to be a critique, as these systems are designed to be application-oriented. Other methods are more general and aimed at generation of open domains, however the abstract representations they employ as their input are too close to syntactic forms and far from general semantic representations. This can be a problem when incorporating such NLG components in contexts other than NLP, for instance as part of a larger pipeline that also includes some kind of knowledge representation and reasoning modules. In such a situation, it is desirable to work with representations that abstract away from linguistic information like syntactic structure. There are however cases when retaining some syntax structure helps, such as transfer-based machine translation.

From an applicative point of view, the typical place of NLG in a larger system is that of a translator from data to text or speech. The problem is of course in the word "data" and its vagueness. Whether the data to be expressed come from weather forecasts, medical sensors, GPS coordinates of birds, or football matches, they describe **events** happening that involve **entities**. Entities often have attributes, and they must be realized in a way that sounds natural, using the correct referring expressions, e.g., pronouns vs. repetition of explicit mentions. A single abstract language to describe the knowledge to be generated would serve as a pivot for the

[1] This chapter is based on the work published in Basile and Bos [2011] and Basile and Bos [2013a], and on [Basile and Bos, 2013b, unpublished].

Table 3.1: Example of data in tabular format: football match Italy-Germany 17 June 1970

Minute	**Player**	**Team**
8	Boninsegna	Italy
90	Schnellinger	Germany
94	Müller	Germany
98	Burgnich	Italy
104	Riva	Italy
110	Müller	Germany
111	Rivera	Italy

many different data representation formats, pushing forward the development of open-domain application-oriented NLG systems. Table 3.1 shows a simple example of data in tabular format, describing a football match, one score per row. There are many ways of describing this data using natural language, for instance "Boninsegna (Italy) scores at the eight minute, Müller (Germany) scores at 94′ and 110′, ..." or "Italy wins 4-3". The records in the table describe an homogeneous series of events (goal scores) involving entities like players and minutes.

entity(BONINSEGNA, player)	event(GOL, 8, BONINSEGNA)
entity(SCHNELLINGER, player)	event(GOL, 90, SCHNELLINGER)
entity(MULLER, player)	event(GOL, 94, MULLER)
entity(BURGNICH, player)	event(GOL, 98, BURGNICH)
entity(RIVA, player)	event(GOL, 104, RIVA)
entity(MULLER, player)	event(GOL, 110, MULLER)
entity(RIVERA, player)	event(GOL, 111, RIVERA)
entity(ITALY, team)	event(WIN, ITALY)
entity(GERMANY, team)	

Figure 3.1: Example of data in terms of events and entities: football match Italy-Germany 17 June 1970

A possible alternative to the tabular format based on a formalization of the concepts of event and entity is given in Figure 3.1. This format, albeit retaining the same amount of information of the alternative one, is more generic, that is, capable of encoding different kinds of information without recurring to specialized predicates and rules.

Finally, an ideal representation of information should be **formal** so that machines and automatic systems can manipulate it in a systematic way, for instance by means of software that parses such representations and extract information from them. For instance, with access to some external source of knowledge (encoding the rules of football, in this case), a computer can infer the last event in the example (the information that one of the two teams wins, encoded as event(WIN, ITALY)) from the previous ones.

3.2 General Ideas

The statistical NLG method of the system proposed in this thesis is based on the idea of an alignment between the abstract representation of meaning and the text that expresses such meaning. The alignment must comply with certain characteristics, for example a sufficiently fine granularity must be assured. Apart from that, different meaning representations call for different strategies of alignment. While the alignment of logic with text is a more general process than just a part of the system pipeline, it is essential to explain how it is done and the choices that have been made, as they actively affect the way the alignment is learned by the system. This section is therefore a necessary digression to introduce the idea of alignment between text and logic, before delving into the internals of the architecture of the NLG system.

Several different formal semantic representations have been proposed in the literature, such as Minimal Recursion Semantics [Copestake et al., 2005], Abstract Meaning Representation [Banarescu et al., 2013], the *deep syntactic structures* from Meaning Text Theory [Mel'cuk, 1988] and their transformation in *semantic structures* proposed by [Bohnet et al., 2011b]. Although they might differ in various aspects, they also have a lot in common. Many semantic representations are variants of first-order logic and share basic building blocks such as entities, properties, and relations, complemented with quantifiers, negation and further scope operators. In other words, those representations encode the *logical form* of natural language. A simple snippet of a formal meaning representation based on first order logic is the following, where symbols are used to abstract away from actual words:

$$\exists x(\textsc{blue}(x) \wedge \textsc{cup}(x))$$

How could this logical form be expressed in natural language? Literally, its translation looks like "there exists something that is blue and that is a cup". To put it differently, how could we realize the variable x in text? As simple as it is, x describes “a blue cup”, or if your target language is Italian, “una tazza blu”, or variants like “there is a blue cup”, or perhaps as “una tazza azzurra”, using a different adjective to express blueness. Moving away from simple examples, what is needed is a systematic way of translating the logic predicates into natural language expressions that is able to scale up to larger and more complex semantic representations.

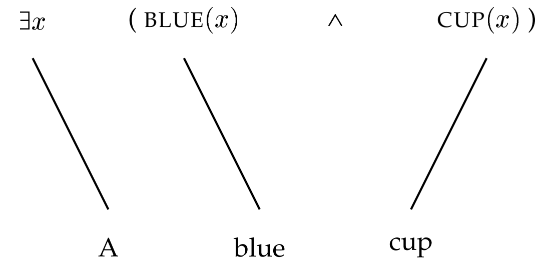


Figure 3.2: The surface form “a blue cup” word-aligned to the logical form representing its meaning.

As pointed out already in this thesis, NLG can be viewed as a machine translation task, but unlike translating from one natural language into another, the task is here to translate a formal (unambiguous) language into a natural language like English or Italian. Current statistical MT techniques are based on large parallel corpora of source and target text aligned at the word of phrase level. Similarly, statistical NLG is based on exploiting a collection of text aligned with logical forms representing its meaning. Figure 3.2 shows an example of aligning the logical form of the example above with the text “a blue cup”.

Even from this simple example it can be noted how decisions have to be made in the alignment strategy. For instance, here the determiner is attached to the existential quantifier and the introduction of the variable x, while a different strategy could be of aligning the phrase “a cup” to the predicate CUP(x).

In this section I introduce a method for precise alignment of formal semantic representations, with certain characteristics, and text. Once we have a good method to do that, the next step is the creation of a large corpus to open the way for statistical approaches to NLG, in the same vein as those used in machine translation. A resource of this kind is described in Chapter 4.

There are different ways of alignment semantic representations and surface strings, each with its advantages and drawbacks. The simplest strategy, but also the least informative, is to align the semantic representation with a sentence or complete text without further information on which part of the representation produces what part of the surface form. This might be enough to develop statistical NLG systems for a small set of short sentences, but it does not scale up to handle larger texts, because the learning system would quickly hit the data sparseness problem. If we want a machine learning approach that is able to predict the alignment of unseen sentences, then the compositional aspects of natural language should be modeled somehow, rather than rigidly learn entire sentences with their respective semantic representations. To do that, one could devise more complex schemes that allow for a more fine-grained alignment between parts of the semantic representation and surface strings (words and phrases). Here there are two routes to follow, namely the *minimal* and *maximal alignment*.

Minimal alignment is the method where every word in the surface string points to one element of the semantic representation. This alignment method forms a good starting point for the development of a statistical NLG component because the learning system does not have to deal with the redundancy of links between words and logical forms. The alignment in Figure 3.2 is an example of minimal alignment.

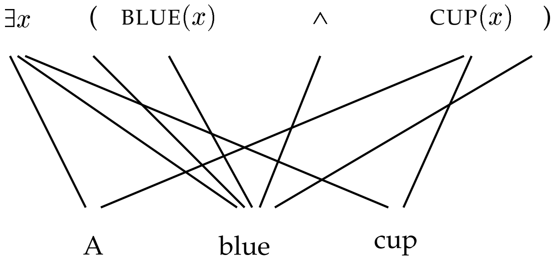


Figure 3.3: The surface form "a blue cup" word-aligned to the logical form representing its meaning using maximal alignment.

In maximal alignment, each single piece of the semantic representation corresponds to the words that express that part of the meaning. One possible problem with this approach is that it could be hard to find a direct correspondence between some bits of the semantic representation and the surface form. The resulting alignment puts words and elements of the semantic representation in a n-to-n relation, where a single word could correspond to various pieces in the semantic representation. A possible maximal alignment of "a blue cup" to its meaning representation is shown in figure 3.3.

Both the maximal and the minimal alignment strategies provide a sufficiently fine-grained alignment to employ for statistical generation, however minimal alignment allows for an easier implementation in practice, since each element of the meaning representation is either aligned to one word or nothing. In Section 3.4 I propose a method to align semantic representations to the surface form at the word level that is based on a minimal alignment strategy, rather than maximal alignment, having found that the former is best suited for the specific abstract meaning representation introduced in the next section.

With sufficient data in the form of texts aligned with semantic representations, this information, independently from which strategy one chooses, can be automatically learned, thus creating a statistical model to generate surface forms from abstract, logical representations. For instance, events in a logical form have a high chance of being expressed, in English, by verbs.

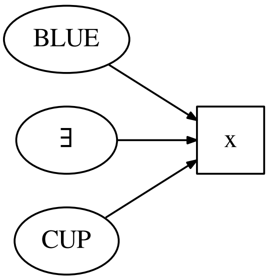


Figure 3.4: Logical form graph representing the meaning of “A blue cup”.

However, aligning semantic representations with words is a difficult enterprise, primarily because formal semantic representations are not flat like a string of words and often form complex structures. While a complete definition of a flat formalism for representing abstract meaning is given in the next section, here a tuple-based format is informally introduced in order to present the features of the alignment method. The basic idea is that formal semantic representations are represented as a set of tuples of fixed cardinality. Returning to the earlier example representation for “a blue cup”, the logical form corresponding to its meaning could be represented by a sequence of tuples like $\langle$BLUE,instance,$x\rangle$ and $\langle$CUP,instance,$x\rangle$, meaning respectively that BLUE is a predicate over x and CUP too. For readability, the tuples (triples, actually, in this case) can be displayed as edges of a directed graph, where the elements of the tuple serve, in order, as tail node, edge label and head node. An example of such visualization is shown in Figure 3.4.

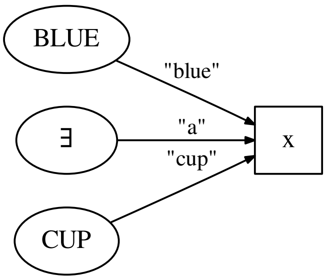


Figure 3.5: Logical form graph representing the meaning of "A blue cup" aligned with the surface form.

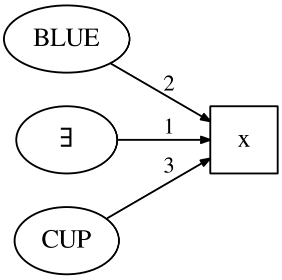


Figure 3.6: Encoding local word order.

Note that in this example some information is missing, such as the conjunction, which is implicitly represented by the fact that multiple edges are incident on the same node. Further down in this chapter a complete formalism is introduced that can indeed represent every bit of semantic information without sacrificing the capability of alignment with the text. The important thing now is to show how alignments between tuples and words can be realized, which is done by adding an element to each tuple denoting the surface string, for instance $\langle$CUP,instance,x,"cup"$\rangle$, as in Figure 3.5.

Note how the node labeled by the existential quantifier in the graph is used to accommodate for the alignment of the determiner. A real, complete formalism will have to account for this kind of representations, for example including the concepts of clauses and scope.

The alignment can be further refined by adding information about the local order of surface expressions. Again, this is done by adding an element to the tuple, in this case one that denotes the local order of a logical term. This will be clear by continuing with the example, where word order information is added, encoded as numerical indexes in the tuple, e.g., ⟨CUP,instance,x,"cup",3⟩, as Figure 3.6 shows. There are arguably different ways of aligning words with the tuples of a meaning representation besides the difference between maximal and minimal alignment. In this case, the information necessary to keep the words in the right order must be encoded somehow. There are at least two strategies to put the word order into the alignment, namely *global* and *local* ordering. With global ordering, the absolute position of each word in the surface form is added as an index to the tuple the word is aligned to. Using the local ordering strategy, instead, means to locally rank subsets of tuples that share the same head.

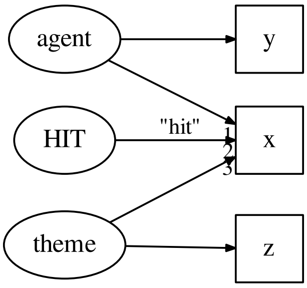


Figure 3.7: Graph for a meaning representation aligned with a partial surface form.

These graphs show the association of the term x with the surface strings "a blue cup", but the way to express local order is not limited to words and can be employed for partial phrases as well. This can be achieved by using the same kind of numerical indexes already used for the alignment of words. The example in Figure 3.7 shows how to represent an event "hit" with the terms filling its roles *agent* and *theme*, preserving their relative order. Note that this is a particular way of representing events, entities and the relationship between them — a *neo-Davidsonian* kind of representation. The choice of this representation actually plays an important role in the method of alignment presented here, that will be clear later in this chapter. The *partial* surface form corresponding to the semantics encoded in the figure is "y hit z", where y and z are placeholders for parts of the surface that have not been realized yet. Surface forms like the one in the example are partial, or *incomplete*, because they contain variables. Conversely, a surface form is *complete* when it only contain words. In a way, this scheme defines a sort of context-free grammar for the variables contained in the logical forms and their respective surface forms. In the example above, we can say that the production rule $x \rightarrow y$ hit z applies, so that the surface form of x depends on the surface forms of y and z.

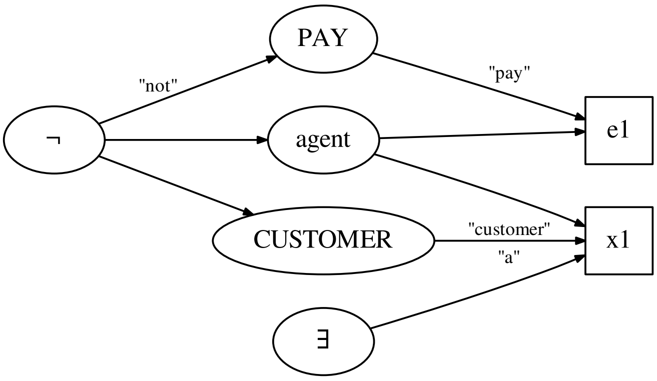


Figure 3.8: Meaning representation of "a customer did not pay". The word "did" cannot be easily aligned within this schema.

This is the basic idea of aligning surface strings with parts of a deep semantic representation. Note that precise alignment is only possible for words whose lexical semantics include variables, e.g., nouns, verbs, etc. For words that introduce scope operators (negation particles, coordinating conjuncts) one cannot have the cake and eat it: specifying the order of the tuples with respect to an entity or event variable directly and at the same time associating it with an operator is not always possible. Figure 3.8 illustrates this problem with an example meaning representation of the text "a customer did not pay" — the auxiliary verb "did" does not find an intuitive place in the alignment. The next section explains how this issue can be addressed by introducing extra tuples that complement the semantic representation to facilitate perfect alignment.

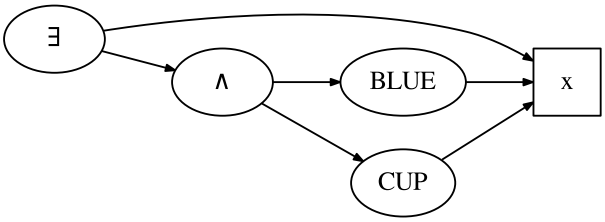


Figure 3.9: Logical form graph representing the meaning of "A blue cup" with the conjunction explicitly represented.

As mentioned earlier, the graph-based representations used as examples throughout this section are arguably incomplete. For ease of presentation, some information has been left outside the graphical representation. For instance, the conjunction in the formula $\exists x(\textsc{blue}(x) \wedge \textsc{cup}(x))$ is rendered implicit, since the main point of these examples is to show how the meaning representation is aligned to the surface, rather than to give a complete formalism (which is done in the next sections). A version of the example in Figure 3.4 where the conjunction is explicitly represented would look like the graph in Figure 3.9.

3.3 A Semantic Representation for Generation

The formalism to encode the input of an NLG system does not have a standard definition in literature, mainly because different systems have different needs and in general they aim at different goals. For instance, the system subject of this thesis focuses on the generation of natural language from structures that already contain the information to be conveyed, although expressed in a formal and language-independent way. This leaves the problem of selecting the relevant information out of the picture, which is not always the case among NLG systems. On the other hand, if the formalism is too close to the natural language, for instance including syntactic or lexical information, then the system would suffer from a loss of generality of the utterances it can produce.

Moreover, as mentioned before, a theoretically sound formalism is an ideal input of a general purpose system for NLG, because of it being capable, for instance, of supporting logical inference. In this section I argue for one specific formalism as a suitable candidate for this task. The formalism is based on an existing one rooted into formal semantics, with the important addition of being transformed into a *flat* formalism in order to facilitate the alignment with surface. In fact, the formalism makes possible a fine-grained alignment with the surface structure at the level of words, of the type described in the previous section.

3.3.1 Discourse Representation Theory

The choice of semantic formalism should ideally be independent from the application of natural language generation itself, to avoid bias and specific tailoring the semantic representation to one's (technical) needs. Furthermore, is it a desirable property for such formalism to have a model-theoretic backbone, to ensure that the semantic representations one works with actually have an interpretation, and can consequently be used in inference tasks using, for instance, automated deduction for first-order logic. Given these criteria, a good candidate is Discourse Representation Theory (DRT) by Kamp [1984], a dynamic theory of the meaning of natural language. DRT models the way humans interpret language by building formal structures called Discourse Representation Structures (DRSs) and updating them when new pieces of discourse are considered. Here, the input consists of a variant of such DRSs, that are in fact abstract representations of meaning. The dynamic aspect of DRT is not relevant for the purpose of this thesis, because here DRSs are used as static representations of meaning, i.e., a formal language to encode meaning.

One of the reasons behind the original formulation of DRT in the eighties is that the formalisms that were existent at the time were not capable of expressing the meaning of certain kinds of sentences. Consider for instance the following two sentences:

1. If **Pepe** owns a donkey, he beats it.
2. If **a farmer** owns a donkey, he beats it.

In the first sentence, the pronoun *he* clearly refers to *Pepe*, the subject of the first clause, and *it* refers to the the donkey he owns. In the second sentence, however, *he* does not refer to a singular farmer, but rather to *every* farmer that owns a donkey, while the pronoun *it* refers to *the donkey that he owns*. The problem is that the indefinite article *a* in natural language is typically translated into an existential quantifier, but, in the case of the second sentence, the donkey should be represented in the scope of a universal quantifier. DRT is indeed capable of dealing with quantifiers (such as *every*) and their scope, thanks to the way DRSs are built. A DRS comprises two parts: a set of *discourse referents* (the entities introduced in the text), and a set of *conditions*, describing the properties of the referents and the relations between them.

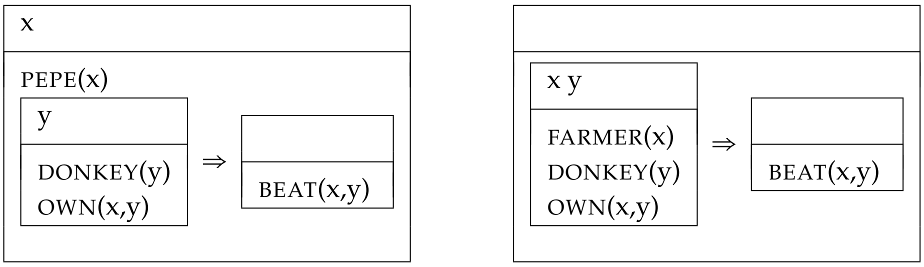


Figure 3.10: DRSs representing the meaning of the two sentences "if Pepe owns a donkey, he beats it." (left) and "if a farmer owns a donkey, he beats it." (right).

DRSs are traditionally visualized as boxes, with the referents placed in the top part, and the DRS conditions in the bottom part. Figure 3.10 shows the two DRSs representing the meaning of the two donkey sentences above.

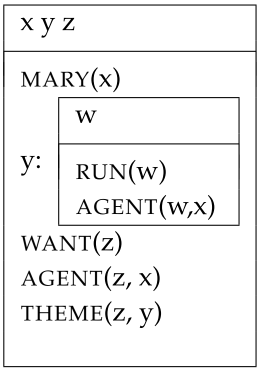


Figure 3.11: DRS for the sentence "Mary wants to run".

The DRS conditions can be divided into basic and complex conditions. The basic conditions are used to describe names of discourse referents (named entities), events, entities, relations between discourse referents, cardinality of discourse referents denoting sets of objects, or to express identity between discourse referents. The complex conditions introduce *embedded* DRSs: implication (⇒), negation (¬), disjunction (∨), and modalities (□, ◊). DRSs are thus of recursive nature, and the embedding of DRSs restricts the resolution of pronouns (and other anaphoric expressions), which is one of the trademarks properties of DRT. Figure 3.11 shows an example of DRS representing the meaning of a sentence containing a subordinate clause, specifically the one introduced by the control verb “want” in the sentence “Mary wants to run”.

The representation introduced in this chapter and used through the rest of the thesis comprises a particular set of well-known extensions of standard DRT, namely thematic roles from VerbNet Kipper et al. [2008], treatment of rhetorical relations [Asher and Lascarides, 2003], presuppositions [Van der Sandt, 1992b], modal logic, and hybrid conditions (conditions connecting a discourse referent to a DRS via labeling). These extensions are implemented in the version of DRSs contained in the linguistic resource introduced in the next chapter. The DRS in Figure 3.11 highlights a couple of features of this kind of semantic representation. First, the embedded DRS is labeled (y) in order to allow the outer condition THEME(z, y) to refer to it (“to run” is the object of what Mary wants). Second, a particular way of encoding semantics is used here, namely a neo-Davidsonian semantic representation [Parsons, 1990]. In this kind of representation events are explicit variables and the participation of an entity in an event is encoded by an additional condition (a *thematic role*) rather than as an argument of the event itself, e.g. OWN(x,y) vs. OWN(z) ∧ AGENT(z,y) ∧ THEME(z,y). Both of these two characteristics are play a role in the new formalism proposed in the next section as a basis for natural language generation.

DRSs are formal structures and come with a model-theoretic interpretation. This interpretation can be given directly [Kamp, 1984] or via a translation into first-order logic [Muskens, 1996]. This is interesting from a practical perspective, because it permits the use of efficient existing inference engines developed by the automated deduction community. Applying logical inference can play a role in tasks surrounding NLG (e.g., summarization, question answering, or textual entailment), but also dedicated components of NLG systems, such as generating definite descriptions, which requires checking contextual restrictions [Gardent et al., 2004].

3.3.2 Discourse Representation Graphs

DRSs are capable of effectively representing the meaning of natural language, covering many linguistic phenomena including pronouns, quantifier scope, negation, modality, and presuppositions. They are recursive structures put together by logical and non-logical symbols, as in predicate logic. The way DRSs are nested inside each other give DRT the ability to explain, among other phenomena, the behavior of pronouns (see the example in Figure 3.10) and presuppositions [Van der Sandt, 1992b]. However, aligning DRSs with text with fine granularity is hard because words and phrases introduce different kinds of semantic objects in a DRS: discourse referents, predicates, relations, but also logical operators such as negation, disjunction and implication that introduce embedded DRSs. A precise alignment of a DRS with its text on the level of words is therefore a non-trivial task. However, since the constructs of DRT are designed to represent natural language, the alignment is more intuitive compared to first-order logic formulas.

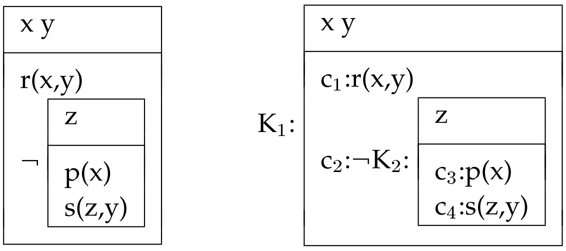
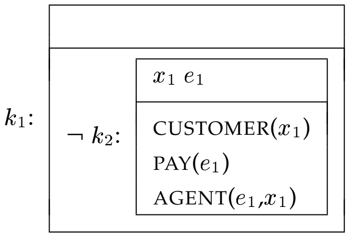


Figure 3.12: From DRS to DRG: labeling.

To overcome this issue, the format of DRSs must be transformed, making all recursion implicit while retaining the same amount of information and not introducing ambiguity. Note that labeling conditions is crucial to distinguish between syntactically equivalent conditions occurring in different (embedded) Here the assumption that conditions appear in the DRS in which their discourse referents are introduced is absent. The example in Figure 3.12 illustrates that this assumption is not sound: the condition p(x) is in a different DRS than where its discourse referent x is introduced. The reification procedure yields "flatter" representations than similar formalisms [Copestake et al., 2005, Reyle, 1993], and this makes it more convenient to align surface strings with DRSs with a high granularity.

While the basic ideas for this new format are already introduced in Section 3.2, here I present a concrete formalism. The format is based on a tuple-based representation of a DRS, where each bit of information in the original structure is translated into triples that can also be seen as *edges* connecting *nodes* of a directed graph. The new structure is called a Discourse Representation Graph (DRG). A DRG can therefore be written in both formats, as a directed graph or as a set of triples <node, edge, node>, so to obtain a tabular representation. Figure 3.13 shows, side by side, a DRS and the corresponding DRG written in tuple-based format and graph format. While the graph notation makes some notions about the DRG format easier to grasp, the representation commonly used in computer implementations is the tuple-based one.



k_1	unary	$\neg$
$\neg$	scope	k_2
k_2	referent	e_1
k_2	referent	x_1
k_2	event	PAY
k_2	concept	CUSTOMER
k_2	role	AGENT
CUSTOMER	instance	x_1
PAY	instance	e_1
AGENT	internal	e_1
AGENT	external	x_1

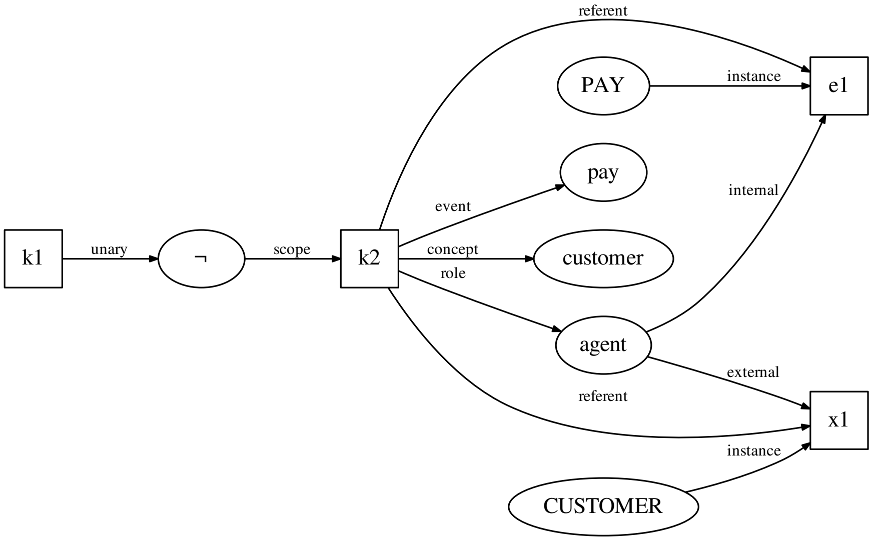


Figure 3.13: DRS and corresponding DRG for "A customer did not pay."

Throughout the thesis both formats are used, depending on whether the emphasis has to be put on readability or on implementation details.

The trick, well-known in computer science, that makes possible this translation of formats is the application of a form of *reification* over DRSs. Every DRSs gets a unique label, and, accordingly, the arity of DRS conditions increases by one for accommodating a DRS label — see for instance the newly introduced labels k_1 and k_2 in the example in Figure 3.13, corresponding to the main and embedded DRSs.

A DRS is an ordered pair of discourse referents (variables over entities) and DRS-conditions. DRS-conditions are basic (representing properties or relations) or complex (to handle negation and disjunction). To reflect these different constructs, we distinguish three types of tuples in DRGs, each with their own sub-classification:

- ⟨K,referent,X⟩ means that X is a discourse referent in K.
- ⟨K,condition,C⟩ means that C is a condition in K, having sub-type concept, event, relation, role, named, cardinality, attribute, unary, and binary.
- ⟨C,argument,A⟩ means that C is a condition with argument A.

The hypothesis is that the DRG format is convenient for the alignment with the surface form. In particular, by "unrolling" every element of the original DRS into a fine-grained set of tuples, the format lends itself naturally to a minimal alignment, where to each tuple can correspond a lexical item. The next section shows with concrete examples how this alignment is made.

With the help of a concrete example, it is easy to see that DRGs have the same expressive power as DRSs. Consider for instance a DRS with negation, before and after labeling it (Figure 3.12).

Now, from the labeled DRS one can derive the following three referent tuples: ⟨K_1,referent,x⟩, ⟨K_1,referent,y⟩, and ⟨K_2,referent,z⟩; the following four condition tuples: ⟨K_1,relation,c_1:r⟩, ⟨K_1,unary,c_2:¬⟩, ⟨K_2,concept,c_3:p⟩, and ⟨K_2,relation,c_4:s⟩; and the following argument tuples: ⟨c_1:r,internal,x⟩, ⟨c_1:r,external,y⟩, ⟨c_2:¬,scope,K_2⟩, ⟨c_3:p,instance,x⟩, ⟨c_4:s,internal,z⟩, and ⟨c_4:s,external,y⟩. From these tuples, it is straightforward to recreate a labeled DRS, and by dropping the labels subsequently, the original DRS resurfaces again.

The formalism of Power [1999] consists of semantic networks enriched with scope information for content planning. Power's scoped semantic networks have a lot in common with DRGs, but there are also formal differences, as they do not have the same expressive power. For instance, scoped semantic networks cannot encode phrases like "a red or blue car" with the correct scope assignments. This is possible with a DRG.

3.4 Word-Aligned DRGs

The two components of the pipeline described in the previous sections have the role of constructing an alignment between the input DRG and the surface form, like the one sketched in Section 3.2. In this section I show how such an alignment between surface text and its logical representation is realized in practice by adding information to the tuples that make up a DRG. This sounds more straightforward than it is. For some word classes this is indeed easy to do. For others, additional machinery needs to be introduced in the formalism.

Determiners are usually associated with referent tuples. Content words, such as nouns, verbs, adverbs and adjectives, are typically directly associated with one-place relation symbols, and can be naturally aligned with argument tuples. Verbs are assigned to *instance* tuples linking its *event* condition; likewise, nouns are typically aligned to *instance* tuples which link discourse referents to the *concepts* they express; adjectives are aligned to tuples of *attribute* conditions. Finally, words expressing relations (such as prepositions), are attached to the *external* argument tuple linking the relation to the discourse referent playing the role of external argument.

k_1	unary	$\neg$		
$\neg$	scope	k_2		
k_2	referent	e_1		
k_2	referent	x_1	**1**	**A**
k_2	event	PAY		
k_2	concept	CUSTOMER		
k_2	role	$agent$		
CUSTOMER	instance	x_1	**2**	**customer**
PAY	instance	e_1	**4**	**pay**
$agent$	internal	e_1	**1**	
$agent$	external	x_1		
k_2	**surface**	e_1	**2**	**did**
k_2	**surface**	e_1	**3**	**not**
k_2	**surface**	e_1	**5**	**.**

Figure 3.14: Word-aligned DRG for "A customer did not pay." All alignment information (including surface tuples) is highlighted.

Although the strategy presented for DRG–text alignment is intuitive and straightforward to implement, there are surface strings that don't correspond to something explicit in the DRS. To this class belong punctuation symbols, and se-

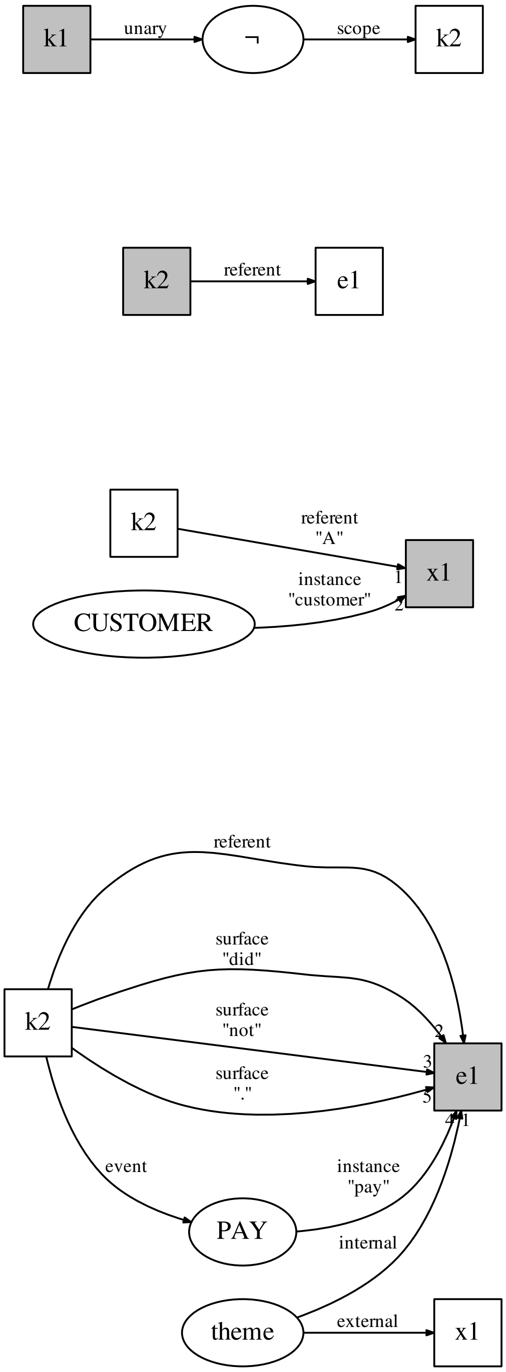


Figure 3.15: Discourse referents of the DRG shown in Figure 3.14 (k_1, k_2, e_1 and x_1 and their incident edges).

mantically empty words such as (in English) the infinitival particle, pleonastic pronouns, auxiliaries, *there* insertion, and so on. Furthermore, function words such as "not", "if", and "or", introduce semantic material, but for the sake of surface string generation could be better aligned with the event that they take the scope of. To deal with all these cases, DRGs are extended with *surface tuples* of the form ⟨K,surface,X⟩, decorated with the required surface strings. Figure 3.14 shows an example of a DRG extended with such surface tuples.

Note that these kind of tuples don't have any influence on the meaning of the original DRS — they just serve for the purpose of alignment of the required surface strings. Also note in Figure 3.14 the indexes that were added to some tuples. They serve to express the local order of surface information.

Following the idea sketched in Section 3.2, the total order of words is transformed into a local ranking of edges relative to discourse referents. This is possible because the tuples that have word tokens aligned to them always have a discourse referent as third element (the *head* of the directed edge, in terms of graphs). Tuples that share the same discourse referent are grouped and then labeled with indexes reflecting the relative order of how these tuples are realized in the surface form. This process is described in greater detail in Chapter 5.

Illustrating this with the example in Figure 3.14, in the DRG there are two discourse referents: x_1 and e_1. The discourse referent x_1 is associated with three tuples, of which two are indexed (with indexes 1 and 2). Generating the surface string for x_1 succeeds by traversing the edges in the order specified, resulting in [A,customer] for x_1. The referent e_1 associates with six tuples, of which four are indexed (with indexes 1–4). The order of these tuples would yield the partial surface string [x_1,did,not,pay,.] for e_1. The relevant four sub-graph are shown in Figure 3.15 where tuple indexes are written on the head of the edges.

Note that the syntax of DRSs ensures that all discourse referents are linked to each other by taking the transitive closure of all binary relations appearing in a DRS, and therefore it is possible to reconstruct the total order from composing the local orders. The next section will introduce an algorithm that generalizes this process in order to construct the complete surface form corresponding to the whole DRG.

3.5 Overview of the System

The traditional architecture of many NLG systems is a pipeline in which the output of a component forms the input for the following one. The previous chapter already covered this kind of architecture and its building blocks, i.e., Macro-planning (or Content Planning), Micro-planning and Surface Realization, in this order.

The system introduced in this section is called the *Unboxer*, a pun on being the inverse of Boxer, the tool for semantic analysis of natural language. It is based on a pipeline architecture, but there are a few distinctions from the standard architecture. First, Content Planning is considered already taken care of. This is because the structures from which the Unboxer generates natural language expressions already encode a kind of content plan, that is, the content to be conveyed but also discourse-level information. Secondly, some components of the Unboxer pipeline may work in parallel, and they do not correspond to the usual NLG modules in their behavior.

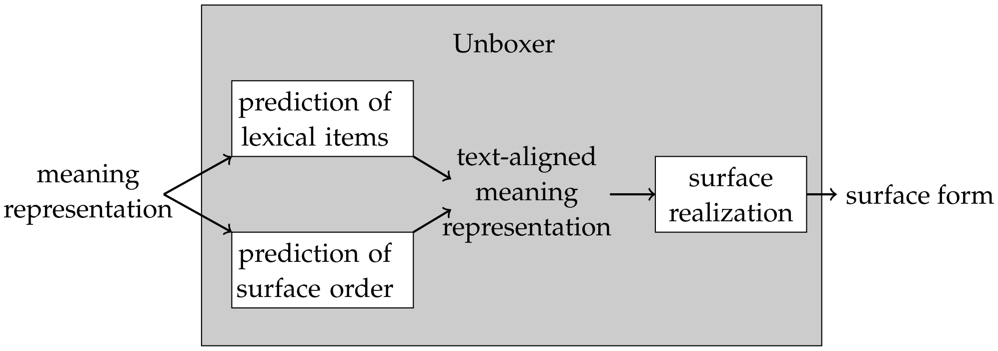

Unboxer
meaning representation
prediction of lexical items
prediction of surface order
text-aligned meaning representation
surface realization
surface form

Figure 3.16: A schematic view of the architecture of the Unboxer system.

The Unboxer is a modular system made of three main components that takes in input an abstract meaning representation and produces a text string. a depiction of the architecture of the system is in Figure 3.16.

The format of the input is a text file containing an abstract representation of meaning. This format is rooted in the semantic theory described in the previous sections and it is able to effectively represent any kind of information to be generated. This file could be, for instance, the output of an external component acting as content planner, or taken from an existing database. The origin of the input of the Unboxer pipeline is out of the scope of this dissertation, as it is not important to answer the research questions relative to the generation of natural language from such structures.

The input meaning representation is fed to two modules that serve the purpose of constructing an *alignment* between the abstract structure and the text. The alignment strategy introduced in Section 3.2 is a key idea of the whole process, and it consists in the association of information about the surface form (content words, their order, etc.) to the abstract structure at a sufficiently fine-grained level. The module labeled “prediction of lexical items” is responsible for the choice of the words to be used to produce the surface form, since the input structure only contains abstract representation of entities, events, attributes and the like. The “prediction of surface order” module is instead responsible for the information on the relative position of words and phrases in the surface form, which is absent in the abstract meaning representation. Sections 3.5.1 and 3.5.2 briefly describe the surface order module and the lexicalization module of the pipeline, while their principles and ways of working are treated in greater detail in two dedicated chapters.

The two modules, in this architecture, work in parallel, starting from the same input and producing different outputs. In this respect, the architecture of the Unboxer deviates from a classic pipeline model, where each transformation step strictly follows the preceding one. Alternative architectures, for instance one in which the two modules somehow inform each other, are left for future exploration. The combination of the output of these two modules is a structure that contains the information needed to generate the final output. Once the surface-aligned structure is ready, the final module of the pipeline uses it to construct the final output, that is, the surface form that expresses the meaning encoded in the input structure. This module is based on an algorithm that works by composition, with interesting properties, that is described in Section 3.5.3.

3.5.1 Surface Order

A DRG is a logical formula, containing no information about the surface form whatsoever, therefore the need for machine learning approaches to predict the alignment with the text. The usual approach found in literature is based on language modeling, that is, to produce all possible generations and then rank them based on a statistical model of the target language. Such statistical models can predict the probability of a given sentence from the probability of the single n-grams it is made of. Language modeling is a field on its own and a very important one in Computational Linguistics, with many applications. For a survey of language modeling methods see Zhai [2008].

The Unboxer approach to the prediction of word order is different from the overgeneration and ranking approach. The component for the prediction of word order is a supervised statistical system that uses the alignment information in gold standard DRGs to learn to predict the same alignment in arbitrary DRGs. In particular, the system exploits the strategy of representing the order of the tuples locally with respect to the discourse referents. Each referent in a DRG is associated with a (usually small) list of edges to rank, and ordering such lists is enough to predict the total order of the final surface form due to the connected structure of the DRGs.

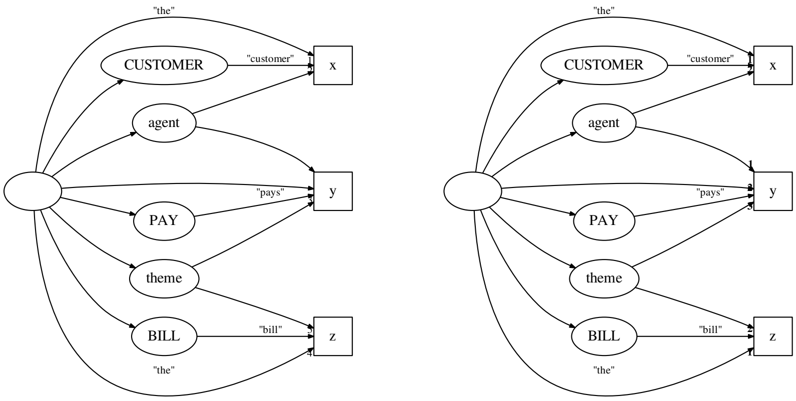


Figure 3.17: Difference between global ordering (left) and local ordering (right) in the alignment of a DRG with the surface form "The customer pays the bill."

Casting the problem this way, the prediction of word order in generation from a DRG is reduced to a *learning to rank* problem, a type of machine learning problem. The elements to rank are the tuples, that is, the problem is to assign a numeric index to each tuple. Each list of tuples linked to each discourse referent is an instance of the problem, and text-aligned DRGs provides a gold standard for training a statistical model. The strategy to encode word order used here is *local* order (see Section 3.2), that is, indexes are added to the tuples encoding their order relative to each discourse referent. It is possible to recover the full word order from an alignment in which local ordering is used, while at the same time thanks to this strategy the task can be cast as a machine learning problem with a manageable input. For these two reasons the strategy implemented in the Unboxer system is local ordering. Figure 3.17 shows the difference between global and local ordering. Note that local ordering needs a higher number of indexes on the tuples.

To automatically learn and predict the order of the words in an alignment in a supervised way is ranking problem. Chapter 5 is dedicated entirely to the component that solves this problem implementing a supervised solution.

3.5.2 Lexicalizaton

DRGs are abstract logical forms in which the symbols do not necessarily hold a meaning outside the formula. In order for DRGs to be useful for practical purposes they have to express predicates and relations over real word entities. For this reason the semantic representation must provide links to some kind of real-world knowledge base. A very basic example of such a knowledge base is a dictionary, where each entry is a particular sense of a word and in general multiple entries correspond to single words. However, the simple flat structure of a common dictionary is not suitable for the representation of knowledge in general, as it often lacks information such as semantic relations between concepts and events.

A better way of formally representing world knowledge is to use an *ontology*, that is, a structured collection of concepts, relations and rules, organized taxonomically and encoded in a format readable by computers. The Unboxer indeed employs a knowledge base that somehow is positioned halfway through an electronic dictionary and a full-fledged computational ontology (see 6.4).

Once the abstract meaning representation is linked to a knowledge base, the problem remains to generate proper words in the target natural language in the surface form. If the knowledge base already provides alternatives in the form of words and phrases, then the problem becomes *lexical choice*, i.e., the task of choosing the most apt words to express a given concept. The result of the lexical choice step must be then complemented with the generation of the correct *morphological form* of the chosen words, in order for the final result to be grammatical, because the lexical choice step only provides uninflected lemmata.

Just as the prediction of surface order, the component of the Unboxer that implements the solution to the lexicalization step is complex, and therefore it requires a chapter on its own. This part of the NLG process, along with the choice of a knowledge base suitable for generation are treated in detail in Chapter 6.

3.5.3 Surface Realization

A completely aligned DRG contains all the information necessary to generate its corresponding surface form, that is, the word forms and the order in which they occur. Still, it is not trivial to use this information to put the pieces together, because the information is scattered all around the (aligned) semantic representation, far from resembling a linear structure. This section introduces an algorithm to generate surface forms by composing partial surface forms expressing concepts, events and discourse units.

The algorithm works in two steps. First, a complete or incomplete surface form is associated with each discourse referent, that is, a sequence of tokens and variables is constructed for each discourse referent in the input structure and its alignment with the surface form. As a second step, these surface forms are put together in a bottom-up fashion, to generate the complete output. The goals of this composition phase is that of associating all of the discourse referents with their own complete surface representation. The surface form associated with the discourse unit that contains all other discourse units is then the text aligned with the original DRG.

The surface forms of discourse referents are lists made of tokens and other discourse referents. Recall that the order of the elements of a discourse referent's surface form is reflected by the local ordering of tuples, as explained in the previous section, and tuples with no index are simply ignored when reconstructing surface strings.

The surface form is composed by taking each tuple belonging to a specific discourse referent, in the correct order, and adding the tokens aligned with the tuple to a list representing the surface string for that discourse referent. Additionally, an important part of this process is that binary DRS relations, represented in the DRG by a pair of internal and external argument tuple, are followed unidirectionally: if the tuple is of the *internal* type, then the discourse referent on the other end of the relation (i.e., following its *external* tuple edge) is added to the list. Surface forms for embedded DRSs include the discourse referents of the events they contain. Going back to the examples of discourse referents in Figure 3.15, let us focus on x_1 and e_1, representing respectively the customer entity and the event of (not) paying. The construction of the surface for x_1 is straightforward: the algorithm considers the two inward edges <k_2, referent, x_1> and <CUSTOMER, instance, x_1> in this order as specified by the indexes, thus creating the list of two work tokens ["A", "customer"] for the discourse referent x_1. This is an example of complete surface form, the final surface form for this discourse referent. The situation is different for e_1, where the inward edge with index 1 is of type *internal*, thus triggering the algorithm to follow the link and include the discourse referent on the other side of the *theme* relation (x_1) into the surface form. Following the usual procedure, the algorithm obtains then [x_1, "did", "not", "pay", "."] as the incomplete surface form for the discourse referent e_1. The surface forms at this step for k_2 is given just by e_1, while that of k_1 is k_2, that is, the surface form corresponding to a DRS is ultimately coming from its head event.

Typically, discourse units contain exactly one event (the main event of the clause). Phenomena such as gerunds (e.g., “the laughing girl”) and relative clauses (e.g., “the man who smokes”) may introduce more than one event in a discourse unit. To ensure correct order and grouping, the alignment implements a technique borrowed from description logic [Horrocks and Sattler, 1999] and invert roles in DRGs. Rather than representing “the laughing girl” as [girl(x) $\wedge$ agent(e,x) $\wedge$ laugh(e)], the formalism represents it as [girl(x) $\wedge$ agent^{-1}(x,e) $\wedge$ laugh(e)], making use of R(x,y) $\equiv$ R^{-1}(y,x) to preserve meaning. This trick ensures a correct encoding of that the local order of noun phrases with relative clauses and alike.

To wrap things up, a *composition* operation is used to derive complete surface forms for DRGs. Composition puts together two surface forms, where one of them is complete, and one of them is incomplete. It is formally defined as follows:

$$\frac{\rho_1 : \tau \qquad \rho_2 : \Lambda_1 \rho_1 \Lambda_2}{\rho_2 : \Lambda_1 \tau \Lambda_2} \tag{3.1}$$

where ρ_1 and ρ_2 are discourse referents, τ is a list of tokens, and Λ_1 and Λ_2 are mixed lists of word tokens and discourse referents. In the example from Figure 3.14, the complete surface form for the discourse unit k_1 is derived by means of composition as formulated in (3.1) as follows:

$$\frac{k_2 : e_1 \qquad \dfrac{x_1 : \text{A customer} \qquad e_1 : x_1 \text{ did not pay}}{e_1 : \text{A customer did not pay .}}}{k_2 : \text{A customer did not pay .}}$$

The procedure for generation described here is reminiscent of the work of Shieber [1988] who also employs a deductive approach. In particular the composition operation can be seen as a simplified *completion*.

3.6 Discussion and Conclusion

To test the strategy for alignment and the algorithm for surface realization presented in this chapter, I took advantage of an existing database of surface-alignment DRGs that will be introduce extensively in the next chapter. After implemented a first prototype of the realization algorithm and running it on examples taken from such annotated corpus, naturally, I came across phenomena that are notoriously hard to analyze. Most of these cases can be handles adequately, but not all of them, which will require further work. Here is a list of interesting and somewhat challenging phenomena.

3.6.1 Embedded Clauses

In the variant of DRT used for this thesis, propositional arguments of verbs introduce embedded DRSs associated with a discourse referent. This is a good test for the surface realization formalism, because it would show that it is capable of recursively generating embedded clauses.

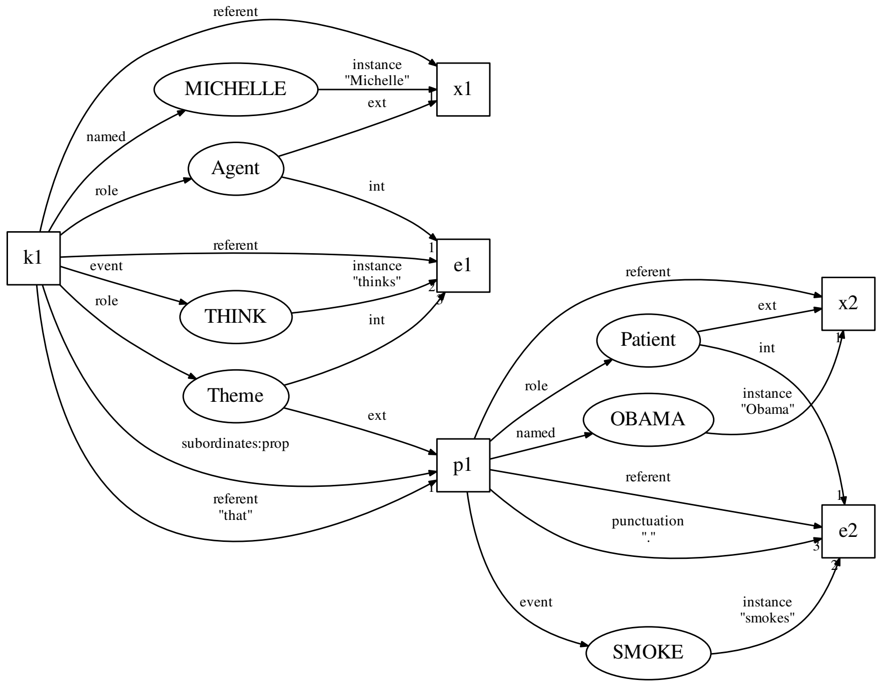


Figure 3.18: Word-aligned DRG for the sentence “Michelle thinks that Obama smokes.”

Figure 3.18 shows the DRG for the sentence “Michelle thinks that Obama

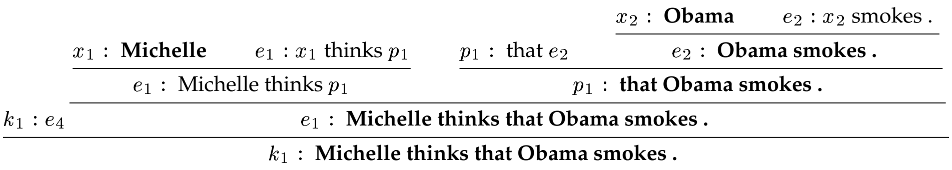

x_2 : **Obama**
$e_2 : x_2$ smokes .
x_1 : **Michelle**
$e_1 : x_1$ thinks p_1
p_1 : that e_2
e_2 : **Obama smokes .**
e_1 : Michelle thinks p_1
p_1 : **that Obama smokes .**
$k_1 : e_4$
e_1 : **Michelle thinks that Obama smokes .**
k_1 : **Michelle thinks that Obama smokes .**

Figure 3.19: Surface composition of embedded structures. The complete surface forms are highlighted.

smokes." Here the surface forms of two discourse units (main and embedded) are generated. In order to generate the complete surface form, first the embedded clause is generated, and then composed with the incomplete surface form of the main clause, as shown in Figure 3.19. As noted earlier, during the composition process, the *complete* surface form for each discourse referent is generated (highlighted in bold face in the figure), showing a clear alignment between the elements of the semantic representation and the surface forms they represent.

3.6.2 Coordination

Coordination is another good test case for a linguistic formalism. Consider for instance the sentence "Subsistence fishing and commercial trawling occur within refuge waters", where two noun phrases are coordinated, giving rise to either a distributive (introducing two events in the DRS) or a collective interpretation (introducing a set formation of discourse referents in the DRS).

Interestingly, using the distributive interpretation DRG as input to the surface realization component could result, depending on how words are aligned, in the surface form "fishing occurs and trawling occurs", rather than "fishing and trawling occur". This kind of phenomena poses the problem of what dimensions are taken into account during the evaluation of NLG. The former realization is not wrong in that it preserves the intended meaning, but yet it is arguably worse than the former in terms of fluency.

The formalism proposed in this chapter can account for both interpretations, as

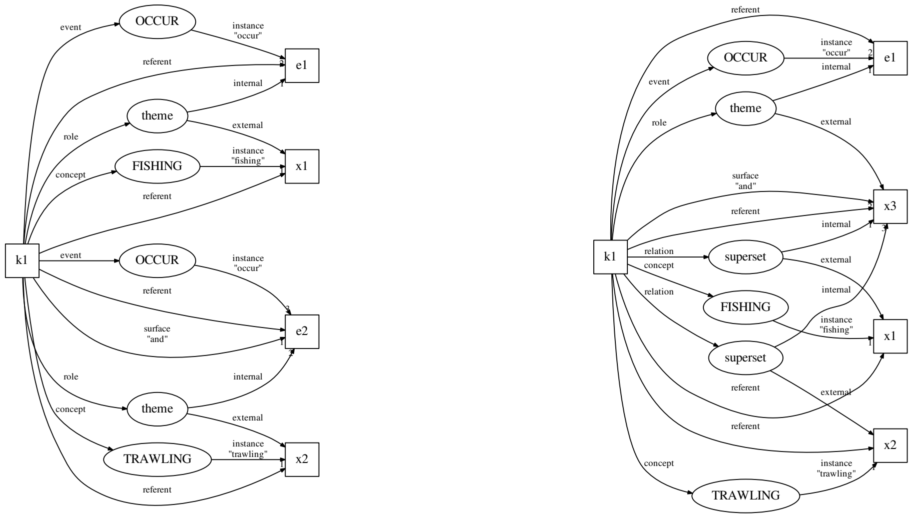


Figure 3.20: Analysis of NP coordination, in a distributive (left) and a collective interpretation (right).

shown in Figure 3.20.

3.6.3 Long-Distance Dependencies

Cases of extraction, for instance with WH-movement, could be problematic to capture with the DRG formalism. This is in particular an issue when extraction crosses more than one clause boundary, as in "Which car does Bill believe John bought". Even though these cases are rare in the real world, a complete formalism for that serves as the basis for a general-purpose NLG pipeline must be able to deal with such cases. The question is whether this is a separate generation task in the domain of syntax [White et al., 2007], or whether the current formalism needs to be adapted to cover such long-distance dependencies. Another range of complications is caused by discontinuous constituents common in languages other than English. For instance, consider the Dutch sentence "Ik heb kaartjes gekocht voor Berlijn" (literally: "I have tickets bought for Berlin"), where the prepositional phrase "voor Berlijn" is an argument of the noun phrase "kaartjes". In the proposed formalism the only alignment possible would result in the sentence "Ik heb kaartjes voor Berlijn gekocht", which is arguably a more fluent realization of the sentence, but does not correspond to the original text. If one uses the original text as gold standard, this could cause problems in evaluation. One could also benefit from this deficiency, and use it to generate more than one gold standard surface string. This is something to explore in future work.

3.6.4 Control Verbs

In constructions like "John wants to swim", the control verb "wants" associates its own subject with the subject of the infinitival clause that it has as argument. Semantically, variable binding is used to account for this kind of constructions. Generating an appropriate surface form for semantic representation with controlled variables is a challenge: a naive approach would generate "John wants John to swim". One possible solution is to add another derivation rule for surface composition dedicated to deal with cases where a placeholder variable occurs in more than one partial surface form, substituting a null string for a variable following some heuristic rules. A second, perhaps more elegant solution is to integrate a language model into the surface composition process as a post-processing step to filter out correct but not natural sounding realizations.

3.6.5 Surface Tuples

Arguably, "surface" tuples as described in Section 3.4, are not part of a logical form. Even when integrating the logic with world knowledge, for instance by linking entities and events to entries in an ontology, it is usually impossible to align all the words. In particular, particles such as infinitivals and logic connectives, or punctuation marks, do not always fit in the structure of a DRG in an intuitive way. Moreover, some of this particles can carry important semantic information, as is the case of logical connectives ("not", "if", "and", "or", ...), but due to the alignment strategy chosen for generation purposes, they find a better place when aligned with surface tuples. The problem with surface tuples is that a realistic DRG that acts as input for an NLG system, e.g., the translation into logic of numeric data, will not contain them. There are two way of approaching this problem. One is to change the procedure of alignment to try and fit every word in the surface form into the logic structure, for instance by aligning sequences of words to single non-surface tuples. This approach presents a problem if applied together with the current surface composition procedure, that is, when the words to be aligned to a single tuple are not consecutive. Considering for instance a question in English like "Did John go to the park?", the words "did" and "go" would be aligned on the same tuple, causing troubles during the composition process that would halt because the operation 3.1 cannot deal with such cases. An alternative approach to eliminate the need of surface tuples is to automatically learn the non-semantic information them from the data. Experiments to test the feasibility of this solution are left for future work.

Chapter 4

A Semantically Annotated Corpus for Generation

The basis of any method that employs statistics to learn from existing data is in the big numbers. Automatic systems need to be fed a vast number of examples in order to learn useful patterns and be able to generalize and apply the learned information to new, unseen instances. The more information is provided to a learning system, the better it will perform at its task. In Natural Language Processing the data sets used for such purposes typically come in the form of text corpora, that is, large collection of digitized texts. Text corpora can be also enriched by manual or automatic annotation. The size of the collection, the coverage and depth of the annotation, the conventions and the theories vary wildly, as do many other factors such as language(s) genre, medium, purpose, and so on. The chapter on related work covers some of the attempts at building large resources for statistical NLP that have been made in the past decades.

From the standpoint of the statistical generation process, there is the need for a corpus that is not only large enough to be statistically relevant, but also enriched with deep semantic analysis. A sizable collection of natural language text paired with formulas expressed in a coherent, formal logic system represents a resource whose potential goes beyond that of the classical syntax-driven treebanks. For one thing, the semantic level of analysis of text is more abstract than syntactic analysis, closer to the abstract representation of information that would be input to a real-world NLG pipeline. As an indirect consequence, formal logic formulas are generally independent from a specific language, thus positioning semantically-motivated NLG in the context of Machine Translation.

At the time of the start of the work described in this thesis, there was no publicly available resource that encompassed all the characteristics of a corpus for statistical generation from logical forms, therefore the need to create such a resource. The Groningen Meaning Bank (GMB) represents an effort to build a large, free corpus of modern English text annotated with several layers of information ranging from the boundaries of words and sentences to complete logical forms representing entire documents.

Building a resource with such features is far from being a trivial task. Regardless, in this chapter I prove that it is indeed possible to construct a semantically annotated corpus to be employed in the framework of statistical general-purpose NLG, by reporting about the work, started in early 2011, that had led to the creation of the GMB. The other important question is about the quality of the data, that is, how to ensure a gold standard level of linguistic annotation. The final part of this chapter will cover how this aspect is approached by means of two complementary strategies.[1]

4.1 The Groningen Meaning Bank

In this chapter I detail the story of how the Groningen Meaning Bank has been conceived, designed and made into existence. Before going further, it is important to clarify that this is by no means a one man job. The GMB is the result of the joint effort of many people, including the co-authors of the articles cited throughout this chapter, external experts on linguistics and semantics that helped with the annotation, and anonymous Internet users from around the world that played our linguistic games. Secondly, the GMB is, and it will probably be for a long time, a work in progress. While the resource has been already employed in other people's work and it is proving its usefulness, it also keeps growing in new directions, be it gathering more raw data, annotation from new sources, or software components added to the existing infrastructure.

First of all, I will try and answer the obvious question that might come to mind: do we really need a new resource like the GMB? As we will see, although the GMB is not the first of its kind, some of its design principles make it stand out as an innovative resource. Later in the chapter I describe the step taken to build the GMB. At a glance, we

- collected public domain English text from the Web
- constructed an integrated software system to automatically analyze the text and produce linguistic annotation

[1] This chapter is based on the work published in Basile et al. [2012a], Basile et al. [2012b] and Venhuizen et al. [2013a]. Throughout this chapter the pronoun *we* refers to the authors of said articles.

- created a collaborative Web interface to allow expert linguists to review and correct the computer-generated annotation
- created a series of computer games to gather linguistic knowledge from the non-expert crowd in a cheap way

For each of these steps several design decisions had to be made, and in some cases a considerable amount of software has been written. Nevertheless, the GMB is currently a stable resource, publicly available, that can be (and has been already) employed for many purposes by the NLP community. One of these purposes is in fact the main subject of this dissertation.

4.1.1 Related Work

Text corpora are not a novel idea. Perhaps the most popular is the PennTreeBank [Marcus et al., 1993], a large corpus annotated with syntactic structures. In the PTB, sentences from various sources are annotated with skeletal parses, i.e., tree-like structures showing dependencies between the words. Countless statistical models have been trained on the PTB data, which can be considered the *de facto* standard in the world of annotated text corpora. Despite its success, there is space for improvement in a corpus like the PTB. For start, we now have the tools to perform deeper analysis of text, that goes beyond the syntactic dependencies and towards the translation of text into purely semantic logical forms.

In Chapter 2, I reviewed some semantic representation formalisms that could serve as an alternative to DRT, in particular Minimal Recursion Logic and Lexical Functional Grammar. The problem with using MRS for statistical generation, despite the expressiveness of the formalism, is that, as of this writing, there is no large semantically annotated corpus available to train a model,. As we will see in the rest of this chapter, even if one would employ a reasonably accurate MRS-based semantic parser to a large collection of documents, there would still be the need for some kind of expert check — a task that requires a non-trivial effort. PARC 700 King et al. [2003] is a *dependency bank* consisting of sentences parsed semi-automatically to provide LFG-based representations. Unfortunately its size (700 sentences, hence the name) is orders of magnitudes smaller than what is needed for robust statistical generation. Finally, Ontonotes Hovy et al. [2006] is a resource comparable in size to the GMB that has been annotated with syntactic parse trees, predicate-argument structure and shallow semantics (the word senses are linked to a foundational ontology). The lack of a deep semantic level of analysis in Ontonotes, however, makes it unsuitable for serving as training material for a NLG pipeline that aims at generating from logical forms.

4.1.2 Motivation

Many factors vary between annotated language resources. A very large number of text corpora exist nowadays that have all kind of different features. When designing the GMB, we decided to focus on **written** text. Moreover we consider only **descriptive** kind of text, that is, not containing dialogue and direct speech. These two restrictions works towards getting better analyses from the computer tools, since the more advanced tools to date are more accurate when applied to this type of text. Regarding languages, the GMB is designed to be a multi-language resource, with the long term goal of being suitable for machine translation. However, so far only **English** documents are included in the collection, mostly because of the language analysis tools only being available for that language. Most of the work in statistical NLP, before the GMB, has been carried out on existing annotated corpora.

Thanks to the software infrastructure described later in this chapter, the GMB comprises analyses that integrate many linguistic phenomena, from the "superficial" ones such as morphology and part-of-speech, to deeper structures representing concepts, predicates and relations. A level of linguistic knowledge that is missing in many state-of-the-art corpora is that of **discourse**. Most existing resources focus on the analysis of isolated sentences, leaving out discourse relations such as causation, explanation, etc. A notable exception is the Penn Discourse TreeBank, a resource built on top of the PTB annotated with many kinds of discourse relations. Unfortunately, the additional layer provided by the PDTB is not sufficient to overcome some of the limitations of the PTB, for instance cross-sentence anaphora are still untreated. The GMB, instead, provides single analyses for complete documents, one analysis per document, including the semantics of discourse units as well as their relations.

The analyses of the GMB are constructed in a way that makes it easy to align them to the original text, that is, linking the text with its meaning with fine granularity. This is particularly useful for the system subject of this thesis, but also for every system that tries to models computationally the relationship between words and their meaning.

Last but not least, most existing corpora provide the analysis of texts that are subject to some form of copyright restrictions. Even when the annotation is freely distributed, the underlying text often cannot be distributed as well, making the whole process more complicated. In the GMB, we explicitly decided to include only **public domain** text such as, for instance, products of government enterprises. This way, we ensure that the GMB is not only accessible, but redistributable in its entirety.

Besides the final result, the methodology for creating the GMB is innovative on its own. At no point during the development we relied on classic manual annotation in the form of trained subjects going over the whole corpus and analyzing everything to produce a gold standard. Our crowdsourcing approach made necessary to take some measures to ensure that the resulting annotation has high quality, but at the same time allowed us to build a large and multi-layered corpus with relatively less effort.

4.1.3 A Collection of Public Domain Text

The foundation of an annotated text corpus is of course the text itself. In many cases the textual data of a corpus is chosen among a particular genre (e.g., bio-medical text), medium (speech/written) or linguistic phenomena (e.g., pair of sentences for RTE tasks). The problem of where to find the text might seem a trivial aspect at first but, when the aim is a multi-genre, domain-independent corpus, finding a reliable and abundant source for data is a challenge in itself. Fortunately, in the era of the Web the collection of large quantities of text has become, at least at the technical level, a feasible task. Still, one cannot just run a program and download whatever text comes to mind, as accessibility, appropriateness, language usage, size, and many other factors are involved in the process.

One important consideration is that of the *terms of use* of a text. Written documents are in general products of human work and therefore their distribution is regulated under copyright laws. As we decided that one of the key feature of the GMB would have been the possibility to redistribute the resource in its entirety, we opted for the inclusion of only *public domain* text. This is a rather strong stance that severely limits the possible sources of raw data for our corpus. Nevertheless we concluded that it is the only way to ensure a durable resource that can be used by the community without any legal constraint.

4.1.4 Data Sources

We identified the first candidate source of data for the GMB in an online newspaper from the United States of America called *Voice of America* [2], for several reasons. Firstly, VoA is funded by the U.S. Government, so its content is by default in the public domain. Secondly, the structure of the mark-up used to publish the articles online is simple and consistent, allowing for a relatively painless process of scraping the text from the Web. Finally, being a collection of newswire text, VoA provides factual descriptions of events, a type of text that lends itself nicely to semantic analysis. We downloaded and cleaned tens of thousands of articles from VoA via custom Python scripts. The VoA articles spanning the period from 2005 to 2009 form the first iteration of the GMB.

To this day, VoA constitutes the majority of the data in the GMB, but in the meantime other sources (*subcorpora*) have been added to account for other genres of text. Here is a list of the additional sources of GMB documents:

- The **CIA World Factbook**, a "world manual" containing detailed information about every geo-political entity recognized by the United States. We downloaded parts of the CIA World Factbook, specifically the general descriptions of the countries and those of their economies. The CIA World Factbook provides relatively short descriptive texts written in plain English and contains a notably large number of named entities.

- **MASC** (Manually Annotated Sub Corpus), a subset of the Open American National Corpus. By including the MASC into the GMB we leverage the work of the experts that annotated the corpus in the MASC project. The texts are of various nature, with some of them unsuitable for the GMB in its present form, e.g., containing direct speech or heavy formatting.

- English translations of ancient Greek **fables**. These texts are typically written in a simple English, as they are intended for an audience that includes children, and are of reasonable size. They also present interesting challenges for the semanticist, in particular for classification of entities, such as animals that speak as if they were humans.

- A collection of **jokes** downloaded from the Web. These are short humorous stories with no copyright claim that help balance the topics in the corpus.

[2] http://www.voanews.com/

- Data from the Third Recognising Textual Entailment Challenge. [3] This is a collection of pairs of short texts manually created to train and test RTE systems, that is, methods to identify semantic entailment.

4.1.5 Structure of the GMB Data

When including new data in the GMB, we follow a specific process that starts with downloading (or otherwise coming into possession of) the text, and subsequently organizing them following the standard of the GMB: one directory per document containing two text files, i.e., the document itself and a file with the metadata information. The document directories are in turn organized in 100 distinct parts, to prevent the number of files in one directory to grow excessively. Moreover, the division of the corpus in parts is useful in other contexts such as dividing the data in training and test set, or focusing the annotation on a subset of the documents. New documents are added to the collection in an horizontal fashion, so to preserve the balance of genres across the parts.

The documents in the GMB are labeled following their **status**. Newly included documents are labeled *uncategorized* and are then subject to manual scan. If a document is found unsuitable for the GMB, for instance because contains corrupt data, duplicate text or perhaps offensive content, it is *rejected*. Some times a document can be difficult to process due to a limitation in the current software employed for the analysis, thus it is marked *postponed*. Finally, if the document passes the manual test, it is classified as *accepted* and as such it becomes immediately accessible online and considered as candidate to be included in the next release. At the time of this writing, the GMB comprises 76,648 documents in total, of which 11,441 are accepted, 6,343 are postponed and 459 are rejected.

4.2 The Annotation of the GMB

The GMB aims at being a self-contained resource while retaining interoperability with complementary resources. Each text in the GMB is coupled with several layers of annotation, which I review in this Section. Different annotation layers follow different formats, but they all have in common the property of being *stand-off* annotations, that is, the original text is never modified and instead new files are created that specify bits of linguistic knowledge linked to portions of the text.

[3] http://pascallin.ecs.soton.ac.uk/Challenges/RTE3/Datasets/

4.2.1 Word and Sentence Boundaries

Before proceeding with any form of linguistic analysis, the researcher or the software system must identify the elementary units of text. In most cases, these are documents, sentences and words. The GMB is created by collecting single documents, thus the first level of subdivision is already in place. There remain two distinct tasks to perform namely sentence splitting and word boundary detection, which respectively divide a text into sentences and a sentence into words.

In the GMB, the boundaries of words and sentences are marked at the character level, i.e., each single character is assigned a label. In practice, the GMB employs a modified IOB tagging scheme, where characters are given one of four possible labels: S (start of a sentence), T (start of a token), I (inside a token) or O (outside tokens). This simple scheme not only casts the tokenization problem in a format suitable to statistical methods, but also includes the other task of sentence boundary detection in a natural way, by identifying the starting word of each sentence.

```
It didn’t matter if the faces were male,
SIOTIITIIOTIIIIIOTIOTIIOTIIIIOTIIIOTIIITO
female or those of children. Eighty-
TIIIIIOTIOTIIIIOTIOTIIIIIIITOSIIIIIIO
three percent of people in the 30-to-34
IIIIIOTIIIIIIOTIOTIIIIIOTIOTIIOTIIIIIIIIO
year old age range gave correct responses.
TIIIOTIIOTIIOTIIIIOTIIIOTIIIIIIOTIIIIIIIIT
```

Figure 4.1: Example of IOST-labeled characters

This scheme offers some nice features, such as allowing for discontinuous tokens (e.g., hyphenated words at line breaks) and starting a new token in the middle of a typographic word if the tokenization scheme requires it, e.g., as in *did|n't*. An example of the tokenization scheme is given in Figure 4.1.

4.2.2 Lexical Items

Words carry information on many levels, from their form to their meaning, to their origin and usage in context. A complete description of these multiple layers and their interconnections belongs to the field of lexicography and is way too vast to be presented in this thesis. The annotation schema of the GMB reflects this characteristic of words by providing several parallel layers of word-level annotation.

To each word[4], defined by the boundaries described in the previous Section, corresponds a list of tags, one for each level of linguistic knowledge described as follows.

Perhaps the most informative way of distinguishing the function of different words is looking at their **part-of-speech (POS)**, i.e., the basic category of the word. In the GMB we use the part-of-speech tagset of CCGbank [Hockenmaier and Steedman, 2007], which is a slight variant of the Penn Treebank tag set comprising 36 POS tags and 12 other tags for punctuation and other symbols. The Penn TreeBank tag set is very articulate, and in some cases too fine grained for specific applications such as for instance word sense disambiguation, so at times the POS tags are reduced to the smaller sets {noun, adjective, verb, adverb}.

The words of the GMB documents contains other levels of information, each encoded in its particular layer of annotation. Words (in English, at least) have base forms and inflections for plurals, past tenses of verbs and so on, as in *cars* (word form) → *car* (lemma). We annotate this information identifying the **lemma** of each word.

Moving forward to more semantic-oriented lexical information, the words representing **named entities** are also tagged as such. When a word or phrase is used to directly indicate a specific instance of an object, person, etc., it is labeled with a tag specifying what type of named entity it refers to. Named entity classification is further explained in Section 4.2.3.

Some words may refer to animate, inanimate, or partly animate entities. This information, referred to as **animacy**, can be useful for the interpretation of anaphoric expressions, for instance. The tagset for animacy used in the GMB, based on Zaenen et al. [2004], is further detailed later in the next section.

Word senses are a fundamental building block to give actual content to a semantic representation. In the GMB we decided to represent the meaning of words with WordNet synsets Miller [1995]. WordNet is an electronic resource of lexical knowledge where words are linked together by several kinds of relations, including synonymy. Each lemma can then be part of many sets of synonyms (synsets) that define its different meanings. Representing lexical meaning as synsets in WordNet has proved to be also useful for the main NLG-related goal of this thesis. The characteristics of the WordNet words and meanings network will be further explored in Chapter 6. In the GMB, each eligible word is linked to the WordNet synset corresponding to its meaning in the particular context.

[4] Throughout this chapter, sometimes the terms *word* and *word token*, as defined in the previous section, are used interchangeably.

Finally, there are layers of linguistic information that goes beyond the word level, such as thematic roles and anaphoric structures. Although we could (and for practical purposes we do) annotate them at the word level, with references to the other entities involved in the annotation, they are not just lexical information, strictly speaking. I discuss these semantic layers of annotation at the end of this section.

4.2.3 Named Entity Classes and Animacy

When doing hard classification of semantic entities, where exactly one label is assigned to each, it is important to choose the right level of granularity. Even though we leveraged existing work in ontology definition for named entities, we found that there are exceptions to every rule, so the final scheme for named entity tagging of the GMB ended up as the result of extensive discussion and trade-offs. The annotation scheme for named entities used in the GMB is based on the shallow classification provided by ACE [Doddington et al., 2004], the international program for developing advanced Information Extraction technologies. We adopted three of their basic categories, and added four other categories inspired by Satoshi Sekine's Extended Named Entity Hierarchy [Sekine et al., 2002]. This results in the following classification:

- Person (PER) - Person entities are limited to individuals that are human or have human characteristics, such as divine entities.
- Organization (ORG) - Organization entities are limited to corporations, agencies, and other groups of people defined by an established organizational structure.
- Location (LOC) - Location entities are limited to geographical entities such as geographical areas and landmasses, bodies of water, and geological formations.
- Artifact (ART) - Artifacts are limited to man-made objects, structures and abstract entities, including buildings, facilities, art and scientific theories.
- Natural Object (NAT) - Natural objects are entities that occur naturally and are not man-made, such as diseases, biological entities and other living things.
- Event (EVE) - Events are incidents and occasions that occur during a particular time.
- Time (TIM) - Time entities are limited to references to certain temporal entities that have a name, such as the days of the week and months of a year.

These seven basic entities are considered to cover all named entities. Currently, all entities receive one named entity tag, but there are infamous ambiguous cases. In order to reduce such ambiguity, we add an extra category for Geo-political entities, which is interpreted as a hybrid tag for Location and Organization:

- Geo-political Entity (GPE) - GPE entities are geographical regions defined by political and/or social groups. A GPE entity subsumes and does not distinguish between a city, a nation, its region, its government, or its people (LOC•ORG).

Technically, each word token in the corpus is labeled independently, so no explicit multi-word named entity is tagged as such. In named entities consisting of multiple word tokens, each word token is tagged (e.g., New | LOC York | LOC). While this scheme works with "simple" multi-word named entities such as *Barack Obama* (Person), it is insufficient to fully represent complex expressions where different types of entities are present, such as *Los Angeles Lakers* where the sub-expression *Los Angeles* is a city (Location) but the complete phrase refers to a sport team (Organization). In nested named entities like *Los Angeles Lakers*, each word token is tagged with the tag appropriate for the outermost named entity that it is part of (e.g., New | ORG York | ORG Yankees | ORG).

In general, only entities with part-of-speech tag NNP or NNPS receive a named entity tag. Exceptions to this rule are demonyms, which have POS tag JJ (e.g., American | ORG), and long names consisting of word tokens with different POS tags (e.g., Paradise | ART By | ART The | ART Dashboard | ART Light | ART).

The animacy layer of annotation is separate from the named entity layer, but they are of course related. For instance, a named entity of type Person would be almost always tagged as Human. The complete list of animacy tags with examples is the following, based on Zaenen et al. [2004] with examples from the GMB:

- Human - Mr. **Calderon** said Mexico has become a worldwide leader ...
- Organization - Mr. Calderon said **Mexico** has become a worldwide leader ...
- Animal - There are only about 1,600 **pandas** still living in the wild in China.
- Place - There are only about 1,600 pandas still living in the wild in **China**.
- Non-concrete - There are only about 1,600 pandas still living in the **wild** in China.
- Concrete - The wind blew so much **dust** around the field today.

- Time - The wind blew so much dust around the field **today**.
- Machine - The astronauts attached the **robot**, called Dextre, to the ...
- Vehicle - Troops fired on the two civilians riding a **motorcycle** ...

In the context of statstical NLG, named entity classes and animacy are features that can be exploited in lexicalization (and possibly other tasks. The choice of words may differ depending on what kind of entities are involved in the situation encoded in a formal meaning representation, for which the NLG system has to produce a natural language expression. For instance, *China*, as an Organization, could *ratify* a treaty, while its president, as a person, could be said to *sign* it. Similarly, animacy classes could influence the lexical choice and surface order of a target realization. In the next chapters I will show the results of experiments that help to determine the precise effect of the semantic information on the NLG output.

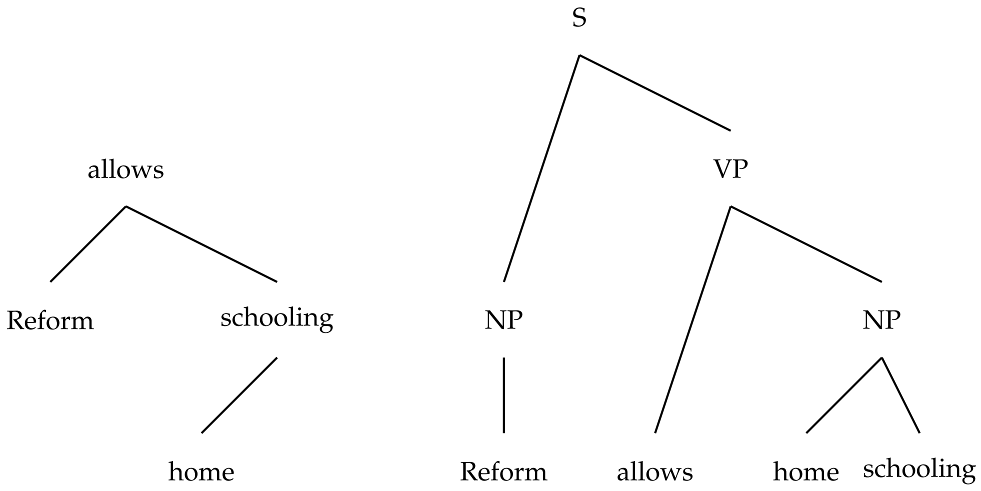


Figure 4.2: Dependency tree (left) and constituent tree (right) for the sentence "Reform allows home schooling".

4.2.4 Syntactic Parse Trees

Above the level of the words, in terms of linguistic analysis, there is the level of the sentences and their syntactic structure. In the literature there is plenty of grammars to choose from when it comes to encoding the syntax of natural language, which mainly belong to one of the two families of *dependency grammars* and *constituent grammars* (also called *phrase structure grammars*). The grammars of the former group define links between words, while the grammars of the latter express rules for combining the words into hierarchical structures [Tesnière, 1959, Chomsky, 1957]. In both cases, the syntactic analysis of a text takes the shape of a *parse tree* with words as the leaves, with the difference that every node of a dependency parse is a word, while in a constituent parse tree words are only the leaves of the tree and inner nodes are other kind of symbols, depending on the specific grammar . Figure 4.2 shows an example of parse trees based on different types of grammars for the same sentence.

Categorial grammars [Adjukiewicz, 1935, Bar-Hillel, 1953] are an instance of constituent grammars. These kind of grammars assigns categories to the lexical elements that can be either atomic or composed by mean of two directional application operator *slash* ("/") and *backslash* ("\"). For example, a noun can be tagged N (atomic category) and an adjective preceding it N/N, meaning *takes an N on its right and returns an N*. The functional application can be nested on multiple levels, with categories of increasing complexity such as (S\NP)/NP, ((S\NP)/PP)/NP, ((S[dcl]\NP)\(S[dcl]\NP))/NP, and so forth, with the help of the parenthesis for disambiguation.

A categorial grammar lends itself extremely well to the syntactic annotation of the GMB documents because it is lexically driven, which means that the categories are assigned at the word token level, just like for the other layers of annotation. Moreover, a categorial grammar has only few grammar rules, and and its type-transparency principle, which says that each syntactic type (a grammar category) corresponds to a unique semantic type, results very useful to the subsequent step of automatic annotation (see next section).

The syntactic formalism used in the GMB is Combinatory Categorial Grammar (CCG) [Steedman, 2001]. CCG is a variant of categorial grammar in which additional operators besides functional application allows the grammar to account for linguistic phenomena traditionally considered problematic. These operators include left and right *composition* ($B_<$ and $B_>$) and left and right type raising ($T_<$ and $T_>$). Composition rules, for instance, allow longer sequences, e.g., coordination, to be constituents while they would not be in standard categorial grammar. Type-raising rules turn arguments into functions over functions-over-such-arguments, thus allowing for constructions such as relative clauses.

In the annotation of the GMB, we employ the CCGbank's [Hockenmaier and

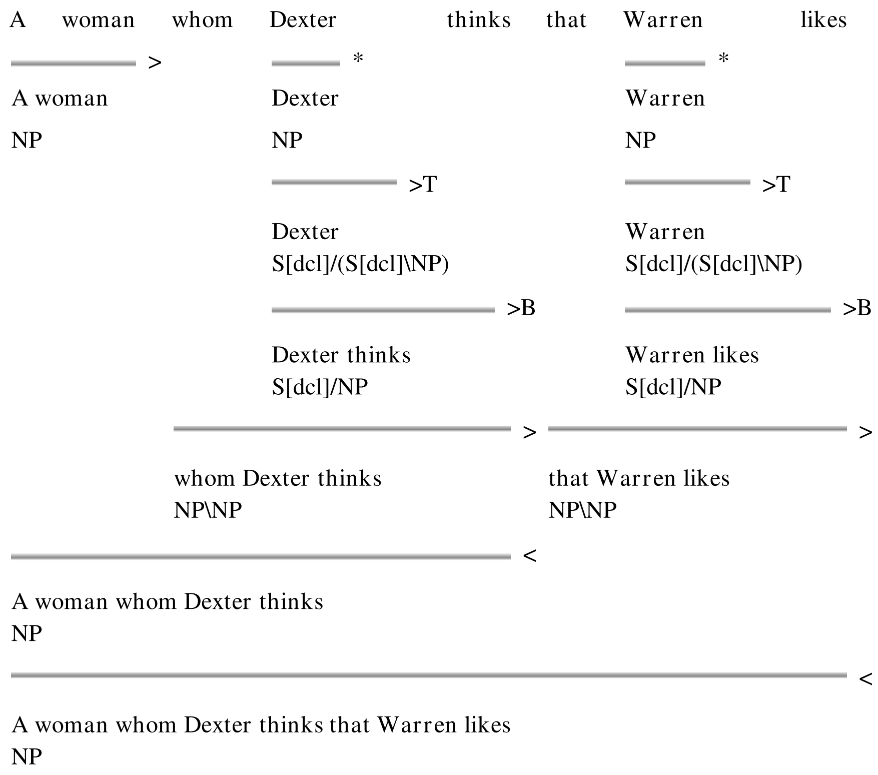


Figure 4.3: An example of a CCG derivation.

Steedman, 2007] flavor of CCG. Notably, this version of CCG includes feature annotations on some atomic categories, which are appended to them in square brackets. An example CCG parse tree including type checking and composition is depicted in Figure 4.3.

The CCG category annotation of the GMB, and the syntactic layer in general, is an important gear in the mechanism that automatically derives full semantic representations of the natural language documents included in the corpus. Nonetheless, syntax do not directly play a role in the generation process presented in this thesis. The Unboxer approach aims at constructing a supervised model of the alignment between text and a formal representation of its meaning expressed as logical form, thus if any syntactic representation is derived, it is at most an implicit one.

Table 4.1: List of basic conditions in DRT

Condition	Example
predicate	HURRICANE(x_1), STRONG(x_1), HIT(e)
relation	AGENT(e,x), in(e, t)
named entity	NAMED(x_2,Katrina)
time expression	TIMEX(t, 2005)
cardinality	CARD(x_3, 1833)
equality	EC(x_1, x_2)

4.2.5 Semantics and Discourse

One of the main goals of the GMB is to provide the semantic representation of a text in a single, integrated formalism. Moreover, the ideal semantic analysis goes beyond the level of isolated sentences with the integration of discourse-related information. For these reasons we chooose to root the formalism for the semantic representation in the GMB on one of the most well-known theory based on dynamic semantics: *Discourse Representation Theory* [Kamp, 1984]. DRT is a powerful enough logic formalism to effectively represent many linguistic phenomena. In DRT, the meaning of natural language expressions is represented as logical formulas called Discourse Representation Structures (DRSs for short). A DRS, typically depicted as a box, is composed of an upper part containing the *discourse referents* and a lower part containing the *conditions* over the referents. The conditions can be simple, that is, predicates and relations between discourse referents, or complex, when the operands are DRSs themselves, thus making DRSs recursive structures. First-order logic operators such as implication, negation, equality, conjunction and disjunction, can be also part of a DRS. In fact, a DRS can be translated directly into a FOL formula [Muskens, 1996]. One-place predicates over discourse referents define their type as concepts, events or attributes. We will see in Chapters 5 and 6 how this distinction is important for generation, as producing the surface form for different referents depends also on their type. Other then predicates, DRT's basic conditions include named relations, entities, time expressions, cardinality and equality. The possible conditions found in the GMB's semantic layer are listed in Table 4.1.

Two-place predicates on pairs of discourse referents encode relations between them, including **thematic roles**. Together with word senses, thematic roles of events are central to formalize the semantic of an expression. In the GMB the thematic roles are annotated following the role inventory of VerbNet Kipper et al. [2008]. Linguistic structures like noun-noun compound contain, at the semantic level, **relations**

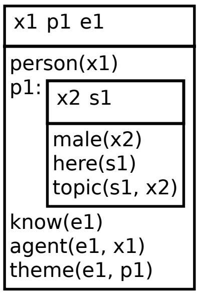


Figure 4.4: DRS for the sentence “I know he is here”.

between their constituents. For instance, an *export trade* is a trade *for* export, while a *reform agenda* could be an agenda *on* reforms. Currently, in the GMB, constituents of a compound or a genitival construction are tagged with one of a closes class identifier, e.g., a preposition. Since each discourse unit is identified by its unique discourse referent, they can be arguments of relations. This is necessary, for instance, to represent subordinate clauses.

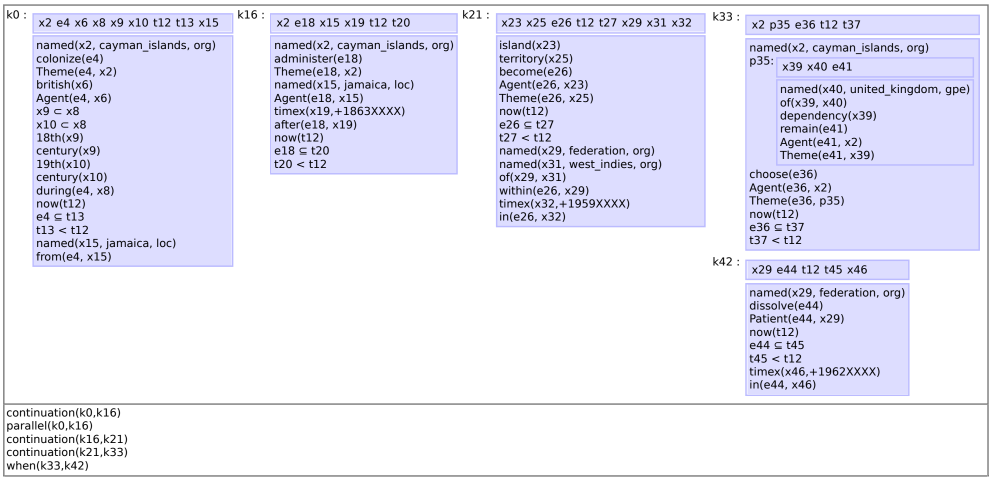


Figure 4.5: An example of the semantic representations in the GMB, with DRSs representing discourse units.

Consider the DRS in Figure 4.4 for the sentence “I know he is here” and note the last condition THEME(e_1, p_1). The explicit scope information that discourse referents carry in a DRS makes it possible to account for anaphoric expressions. Referring again to the example in Figure 4.4, the referent x_2 is defined in the innermost DRS, thus it is not accessible from the outermost one. In the GMB, **anaphoric expressions** such as pronouns are linked to the entity they refer to, both in the DRS and at the word level by specifying the boundaries of the referred word or phrase.

In order to account for the semantics of whole texts, the GMB employs an extension of DRT called *Segmented Discourse Representation Theory* [Asher and Lascarides, 2003]. SDRT augments DRT with a set of *discourse relations* that allow to produce one single representation for a text made by multiple sentences by putting them in explicit relations. In a Segmented DRS (SDRS) discourse units are bound together by binary discourse relations such as *continuation* (e.g., one sentence simply follows another), *elaboration* (e.g., a sentence explains the previous one) and so on.

Figure 4.5 shows an example of a (S)DRS from the GMB, whose original text is "The Cayman Islands were colonized from Jamaica by the British during the 18th and 19th centuries and were administered by Jamaica after 1863. In 1959, the islands became a territory within the Federation of the West Indies. When the Federation dissolved in 1962, the Cayman Islands chose to remain a British dependency".

Finally, extensions to DRT provide a representation for verb tenses, cardinality of entities and presupposition. Van der Sandt [1992a]. A recent study on the generalization of treatment of projective phenomena such as presupposition led to a new theory that extends DRT called Projective Discourse Representation Theory Venhuizen et al. [2013b]. In a near future, PDRT will become the new standard logic formalism to represent meaning in the GMB.

4.3 A Toolchain for Automatic Annotation

Building a full stack of linguistic annotation on top of a large collection of documents is a complex task. Fortunately, a set of software packages for a variety of tasks were available at the time when the GMB was created. We still had to do the work of assembling the tools into an organic architecture, making them inter-operable with common formats, and run the resulting system on the whole GMB text corpus.

In this Section I describe the components that form the pipeline of tools that provides the automatic linguistic annotation on the GMB. I will also provide a description of the general formalism we devised for the annotation that makes it possible to integrate linguistic knowledge from several sources (including human expertise) into the automatic process.

4.3.1 Word and Sentence Segmentation

The first step towards the linguistic analysis of a document in the GMB is the tokenization, that is, the process of breaking down the linear string of characters into meaningful units.

The automatic detection of boundaries of sentences has been addressed in the past and many solutions are found in literature, for instance Punkt [Kiss and Strunk, 2006] rely on statistical methods to learn the distinction between the dot in an abbreviation ("Mr.") and a sentence-ending full stop. The segmentation of a text into sentences, and of a sentence into words, commonly referred to as *tokenization*, is considered to be an almost "solved" problem, e.g., by Dridan and Oepen [2012], by means of rule-base approaches, but there are often corner cases that require additional rules for specific languages and domains.

At the beginning of the development of the GMB we chose to use the tokenization script included in the C&C tools called *t* (just one letter), which is in turn a replacement of the original C&C tools tokenizer *tokkie*. *t* consists of a series of manually crafted rules implemented as Prolog clauses, and performs both tasks of sentence and word boundary detection The GMB approach to tokanization is not based on rules, but rather on a stochastic system, named Elephant [Evang et al., 2013], developed with the Wapiti implementation of Conditional Random Fields [Lafferty et al., 2001, Lavergne et al., 2010], using as features the output label of each character, combined with 1) the character itself, 2) the output label on the previous character, 3) characters and/or their Unicode categories from context windows of varying sizes. For example, with a context size of 3, in Figure 4.1, features for the E in *Eighty-three* with the output label S would be E/S, O/S, ␣/S, i/S, Space/S, Lowercase/S. The intuition is that the 31 existing Unicode categories can generalize across similar characters whereas character features can identify specific contexts such as abbreviations or contractions (e.g., *didn't*).

In addition to character n-gram features, Elephant makes use of information extracted from the text in a fully unsupervised way, that is, a neural network-based feature learning approach. The additional representations correspond to the activation of the hidden layer in a simple recurrent neural (SRN) network [Elman, 1990, 1991], implemented in a customized version of Mikolov et al. [2010]'s RNNLM toolkit. The network is sequentially presented a large amount of raw text and learns to predict the next character in the sequence. It uses the units in the hidden layer to store a generalized representation of the recent history.

In order to test the Elephant method, we used context windows of size 0, 1, 3, 5, 7, 9, 11 and 13, centered around the focus character. The best models (window size of 7 for English and Dutch, 11 for Italian, with the addition of SRN-based features) obtained error rates of 0.027% (English), 0.035% (Dutch) and 0.076% (Italian).

A comparison with other approaches, in particular with rule-based methods, is hard because of the difference in datasets and task definition (combined word/sentence segmentation). We compared the performance of Elephant only for sentence segmentation with Punkt. With its standard distributed models, Punkt achieves 98.51% on out English test set, 98.87% on Dutch and 98.34% on Italian, compared with respectively 100%, 99.54% and 99.51% of Elephant.

In Evang et al. [2013] further details can be found on the methodology and implementation of the Elephant system, along with the results of experimental tests on several languages.

Table 4.2: Occurrence rate of several part-of-speech tags in the GMB.

POS tag	**part-of-speech**	**occurrences in the GMB**
NN	common noun	2,713,594 (17.861)
NNP	proper noun	1,675,408 (11.028)
IN	Preposition	1,648,542 (10.851)
JJ	adjective	1,189,555 (7.830)
DT	determiner	949,180 (5.971)
.	full stop	716,418 (4.716)
VB	verb, base form	418,205 (2.753)
RB	Adverb	363,427 (2.392)
VBD	verb, past tense	200,531 (1.320)
MD	modal	173,564 (1.142)
PRP	personal pronoun	140,753 (0.926)
FW	foreign word	3,104 (0.020)
SYM	symbol	738 (0.005)
UH	interjection	366 (0.002)

4.3.2 Tagging of Lexical Units

Once word boundaries have been defined, we can shift the focus of the analysis to the word level. Words, as a basic unit of language, carry a great deal of information often in a multi-dimensional way, that is, one word have different facets that are important to different kinds of analysis, as seen in the previous Section. This stratified nature of lexical information is reflected in an array of labels attached to them. POS-tagging is performed by the Entropy-Maximization tagger included in the C&C tools. To give an idea of the distribution of the POS tags we found in the GMB corpus, the occurrence rate of some part-of-speech is shown in Table 4.2.

The morphological information is annotated at the word level with just the lemma as returned by the Morpha [Minnen et al., 2001] software for morphological analysis of English. Morpha takes as input a word and its context and returns its lemma and its inflection in the form of one of the four possible suffixes -s, -ing, -ed and -en. For instance, given the word *living* as in "he had been living in Afghanistan for years", Morpha analyses it as live (lemma) + -ing (inflection). Morpha is part of a set of tools comprising also Morphg, a software that implements the inverse process of Morpha for generation purposes. Morphg takes a lemma and an inflection suffix, e.g., novelty + -s becomes *novelties*. Morpha and Morphg also works with irregular verb tenses like *went* (go + -ed) and similar irregular forms.

Named entities in the GMB are tagged, like parts-of-speech, by an Entropy-Maximization named-entity tagger included in the C&C suite [Curran and Clark, 2003]. The tagger is trained on the MUC-7 data set [Chinchor and Robinson, 1997], originally compiled for the series of shared tasks of the Message Understanding Conference.

The classification of animacy is carried out by a multiclass logistic regression model implementing uses a one-vs.-all (OvA) scheme. The animacy classifier is trained on data extracted from the NXT Switchboard corpus, as well as some of the gold data in the GMB.

Word senses in the GMB are tagged trivially by their most frequent sense, using WordNet synset as sense inventory. This is known to be a very strong baseline for the task of word sense disambiguation — see for instance Preiss et al. [2009]. Despite the availability of off-the-shelf word sense disambiguation systems for English, this particular component has not been integrated in the pipeline so far. The rationale behind this choice is that WSD is a very difficult task, highly dependent on the contexts of the target words to disambiguate. The annotation of word senses is still possible though, either manually or by means of automatic scripts. In the next sections the capabilities of the interface to do so are presented in detail.

The last piece of information at the lexical level is the syntactic category, later used by the parser to derive its syntactic analysis. In the GMB we use the supertagger included in the C&C tools to assign the categories to the words. The supertagger is trained on the CCGBank data just as the POS tagger.

4.3.3 The C&C Parser

The syntactic analysis, usually referred to as *parsing*, is a very important step in the pipeline that builds on the results of the previous steps of tagging the lexical units. The importance of having full parse trees of the sentences is given by the need for analysis that go beyond the single words. In other words, the meaning of a text cannot be computed by looking at the words in isolation. Syntactic analysis is also important in the practice of building the GMB, since it forms the basis upon which the semantic representation is then constructed.

The existence of CCGbank and the availability of robust parsers trained on it Clark and Curran [2004] makes CCG a practically motivated choice. The C&C tools suite provides a single executable that bundles the taggers and the parser, but also the single executables to run in isolation from one another. For the GMB annotation pipeline we chose to use the latter solution, where the increase in complexity of the operations is balanced by a higher flexibility that allows us to intertwine the execution of the single processes with inputs from external tools.

4.3.4 Boxer

Boxer Bos [2008] is a rule-based software component for semantic analysis of text, based on Combinatory Categorial Grammar and Discourse Representation Theory. Boxer constructs DRSs for English texts starting from CCG parse trees and several tags on the lexical units such as part-of-speech and named entity class. The name "Boxer" in fact refers to the shape of the DRSs, traditionally called "boxes" in an informal way.

Boxer operates by translating the syntactic categories of the CCG parse tree into semantic representations, defined as lambda-DRSs. Boxer implements almost all categories employed by the C&C parser, which is a subset of the ones found in CCGbank, leaving out extremely rare cases for the sake of efficiency. The additional information at the lexical level is necessary to define unambiguously the lexical semantics — one lexical category could give rise to several different semantic interpretations.

Presupposition phenomena are also accounted for, e.g., when analyzing different types of pronouns, proper names, or other anaphoric phenomena. A separate step, following the building of the semantic representation, takes care of the resolution of anaphora and presupposition. Bos [2003]

4.3.5 The Pipeline and the Daemon

In the previous sections I presented, one by one, the software components that forms the pipeline performing the automatic annotation of the GMB. What is left is how we assembled them all together to work as a whole, and the design of the automation procedure.

One of the first design challenges, at the beginning of the development of the GMB, was the need for an infrastructure capable of processing a large number of documents in reasonable time. A fair assumption in cases like this is that redundancy of operations is an undesirable property, that is, the same operation on the same document should not be performed unless a different outcome is expected. We identified a candidate for a system with these characteristics in the software tool *GNU Make*.

Make is a popular tool used in a vast majority of software distributions to automate the processes of compilation, linking, testing and collateral operations. However, Make is a more general tool, capable of orchestrating many interdependent tasks. In a way, our usage of Make for the GMB pipeline is unorthodox, having nothing to do with compilation of source code, but the tool adapted perfectly to our needs, which still consist of handling a series of operations on text-based files.

Make is designed to look at the time of the last modification of files to avoid repeating unnecessary operations. By producing intermediate result files at each step of the pipeline, we make sure that the only operation performed by the tools are the ones with the potential to yield new results, leaving the necessary checks to Make.

Over time, the software infrastructure around the GMB pipeline has grown, while GNU Make has remained its pivotal component. To increase the level of automation, we developed an additional component that executes the Make-based architecture periodically in a fully automated way. This daemon[5] ensure that all the analyses are updated at any time, for instance when a new version of one of the tools is available, with a delay of a few hours at most.

[5] *daemon* is slang for a computer program that runs as a background process.

4.4 Manual Correction: Experts and the Crowd

The automatic annotation performed by our pipeline of linguistic analysis tools is fairly accurate, even though the quality of the annotation varies with the different layers. To obtain perfectly correct annotation, one cannot rely on automatic systems only, but, on the other hand, manual annotating a whole corpus the size of the GMB is a gigantic task. To help alleviate the issues that come with manual annotation, we devised two parallel ways of obtaining linguistic knowledge from human subjects. The first is an open, wiki-like web interface to access, review and modify the GMB. Experts in linguistics and logic can provide their annotations on one or more phenomena in a collaborative effort to produce a corpus of increasingly better quality. The second tool we developed is an online game with which we collect data generated by players and distill them into proper linguistic annotations.

4.4.1 Silver Standard and Gold Standard

The two main points of having an annotated text corpus are its sheer size and the fact that the annotation is correct. Ideally, the annotation is performed or at least checked by human experts — this is referred to as *gold standard* annotation. Unfortunately one cannot have the cake and eat it when it comes to annotated corpora, that is, the more data is included in the collection the harder and more time-consuming would be to obtain a gold standard. This is complicated even more by the desire to have multiple layers of linguistic knowledge which require not only more manual work but also different expertise.

We have designed the GMB to be an ever-growing resource. To this day, batches of new documents are added to the collection and processed. More important, the automatic annotation is continuously improved with the inclusion of linguistic knowledge coming from human judgment. The quality level of the annotation of the GMB is then *silver* standard, because of the mixture of automatic and human-provided annotation.

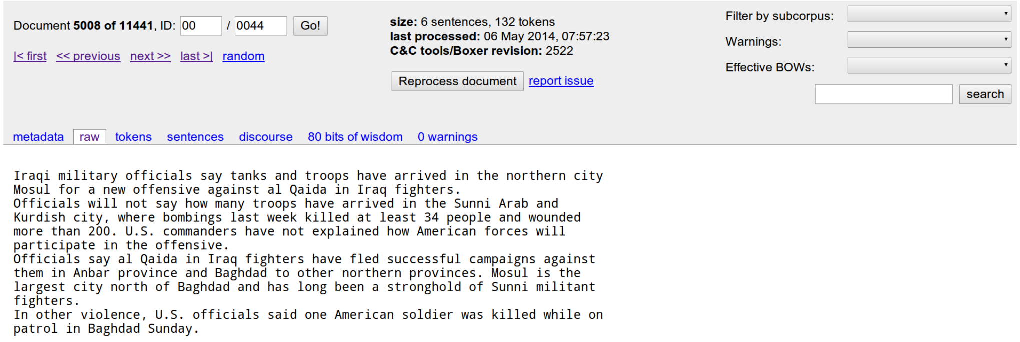
But how is this correction actually carried out? The first step is to define a meaningful atomic unit of linguistic annotation. We called this unit a Bit Of Wisdom (BOW). A BOW is essentially a fact stated on a specific span of text that refers to one layer of annotation. For instance, a BOW can encode that the part-of-speech annotation of *causes* is a verb (3rd person singular present) in the sentence "Smoking causes diseases".

BOWs for tokenization and sentence boundaries have a different structure, where they simply state that a certain character, specified by its offset in the text, has one of the labels described in Section 4.2.1. As an example, a tokenization that wrongly assumes the full stop is a sentence ending in an abbreviation such as "Mr. Basile" can be corrected with a BOW asserting that the 4th character (the capital B) is the start of a word rather than the start of a sentence.

4.4.2 The GMB Explorer

Text corpora in electronic formats are typically distributed digitally, altought every resource comes with its own file types and license. The most basic way of distributing a corpus is perhaps to make available its files for download, thus enabling other scholars to manipulate and study the text and annotation with the most suitable tools. Free access to the raw data files is also crucial to ensure the reproducibility of experiments.

While machine-readable formats are ideal for corpus studies, sometimes it is convenient to provide a more human-friendly tool to access a corpus. Many resources today are associated with some kind of interactive interface, often Web-based, for many reasons including ease of querying the corpus or sometimes to restrict the access to the raw data.



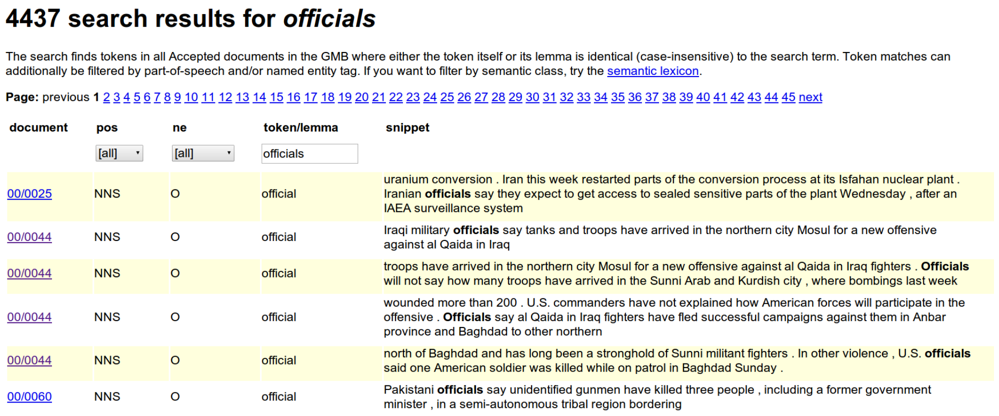
Document **5008 of 11441**, ID: 00 / 0044

size: 6 sentences, 132 tokens
last processed: 06 May 2014, 07:57:23
C&C tools/Boxer revision: 2522

```
Iraqi military officials say tanks and troops have arrived in the northern city
Mosul for a new offensive against al Qaida in Iraq fighters.
Officials will not say how many troops have arrived in the Sunni Arab and
Kurdish city, where bombings last week killed at least 34 people and wounded
more than 200. U.S. commanders have not explained how American forces will
participate in the offensive.
Officials say al Qaida in Iraq fighters have fled successful campaigns against
them in Anbar province and Baghdad to other northern provinces. Mosul is the
largest city north of Baghdad and has long been a stronghold of Sunni militant
fighters.
In other violence, U.S. officials said one American soldier was killed while on
patrol in Baghdad Sunday.
```

Figure 4.6: The GMB Explorer Web interface showing the tokenization of one document and some of the options.

We started the development of a Web-based interface for the GMB at the very beginning of the project, called the GMB Explorer. The first version of the Explorer, online since the beginning of 2011, consisted of a collection of PHP scripts implementing core functionalities such as browsing the documents and searching for words.



4437 search results for *officials*

The search finds tokens in all Accepted documents in the GMB where either the token itself or its lemma is identical (case-insensitive) to the search term. Token matches can additionally be filtered by part-of-speech and/or named entity tag. If you want to filter by semantic class, try the semantic lexicon.

Page: previous **1** 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 next

document	pos	ne	token/lemma	snippet
	[all]	[all]	officials	
00/0025	NNS	O	official	uranium conversion . Iran this week restarted parts of the conversion process at its Isfahan nuclear plant . Iranian **officials** say they expect to get access to sealed sensitive parts of the plant Wednesday , after an IAEA surveillance system
00/0044	NNS	O	official	Iraqi military **officials** say tanks and troops have arrived in the northern city Mosul for a new offensive against al Qaida in Iraq
00/0044	NNS	O	official	troops have arrived in the northern city Mosul for a new offensive against al Qaida in Iraq fighters . **Officials** will not say how many troops have arrived in the Sunni Arab and Kurdish city , where bombings last week
00/0044	NNS	O	official	wounded more than 200 . U.S. commanders have not explained how American forces will participate in the offensive . **Officials** say al Qaida in Iraq fighters have fled successful campaigns against them in Anbar province and Baghdad to other northern
00/0044	NNS	O	official	north of Baghdad and has long been a stronghold of Sunni militant fighters . In other violence , U.S. **officials** said one American soldier was killed while on patrol in Baghdad Sunday .
00/0060	NNS	O	official	Pakistani **officials** say unidentified gunmen have killed three people , including a former government minister , in a semi-autonomous tribal region bordering

Figure 4.7: The result of searching for the word *officials* in the GMB Explorer.

Since its inception, the codebase of the GMB Explorer has grown significantly and it underwent several refactoring, and now it includes many features, detailed in the following list.

- **Document view** - the most basic function of the Web interface is to provide access to the documents and the annotations in the GMB. A document is presented to the user in a multi-tab page, different tabs provide different views on the document. The *metadata* tab shows some basic information on the document, that is, its title, origin, genre and license terms. The actual text, exactly as it is stored on disk, is shown in the *raw* tab, using a monospaced font to preserve the original formatting. The *tokens* table shows again the text of the document but with the tokenization made explicit, using the convention of having one sentence token per line and each word token separated by whitespace. The next tab, labeled *sentences*, shows the annotations at the word level and the syntax tree of each sentence in the document. The visualization of each single layer of annotation can be switched on and off to improve the readability, and the CCG parse tree can be folded and unfolded interactively by clicking on its constituents. Finally, the **discourse** tab shows a graphical representation of the semantic analysis of the text in the form of a DRS. Since a discourse referent in a DRS can appear in multiple places, and in a middle-sized DRS they can be hard to track, when the mouse pointer hover over one of them, all the other occurrences are highlighted.
- **Browsing and filtering** - the default order of the documents in the GMB Ex-

plorer is by size, that is, they are ordered according to the number of their words. When opening the interface for the first time, the shortest document is shown. By clicking on the links in the navigation section of the interface, it is possible to move through the collection one document at the time, or to jump at the beginning or the end of the sequence. An additional link opens a document chosen at random. It is possible to look for documents in a specific subcorpus of the GMB by selecting from a dropdown menu, as well as only view documents with/without BOWs or that produce (or do not produce) warnings.[6]

- **Search function** - The GMB Explorer provides a basic search facility to look up words in the documents. The tokenized files are indexed using Apache Lucene[7], a free text search engine library, then the index is used to retrieve documents matching one or more words. The result of the text search is presented in a separate page in a list containing one matching document per row, with a link to the document page, and the relevant context with the search key highlighted. In the results page it is also possible to refine the search by specifying a part-of-speech, a named entity tag and/or a lemma.

A screenshot of the GMB Explorer, showing some of the features described above, is in Figure 4.6, while Figure 4.7 shows the result of a search in the corpus.

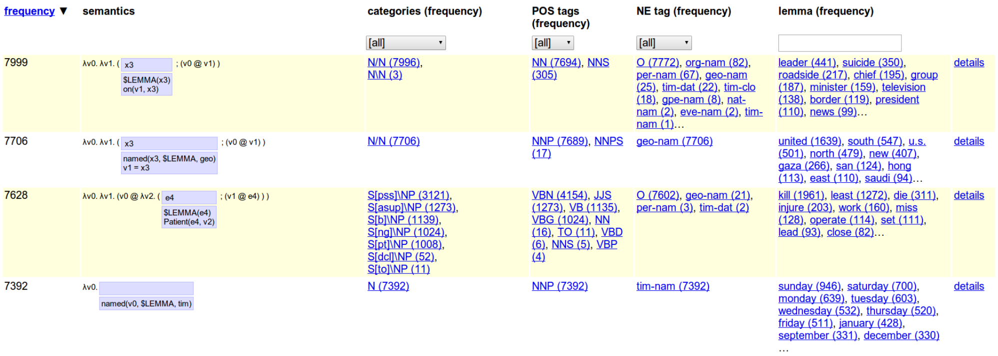
The features described so far, with the exception of the filters on documents, are immediately accessible to the visitor of the GMB Explorer page. For some other actions we need to keep track of who performs them, thus some authentication mechanism is required. We employ a standard username/password authentication, implemented by the PHP library User Cake[8]. A visitor can register a new user account by clicking on the *new user* link and providing a username, a password and an email address. To avoid automatic user registration e.g., by Internet bots, an email is sent to the user with a link to visit in order to activate the account. Once the account is activated and the user has logged into the system, a few more functions are unlocked, most notably the possibility to edit the annotation, a feature described in detail later in this chapter. The features described in the following list require user authentication.

- **Semantic lexicon** - in the GMB every word is assigned a semantics in the form of a lambda calculus expression. A semantic lexicon is extracted from the GMB

[6] When the software pipeline fails to analyze a document, its error message is shown in the GMB Explorer as a *warning*.

[7] http://lucene.apache.org/core/

[8] http://usercake.com/



frequency ▼	semantics	categories (frequency)	POS tags (frequency)	NE tag (frequency)	lemma (frequency)	
		[all] ▾	[all] ▾	[all] ▾		
7999	λv0. λv1. (x3 ; (v0 @ v1)) $LEMMA(x3) on(v1, x3)	N/N (7996), N\N (3)	NN (7694), NNS (305)	O (7772), org-nam (82), per-nam (67), geo-nam (25), tim-dat (22), tim-clo (18), gpe-nam (8), nat-nam (2), eve-nam (2), tim-nam (1)…	leader (441), suicide (350), roadside (217), chief (195), group (187), minister (159), television (138), border (119), president (110), news (99)…	details
7706	λv0. λv1. (x3 ; (v0 @ v1)) named(x3, $LEMMA, geo) v1 = x3	N/N (7706)	NNP (7689), NNPS (17)	geo-nam (7706)	united (1639), south (547), u.s. (501), north (479), new (407), gaza (266), san (124), hong (113), east (110), saudi (94)…	details
7628	λv0. λv1. (v0 @ λv2. (e4 ; (v1 @ e4))) $LEMMA(e4) Patient(e4, v2)	S[pss]\NP (3121), S[asup]\NP (1273), S[b]\NP (1139), S[ng]\NP (1024), S[pt]\NP (1008), S[dcl]\NP (52), S[to]\NP (11)	VBN (4154), JJS (1273), VB (1135), VBG (1024), NN (16), TO (11), VBD (6), NNS (5), VBP (4)	O (7602), geo-nam (21), per-nam (3), tim-dat (2)	kill (1961), least (1272), die (311), injure (203), work (160), miss (128), operate (114), set (111), lead (93), close (82)…	details
7392	λv0. named(v0, $LEMMA, tim)	N (7392)	NNP (7392)	tim-nam (7392)	sunday (946), saturday (700), monday (639), tuesday (603), wednesday (532), thursday (520), friday (511), january (428), september (331), december (330) …	details

Figure 4.8: The interface to the semantic lexicon as shown in the GMB Explorer.

data as a catalog of all possible semantics occurring in the analysis of the GMB text. In the semantic lexicon page in the GMB Explorer, it is possible to browse all the items or restrict the shown list by searching for items with a specific syntactic category, part-of-speech or named entity label. At the time of this writing, the semantic lexicon includes 1,071 entries. An example screen of the Web interface to browse the semantic lexicon is in Figure 4.8.

- **Statistics and logging** - the *statistics* link sends to a page where the numbers of the GMB are displayed. Here the information is shown about the size of the collection (number of documents, words and sentences broken down by subcorpus) averages (words per sentence, sentences per document), and statistics on the number of *effective* BOWs, that is, the BOWs that are actually picked up by the toolchain. The numbers in the statistics page are all automatically generated on the fly, thus they are always up to date. A log of all the activities of the GMB Explorer users and the toolchain is found in the *news feed* page. The *warnings* page gives an overview of the errors returned by the software tools and provides links to the documents that generate such errors.

- **Reprocess document** - clicking the button labeled *reprocess document* will run remotely the software pipeline on the selected document. Even though the documents are periodically reprocessed in an automatic way, manually clicking the button will speed up the process by prioritizing the document. This feature is useful when new annotations are added to the document, or when a new version of the software is available.

- **Report issue** - this link opens a page containing a Web form in which the user can enter information such as bug reports or requests for new features. This

Table 4.3: Examples of Bits of Wisdoms and their meaning.

BOW type	BOW content	meaning
tag	69,0239,0,5,pos,JJ	token *Iraqi* at 0-5 of doc. 69/0239 has pos tag JJ
tag	02,0293,431,436,sense,videotape.v.01	*tape* at 431-436 of doc. 02/0293 has sense id videotape.v.01
tag	00,0044,661,665,NE,GPE	token *U.S.* at 661-665 of doc. 00/0044 has ne tag GPE
tag	00,0044,397,406,animacy,human	token *Officials* at 397-406 of doc. 00/0044 has animacy Human
tokenization	33,0234,451,T	character at 451 of doc. 33/0234 (,) labeled Start of Token

information, together with a link to the document from which the report is submitted, is sent to a bug tracking system and assigned to one of the maintainers of the GMB.

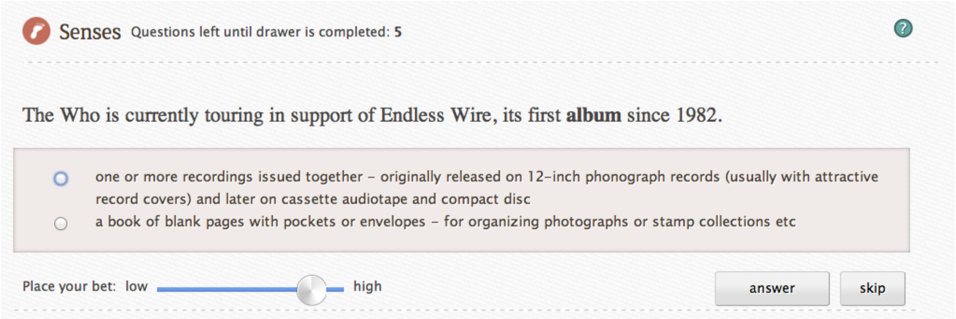
The edit operations in the GMB Explorer produce BOWs that are stored in a central database. The BOWs can have different types, yet they share a common structure. In the database, the BOW entries contain a unique identifier, the identifier of the document, a string encoding its type (tokenization BOW or tag BOW), and the value. The value of a BOW follows a different format depending on its type. Tokenization BOWs assert that a specific character, identified by its offset, has one of the four character labels used in our tokenization scheme (S, T, I or O). Tag BOWs contain two character offset to identify a word token, the type of tag (POS, NE, sense, ...) and the tag itself. In the case of word senses, the tag is the WordNet synset representing the meaning of the given word. An assorted sample of BOWs is given as example in Table 4.3

4.4.3 A Game With a Purpose for Linguistic Annotation

Supervised statistical models rely on gold standard data from human annotators, but this data is very often time-consuming and expensive to obtain. In the context of constructing the Groningen Meaning Bank, we addressed this problem by outsourcing the manual labor to expert linguists over the Internet, as seen in the previous Section. The limited number of active participants is not due to technical issues, on the contrary the GMB Explorer could in theory be used by thousands of users without changing its structure. The reason why the Explorer is underused with respect to its capacity is that we are asking difficult questions, and experts are rare. To overcome this limitation and leverage the potential of global connettivity we tried the alternative route of *crowdsourcing*.

The idea of crowdsourcing is that some tasks that are difficult to solve for computers but easy for humans may be outsourced to a large number of people across the globe. One of the first and most well-known crowdsourcing platforms is Amazon's Mechanical Turk[9], an online labor marketplace where workers get paid small amounts to complete small tasks. Mechanical Turk has already been successfully applied for the purpose of word sense disambiguation and clustering Akkaya et al. [2010], Rumshisky et al. [2012].

Another crowdsourcing technique that is rapidly gaining momentum is the "Game with a Purpose". A GWAP rewards contributors with entertainment rather than money. GWAPs challenge players to score high on specifically designed tasks, thereby contributing their knowledge. GWAPs were successfully pioneered in NLP by initiatives such as 'Phrase Detectives' for anaphora resolution Chamberlain et al. [2008] and 'JeuxDeMots' for term relations Artignan et al. [2009]. We have developed an online GWAP platform for semantic annotation, called *Wordrobe*.



The Who is currently touring in support of Endless Wire, its first **album** since 1982.

○ one or more recordings issued together – originally released on 12-inch phonograph records (usually with attractive record covers) and later on cassette audiotape and compact disc

○ a book of blank pages with pockets or envelopes – for organizing photographs or stamp collections etc

Figure 4.9: An example question from one of the games of Wordrobe, as shown in the game interface.

Wordrobe[10] is actually a collection of Games With A Purpose, each targeting a specific level of linguistic annotation. The collection is growing as we design and add new games to the platform. Current challenges include part-of-speech tagging, named entity tagging, co-reference resolution, animacy tagging, relation identification and word sense disambiguation. Wordrobe is designed to be used by non-experts, who can use their intuitions about language to annotate linguistic phenomena, without being discouraged by technical linguistic terminology, therefore the games include as little instructions as possible. All games share the same structure: a multiple-choice question with a small piece of text (generally one or two sentences) in which one or more words are highlighted, depending on the type of game. For each question, players can select an answer or use the button marked *skip* to go to

[9] http://aws.amazon.com/mturk/

[10] http://www.wordrobe.org/

the next question. The way in which questions are actually presented to the players in the game is depicted in Figure 4.9.

In order to encourage players to answer a large number of questions and to give good answers there must be some game element involved. In Wordrobe a player finds two kind of challenges, not mutually exclusive: collecting badges and playing against other players. The more questions a player answer, the more she unlocks icons that decorate her personal profile page. While the badge system encourage players to keep on answering question, scores and leaderboards motivate players to play with attention. The points are calculated on the basis of two factors: the *agreement* with other players who answered the same question and a *bet* that the player optionally put at stake. Players can place a bet reflecting the certainty about their answer. The default choice is a minimum bet and once a player adjusts the bet, this new value is remembered as the new preset value for the next question. Higher bets will result in higher gains when the answer is correct, and lower points when the answer is wrong. Since Wordrobe is designed to create gold standard annotations, the correct choice is unknown (this is exactly what we want to obtain!), therefore the points are calculated on the basis of the answers given by other players, as in Phrase Detectives Chamberlain et al. [2008]. The idea is that the majority rules, the choice that gets selected most by human players is likely to be the correct one. The more players agree with each other, the more points they gain. As a consequence, the score of a player is continually updated — even when the player is not playing — in order to take into account the answers provided by other players answering the same questions.

The questions of Wordrobe are generated automatically from the GMB data. Depending on the game, different set of rules and restrictions are applied to the texts of the GMB in order to extract elementary questions about single phenomena on specific unit of text. For instance, in the game *Senses*, a game of sense disambiguation, the question generation procedure considers all the word tokens that are tagged as nouns or verbs, looks them up in WordNet and selects those having four or five senses. The inventory of possible choices for the questions of Senses is given by the definitions of the senses of the word to disambiguate. Internally, the possible answers are represented by BOWs, so that the system keeps tracks of the original GMB document from where the question was extracted, as well as the answer itself.

Here is an example of question for the game Senses as stored in the internal database together with its possible answers.

Question:

- **id**: 950
- **game id**: 1
- **text**: Delp left two notes **taped** to a door, along with letters to his family and his fiancee, Pamela Sullivan.

Choices:

question id	text	BOW
950	fasten or attach with tape	02,0293,431,436,sense,tape.v.01
950	record on videotape	02,0293,431,436,sense,videotape.v.01
950	register electronically	02,0293,431,436,sense,record.v.02

In total, we developed eight games. At the moment six games are online in Wordrobe, while two of the games have been retired. In *Names* the player is asked to identify the Named Entity class of a given sequence of proper nouns. *Pointers* is a game about anaphora, where one must find the correct antecedent of a personal pronoun. *Burgers* presents noun-noun compounds in context and asks to identify the underlying relation between the two nouns, e.g., an *emergency plan* is a plan *for* emergency. *Animals* is a game similar to Names, but concerning Animacy classification. Finally, in *Others* the player is asked to find what entity the word *other* is referring to in a given context. The two games that are not online anymore are *Twins*, a simple game of POS-tag disambiguation between nouns and verbs, and *Viittaukset*, an experimental version of Pointers on Finnish text.

4.4.4 Experts vs Crowd: Evaluation

With the system online and working, collecting answers remains only a matter of time and advertisement, but the annotation problem is not solved yet. There is obviously no one-to-one correspondence between the players' answers and the BOWs that we want to obtain at the end, thus there is the need for a method of aggregating the answers in a meaningful way and extract the linguistic knowledge for the GMB. The intuition is to resort to some measure of agreement between players, similarly to the approach followed for granting points during the game. We designed an experiment to investigate agreement-based methods of extracting BOWs from Wordrobe answers, and their differences in terms of quality of the annotation. The experiment is based on the aforementioned game Senses.

The number of automatically generated questions for the first version of Senses was 3,121. After the first few weeks of Wordrobe going live, we had received 5,478 answers. Roughly half (1,673) of the questions received at least one answer, with an average of three answers per question. We created a gold standard annotation for a set of 115 questions with exactly six answers each (a subset of the questions with a reasonable response rate) which was used to evaluate the answers given by the players of Wordrobe. Four trained human annotators individually selected the correct sense for each of the target words in the test set. Fleiss's kappa was calculated to evaluate inter-annotator agreement, resulting in $\kappa = 0.79$, which is generally taken to reflect substantial agreement. Unanimity was obtained for 64% of the questions and 86% of the questions had an absolute majority vote. In a second step of evaluation, the non-unanimous answers were discussed between the annotators in order to obtain 100% agreement on all questions, the result of which was used as the gold standard annotation.

Given a question and a set of player answers, we tested procedures to decide whether to accept a particular choice into our annotated corpus. One important factor, as said earlier, is agreement: if a great majority of players agrees on the same choice, this choice is probably the correct one. Smaller majorities of players are more likely to be wrong. Another important factor is the number of answers: the more players have answered a question, the more we can presumably rely on the majority's judgment. In this experiment, we focused on the first factor (agreement) because the average answer rate per question is quite low throughout our data set. We tested a couple of simple agreement measures that determine whether a choice is counted as a winning answer and measured recall and precision for each measure with respect to the gold standard.

The simplest measure accepts every choice that has a relative majority. It always accepts some choice, unless the two choices with the most answers are tied. A stricter measure ("absolute majority") accepts only the choices that were chosen by at least a certain fraction of players who answered the question, with some threshold $t \geqslant 0.5$. We used the values 0.5, 0.7 and 1.0 as threshold, the latter only accepting choices unanimously picked by players.

The measures described above simply choose the majority answer relative to some threshold, but fail to take into account the total number of players that answered the question and the number of possible choices for a question. These factors will become more important when we evaluate questions with a higher number of answers. We need a measure that determines whether the majority answer is chosen significantly more often than the other answers. This means that the answers should be significantly skewed towards one answer. In order to test such an effect, we used Pearson's chi-square test, which determines the goodness-of-fit of a given distribution relative to a uniform distribution. If we take the distribution of answers over the set of possible choices, we can say that only those questions for which this distribution significantly differs from a uniform distribution ($p < 0.05$) are considered to provide an acceptable answer. Because the number of answers per question in our gold standard set is relatively small, a significant result means that there is one choice towards which the answers accumulate. Determining which choice this is can accordingly be done using the relative-majority measure described above.

Table 4.4: Precision and recall based on different agreement measures

Strategy	Precision	Recall	F-score
Relative majority	0.880	$\mathbf{0.834}$	$\mathbf{0.857}$
Absolute majority ($t = 0.5$)	0.882	0.782	0.829
Absolute majority ($t = 0.7$)	0.945	0.608	0.740
Unanimity ($t = 1$)	$\mathbf{0.975}$	0.347	0.512
Chi-square test ($p < 0.05$)	0.923	0.521	0.666

We evaluated the annotations obtained from Wordrobe by comparing the data of the test set (115 questions) to the gold standard. We used each of the agreement measures described above to select the answers with a high enough majority, and calculated precision (the number of correct answers with respect to the total number of selected answers), recall (the number of correct answers with respect to the total number of questions), and the corresponding F-score. The results are shown in Table 4.4.

As expected, the highest recall is obtained using the relative majority measure since this measure is the least conservative in accepting a majority choice. As the threshold for accepting a choice is set higher, recall drops and precision rises, up to a very high precision for the unanimity measure, but with a significant loss in recall. The measure based on Pearson's chi-square test is similar in being conservative; having only six answers per question in the test set, only the questions that are very skewed towards one choice give a significant result of the chi-square test.

As described above, each answer is associated with a bet between 10% and 100% of the points available for a question, which players can adjust based on how certain they are about their answer. The distribution of bets over all answers shows two significant peaks for these extremes: in 66% of the cases the maximum bet was chosen, and the default minimum bet was chosen in 12% of the cases. The main motivation for inserting the betting function was to be able to identify questions that were more difficult for players by looking for low bets. We tested the correlation between the average bet per question and the relative size of the majority (indicating agreement between players) over all questions using Pearson's product-moment correlation and found a small but significant positive effect ($r = 0.150$, $p < 0.01$). We expect that this effect will increase if more data is available.

Table 4.5: Precision and recall based on different agreement measures for questions with $\bar{b} \geqslant 80\%$

Strategy	Precision	Recall	F-score
Relative majority	0.917	**0.478**	**0.629**
Absolute majority ($t = 0.5$)	0.930	0.461	0.616
Absolute majority ($t = 0.7$)	0.956	0.383	0.547
Unanimity ($t = 1$)	**0.961**	0.217	0.355
Chi-square test ($p < 0.05$)	0.950	0.330	0.355

In order to test whether questions with high average bets were easier, we repeated the evaluation, including only questions with a high average bet: $\bar{b} \geqslant 80\%$ (see Table 4.5). Recall is reduced strongly, as one would expect, but we do observe an increase in precision for all measures except unanimity. This higher precision suggests that indeed the results of the questions for which players on average place a high bet are more similar to the gold standard. However, we will need more data to confirm this point.

From its launch in September 2012 to the end of 2013 Wordrobe collected over 60,000 single answers. Of course the actual number of BOWs we can extract is much smaller. One reason is that there needs to be a minimum number of answers to a specific question in order to consider the information reliable. Besides, some measure of inter-player agreement needs to be computed and used to select the answers on which players mostly agree. A pilot study that explores the data extracted from one of the Wordrobe games (*Senses*, a game of sense disambiguation) showed that a few parameters have to be carefully tuned in order to obtain high quality annotation from the game. Venhuizen et al. [2013a]

The first BOW coming from the GMB Explorer is dated 18/10/2011. Since then until the time of this writing (a period of about 2.5 years) we collected through the Web interface 35,867 BOWs. 18 experts contributed to the resource, with the five most active users making the 97% of the total number of BOWs.

To the manually produced BOWs we must add the BOWs automatically produced by scripts or taken from external resources. To date, the GMB comprises 65,211 BOWs automatically created by running several scripts implementing simple heuristics, plus 463,843 BOWs converted from the MASC data. The latter set of BOWs, however, spans the whole collection of documents regardless of the document status (accepted, uncategorized, ...) while the other BOWs are relative only to accepted documents with very few exceptions. The actual number of BOWs from MASC, mainly POS-tags, relative to accepted documents in the GMB is 7,052.

4.5 Conclusion

In this chapter, I reported about the process of creation of the GMB. This includes the actual collection of documents, many technical and linguistic choices, the development of a software infrastructure for automatic linguistic annotation, the creation of a series of games to gather data from the Internet crowd, and some evaluation experiments.

Although this was not the primary intent of the project, the GMB is the resource that allows the Unboxer to be a *supervised* approach to NLG, one that learns from text and annotation and can be made better with the addition of new data.

A supervised model is in principle more robust, whereas designers of rule-based NLG systems have to hand-craft rules, templates and the like whenever the need arises for the application to a different domain than the original. Moreover, a system that learns from real world data is bound to produce more natural sounding utterances.

The next chapter will make clear how the Unboxer approach is implemented and how the GMB data is manipulated in order to create a suitable data set for learning the surface forms associated with logic formulas.

Chapter 5

Learning Surface Order

In the previous chapter I introduced the Groningen Meaning Bank, a large linguistic resource with many applications in statistical NLP. In fact, the most important reason why a resource like the GMB needs to be of a considerable size (measured in millions of words) lies in the need for huge data sets to serve as training material for supervised statistical models. Among such applications we find Natural Language Generation and its many sub-tasks.

In Chapter 3 I proposed a new approach to build a complete NLG system that takes Discourse Representation Graphs as input and produces a surface form that expresses its meaning. The system, called Unboxer, is made of several interconnected parts, including a statistical component whose purpose is to predict the word order-related information of the input semantic structure, ultimately predicting the final surface order in an indirect way. Such a component works in a supervised fashion, using the data in the GMB as its training base, and it is the topic of this chapter.

Word order is fundamental information to give meaning to what otherwise would be a "word salad". In NLG, one can think of at least two approaches, opposite to one another. The first is to rely on information acquired from existing text on the probabilities of the possible orders of a given list of words, or, in other words, constructing a statistical model of the language to assign the most probable order to a set of words. In literature there are many examples of there *language models* applied to linearization, that is, the problem of giving a linear structure to a set of lexical items.

A different approach, adopted in this thesis, consists of computing the order information inside the meaning representation, that is, *before* or in *parallel* to the generation of the word themselves. This approach has the advantages of (a) being more robust in those cases where the appropriate word for a concept cannot be found, or it is so uncommon that a language model do not contains it, and (b) not depending on a lexical choice step, thus in principle being more independent of the language.

In the rest of this chapter, this novel approach to the word order problem is tested in the framework of generation from logical forms, specifically from DRSs. The motivation for the experiments presented in this chapter is twofold. First, I want to prove that it is indeed possible, to a certain extent, to predict the order of words and constituents based only on the information contained in a structure representing its meaning. Second, the aim of the multiple runs of the experiments is to find out which features of the meaning representation are more important to the prediction of the surface order.

5.1 Method

In Chapter 3, I motivated the choice of *local ordering* as a strategy to encode word order in a text-to-logic alignment. To recap, each tuple relative to a discourse referent in a DRG is labeled with an index, that is, an integer positive number that specifies its position among all other tuples relative to the same referent.

k_1	unary	$\neg$		
$\neg$	scope	k_2		
k_2	referent	e_1		
k_2	referent	x_1	**1**	A
k_2	event	pay		
k_2	concept	$customer$		
k_2	role	$agent$		
$customer$	instance	x_1	**2**	customer
pay	instance	e_1	**4**	pay
$agent$	internal	e_1	**1**	
$agent$	external	x_1		
k_2	surface	e_1	**2**	did
k_2	surface	e_1	**3**	not
k_2	surface	e_1	**5**	.

Figure 5.1: Word-aligned DRG for "A customer did not pay." The alignment information relative to the order is highlighted.

The structure of DRGs ensures that this way of encoding word order makes possible to reconstruct the global word order of a text. Gold standard DRGs have their

k_2	referent	x_1	**1**	A
$customer$	instance	x_1	**2**	customer

$agent$	internal	e_1	**1**	
k_2	surface	e_1	**2**	did
k_2	surface	e_1	**3**	not
pay	instance	e_1	**4**	pay
k_2	surface	e_1	**5**	.

Figure 5.2: Tuples with local ordering relative to x_1 and e_1 respectively.

tuples labeled with such indexes, thus providing training material for a supervised algorithm.

A DRG that has been aligned with one of its potential corresponding surface form looks like the one depicted in Figure 5.1, where we can see the local ordering in action. The last two column represent the alignment with the surface form "A customer did not pay". In fact, in this example there are two separate groups of ordered tuples, relative to the discourse referents x_1 (the customer) and e_1 (the paying event). In Figure 5.2 the two groups of tuples are shown separately. Visualizing the DRG as a directed graph (see Chapter 3), these lists of tuples would be represented by edges having the same head node, i.e., the discourse referent in question.

As mentioned in the beginning of this chapter, in the Unboxer architecture the information regarding the order of the words in the output is computed independently from the actual words. To do so, the system tries to predict the local ordering of groups of tuples, that is, it tries to rank sets of tuples, one referent at the time. This type of machine learning problem is referred to as *Learning to Rank*, a family of methods that aims at predicting the correct order of given lists, therefore useful in settings like the ranking of results of search engines. There are at list three types of Learning to Rank approaches: point-wise, pairwise and list-wise. The point-wise approach tries to predict the position in the ranking of each element independently, which is useful when a large set has to be ranked, or when the size of the set is unknown. The pairwise approach approximates the ranking problem to a classification one, predicting the relative position of pairs of elements. The list-wise approach is a generalization of the two methods averaged over all elements or pairs, that is to say, a list-wise Learning to Rank algorithm tries to order a whole list of elements at once. List-wise algorithms are computationally more expensive and needs a training set made of complete ranked lists. In the DRG edge ranking problem, however, both these drawbacks are not present, that is, the data set is made of relatively short lists and the complete rankings are in the gold standard lists provided by the DRGs in the GMB.

In order to experiment with Learning to Rank applied to the problem of learning word order, I implemented a customized version of a list-wise approach algorithm called ListNet [Cao et al., 2007]. I then applied ListNet to a data set extracted from the GMB to try and predict the local rankings of tuples connected to discourse referents.

In the Unboxer approach to learning word order, prominent importance is given to the selection of a suitable set of features, in particular when information found far apart in the structure influences the local order of tuples. For instance, consider an event *enter#v#1* where an entity *passenger#n#1* is the agent and an entity *car#n#2* is the patient. Good predictors for the correct order ("the passenger enters the car" rather than "the car enters the passenger") could be that *car#n#2* is a vehicle in the WordNet hierarchy while *passenger#n#1* is a person. This information is not attached to any of the tuples relative to the event referent, so the DRG structure has to be traversed in order to retrieve it.

The lists of tuples that have to be ranked also contain zero-labeled tuples that have no index. A typical example would be a concept with no determiner, whose *referent*-type tuple contains no surface tokens, thus its position in the local ordering is irrelevant. An additional task therefore needs to be performed, that is, a binary classification, zero vs. positive integer index. This problem is more straightforward than the ordering problem, given a suitable set of features, and can be approached via a standard logistic regression.

Alternatively, the prediction of the order of the DRG tuples can be approached by a single method that combines the two subtasks seen before. For instance, one can assign negative index to the tuples that do not contain an index and then eliminating them in a successive step, thus only applying a Learning to Rank algorithm to solve the entire problem. Experiments on this approach are left as future work.

In the following sections, I describe the algorithms implemented in the ordering component of the Unboxer system (5.1.1). Being supervised learning methods, they need suitable datasets to be trained on. While the raw data is provided by the collection of lexical-aligned DRGs in the GMB, they need to be transformed to fit the input format of the algorithms, by extracting *features* from the tuples to be ordered. The features used by the learning algorithms are described in Section 5.1.2.

5.1.1 Algorithms

ListNet is a list-wise Learning to Rank algorithm based on a simple linear neural network. ListNet is based on, and is a generalization of, RankNet [Burges et al., 2005], a pairwise approach to Learning to Rank.

As a list-wise algorithm, ListNet is reported to perform better than pairwise algorithms on experimental tasks such as document retrieval. On the other hand, following the list-wise approach one has to define an appropriate loss function that can turn up to be hard to compute. ListNet uses a list-wise loss function based on the idea of *top one probability*, that is, the probability of an element of being the first in the ranking. The top one probability model approximates the *permutation probability* model that assigns a probability to each possible permutation of an ordered list, a necessary approximation, given that the number of permutation of a list of n elements is $n!$, therefore making the problem intractable.

Formally, the top one probability of an object j is defined as

$$P_s(j) = \sum_{\pi(1)=j,\pi\in\Omega_n} P_s(\pi)$$

that is, the sum of the probabilities of all the possible permutations of n objects (denoted as Ω_n) where j is the first element. Here $s = (s_1, ..., s_n)$ is a given list of *scores*, that is, the position of elements in the list. Considering two permutations of the same list y and z (for instance, the predicted order and the gold standard order) their "distance" is computed using a metric such as *cross entropy*. This distance, together with the top one probabilities of the list elements are used as loss function:

$$L(y, z) = -\sum_{j=1}^{n} P_y(j)log(P_z(j))$$

The list-wise loss function is plugged into a linear neural network model to provide a learning environment. The learning procedure of ListNet works by taking as input, sequentially, ordered lists of feature vectors. The features are encoded as numeric vectors. The weights of the underlying neural network are iteratively adjusted by computing a list-wise cost function that measure the distance between the gold standard ranking and the prediction of the model and passing its value to the Gradient Descent algorithm for optimization of the parameters.

On top of the algorithm of Cao et al. [2007], presented so far in this section, I extended ListNet used with the additional feature of a regularization parameter

λ. With $\lambda > 0$ the list-wise cost is increased by a small factor in order to prevent overfitting.

As mentioned in the previous section, the problem of predicting the local order of tuples is not fully solved by the application of a Learning to Rank algorithm, since one is still left with the task of deciding whether a tuple has an index at all. Unlike the ordering problem, this task takes single tuples in input and classify them into either 0-indexed or positive-indexed, that is, the task is an instance of binary classification and it is easily solved with a regression classifier. For this thesis, I used an existing implementation of a SVM model [Cortes and Vapnik, 1995] called *svmlight* [Joachims, 1999].

Both algorithms works on numeric vectors, where different features of the elements to classify (or to order) are represented as numbers. The next section explains in detail these features and how they are represented for the purpose of machine learning.

x_1 x_3 s_4 s_5 e_6 t_8 t_9

named(x_1, john, org)
Topic(s_4, x_3)
big(s_4)
Topic(s_5, x_3)
blue(s_5)
bike(x_3)
ride(e_6)
Agent(e_6, x_1)
Theme(e_6, x_3)
now(t_8)
$e_6 \subseteq t_9$
$t_9 = t_8$

Figure 5.3: DRS for "John rides a big blue bike."

5.1.2 Features

As a supervised approach, the present method to learn the order of the tuples of a DRG needs training data. Specifically, many examples of ordered lists of tuples are

k_{10}	attribute	$c_2 : big : 0$	0	
k_{10}	attribute	$c_4 : blue : 0$	0	
k_{10}	attribute	$c_9 : now : 1$	0	
k_{10}	concept	$c_5 : bike : 0$	0	
k_{10}	event	$c_6 : ride : 0$	0	
k_{10}	named	$c_0 : john : org$	0	
k_{10}	referent	$k_{10} : e_6$	0	
k_{10}	referent	$k_{10} : s_4$	0	
k_{10}	referent	$k_{10} : s_5$	0	
k_{10}	referent	$k_{10} : t_8$	0	
k_{10}	referent	$k_{10} : t_9$	0	
k_{10}	referent	$k_{10} : x_1$	0	
k_{10}	relation	$c_{10} : temp\_included : 1$	0	
k_{10}	relation	$c_{11} : equality$	0	
k_{10}	role	$c_1 : Topic : -1$	0	
k_{10}	role	$c_3 : Topic : -1$	0	
k_{10}	role	$c_7 : Agent : 1$	0	
k_{10}	role	$c_8 : Theme : 1$	0	
$c_1 : Topic : -1$	ext	$k_{10} : s_4$	0	
$c_{10} : temp\_included : 1$	ext	$k_{10} : t$9	0	
$c_{10} : temp\_included : 1$	int	$k_{10} : e$6	0	
$c_{11} : equality$	ext	$k_{10} : t_8$	0	
$c_{11} : equality$	int	$k_{10} : t_9$	0	
$c_3 : Topic : -1$	ext	$k_{10} : s_5$	0	
$c_7 : Agent : 1$	ext	$k_{10} : x_1$	0	
$c_8 : Theme : 1$	ext	$k_{10} : x_3$	0	
$c_9 : now : 1$	arg	$k_{10} : t_8$	0	
$c_0 : john : org$	instance	$k_{10} : x_1$	1	John
$c_7 : Agent : 1$	int	$k_{10} : e_6$	1	
k_{10}:e6	main	k_{10}	1	
$c_6 : ride : 0$	instance	$k_{10} : e_6$	2	rides
k_{10}	referent	$k_{10} : x_3$	1	a
$c_2 : big : 0$	arg	$k_{10} : s_4$	1	big
$c_1 : Topic : -1$	int	$k_{10} : x_3$	2	
$c_4 : blue : 0$	arg	$k_{10} : s_5$	1	blue
$c_3 : Topic : -1$	int	$k_{10} : x_3$	3	
$c_5 : bike : 0$	instance	$k_{10} : x_3$	4	bike
$c_8 : Theme : 1$	int	$k_{10} : e_6$	3	
k_{10}	punctuation	$k_{10} : x_3$	5	.

Figure 5.4: Text-aligned DRG for the sentence "John rides a big blue bike.".

needed, and each tuple must provide as much information as possible in order for the algorithm to create a valid representation.

The training set used here is extracted from the Groningen Meaning Bank — see Chapter 4. Recall that the GMB makes available a collection of thousands of texts word-aligned with a formal representation of their meaning, in the form of a DRG. An example of a gold standard DRG as found in the GMB is shown in Figure 5.4 and will serve as a working example in this section. The example DRG is equivalent to the DRS in Figure 5.3.

In the GMB, the symbols for concepts, events and attributes of a DRG are also linked to a WordNet synset to specify their meaning. For readability purposes, the synsets are hidden in Figures 5.3 and 5.4.

Each DRG in the training set contains several discourse referents. For each discourse referent x, the list of local tuples is extracted, that is, the tuples having x as third element. The discourse referents belong to one of three kinds: entity, event or discourse unit. Features are extracted per-tuple and transformed into a numeric vector, each having a fixed number of elements. Here follows a description of the features, grouped into four classes.

Table 5.1: Features extracted from the tuple <$c_6 : ride : 0$, instance, $k_{10} : e_6$>

Feature	**Value**
Discourse Unit type	main clause
Punctuation	false
Surface	false
Referent type	event
Binary relations	Agent, Patient
Concept hypernym	none
Verb category	travel
Cardinality	none
Temporal relations	included

Structural Features

- *Discourse unit type*: this feature specifies if the discourse unit in which the discourse referent appear is a main, subordinate or embedded clause (e.g., relative clause). The type of subordination (e.g., negation, conditional, etc.) is also included in the feature value.

- *Punctuation, Surface*: this binary features indicates the presence of a "punctuation" or "surface" kind of edge.
- *Referent type*: the type of the node on the other side of the edge. This can be one of the following: attribute, concept, discourse unit, event, named entity, referent, binary relation, thematic role.
- *Binary relations*: if the edge represents one argument of a binary relation, this feature encodes what type of relation, e.g., thematic role, temporal relation, equality, etc.

Semantic Features

- *Entity hypernym*: two-level hypernym of the synset associated with the entity, from the WordNet. If the entity is already in a high position in the hypernym hierarchy, then the direct hypernym is taken, or the entity synset itself.
- *Event category*: category of the synset associated with the event, as defined in WordNet. One reason to look at noun hypernyms and verb categories, instead of simply taking the synsets, is to avoid very sparse feature vectors.

Cardinality Features

- *Cardinality of entities*: information about cardinality of entities, when existing. For instance, in the example DRG, the referent x_3 (the bike) has no cardinality, thus by default it is considered singular.

Tense Features

- *Temporal information of events*: following Kamp and Reyle, event referents are linked to a "reference point" representing the present moment in time, through temporal relations such as *before*, *included*, *overlap*, or *abut*. In the example DRG, the temporal relations in which e_6 is involved encode its tense (present, in this case).

For practical purposes, the features are transformed into binary vectors. The final representation of one tuple is then the concatenation of its feature vectors. This representation also presents the advantage of allowing combinations of values for each feature. For instance, if a discourse referent is member of two distinct relations, the "binary relations" section of its feature vector representation will have two 1s and 0s in the other element places.

As an example of feature extraction, consider the tuple <$c_6 : ride : 0$, instance, $k_{10} : e_6$> in the DRG in Figure 5.4 (the last two elements <2, rides> are the gold standard alignment and play no role in the feature extraction). Table 5.1 shows the features provided by the DRG for this particular tuple.

5.2 Experimental Setup

The method described in the previous section needs to be systematically tested in order to be considered valid, and therefore to consider its inclusion in the general architecture of the Unboxer system. For this reason, I devised a series of experiments to test the approach to prediction of word order in general, and in particular the effect of different kind of features on the results.

The experiments described in this section follow a typical pattern in Computational Linguistics. Starting from a gold standard corpus, a large set of instances of the problem paired with their solution, a statistical model is trained to learn the pairing by making an inner representation of the problem itself. If the training phase is successful, the model should be able to generalize, that is, to be able to find a correct solution for instances of the problem not encountered beforehand. In the present case, the tasks are the binary realization classification (whether or not a tuple is to assigned an index) and list-wise ordering of groups of tuples. The training instances are single tuples for the first task, ordered lists of tuples for the second task. At the end of the training phase, models will be available that are capable of predict the indexes of new tuples. To test if and to what extent the algorithm is successful, the models are used to make predictions, that is, guessing the order of a different set of instances that were not used for training.

The availability of a resource such as the GMB makes possible to design a series of experiments on a number of examples large enough for the results to be significant. Moreover, these experiments can be carried out in a fully automatic way, thus eliminating possible selection biases and enabling the researcher to easily reproduce the results. Still, the data sets have to be extracted and processed, and the necessary software infrastructure has to be developed.

In this section, I describe the data I used to test the algorithm for ranking edges of a DRG, how I extracted and translated them in the right format, and the software technology underlying the experiments.

5.2.1 Data Set

In order to test the effectiveness of method presented in the previous section, a training data set is needed. I derived one by extracting 30.000 gold standard DRGs from the whole GMB corpus, specifically its development version, each corresponding to a single sentence. Lists of tuples are automatically extracted for two discourse referent types: entity and event. 442,813 tuples are relative to entities (157,345 referents, on average 2.8 tuples per list) and 317,657 to events (85,832 referents, on average 3.7 tuples per list). As test set for cross-validation, I extracted the same kind of tuples from a separate set of 5,000 sentences.

5.2.2 Software Implementation

When running multiple tests while experimenting with a number of parameters, features, algorithms, and data sets, it is often convenient to structure the body of running code in an efficient way. Starting by directly writing code can yield quick results for a pilot study, but it is a practice to avoid, that often leads to duplicate code, poor documentation, and hardly reusable components. For these reasons, I designed an object-oriented version of the DRG format, complete with a parser that reads directly the output of Boxer. The DRG library is powered by the programming language Python [1], a choice motivated by several factors, mainly its interoperability, the availability of a large number of working modules, and its shallow learning curve that makes it a great tool for rapid prototyping of software. The software developed to support the experiments presented in this chapter consists of roughly three thousands lines of Python code.

In Section 5.1.1, I mention the implementation of a customized version of the ListNet list-wise Learning to rank algorithm. The ListNet algorithm has been implemented from scratch for the purpose of the experiments presented on in this chapter. Nevertheless, the implementation is functional and general-purpose, not constrained to the specific domain of the topic of this thesis. The software is imple-

[1] http://www.python.org/

mented in the GNU Octave language[2] and is available online as Free Software[3]. GNU Octave is a high level programming language initially designed for linear algebra. In Octave, matrices are first-class citizens and matrix multiplication is a built-in operator, thus the number of lines of code to implement neural network-like functions is drastically reduced. Moreover, Octave makes it easy to vectorize operations like matrix and vector multiplications or the application of custom functions to vectors of values. Thanks to this feature, paired with built-in support for concurrent processing, the passage to a parallel version of the code has been trivial and it yielded a great speed improvement.

Finally, all tasks are kept together and automated by means of scripts for the GNU/Linux shell and the GNU Make tool [4]. Make, in particular, is very effective for the parallelization of independent tasks on a multi-processor computer, and it helps to avoid the replication of intermediate steps.

5.3 Evaluation

In this section I present the results of two series of experiments. The evaluation of the effectiveness of these models for prediction is problematic for a number of reasons. First, the evaluation metrics are not always directly interpretable in a natural way. Second, and somehow linked to the first reason, since the Unboxer comprises a novel approach to NLG, there is no existing testbed to evaluate against — no previous results on this specific task, or a comparable one, is available. That being stated, it is still possible to devise ways of assessing the performance of the supervised models for learning the order of tuples. The first evaluation method is a cross-validation experiment carried out on the two subtasks of binary realization classification (whether or not a tuple has a positive index) and prediction of order. Although it is difficult to give a meaning to the numeric results of this kind of experiments, they are useful in assessing the impact of different combinations of feature sets on the output quality.

Some aspects of the quality of generation are not captured by simple numerical comparisons, therefore I propose an alternate way of assessing the performance of the Learning to Rank model based on the comparison between an original text and its re-generated version by means of automatic metrics. This method of evaluation helps gaining a better perspective on the effect of variating the many parameters of

[2] http://www.gnu.org/software/octave/
[3] https://github.com/valeriobasile/listnet
[4] http://www.gnu.org/software/make/

the models on the final surface form returned by the system, and it also gives an idea of the overall quality of the word order generated by the system.

The results presented in this section are obtained by training the models on the data set described in Section 5.2.1. Several models are trained on data sets consisting of different combinations of features sets, as described in Section 5.1.2.

5.3.1 Intrinsic Evaluation

The first experiment is a typical cross-validation setup where a statistical model is trained on a data set and used to predict the target values on a test set. Since both the training set and the test set are gold standard, the quality of the model prediction can be measured against the test set, using an appropriate metric.

The results of the binary classification experiment are evaluated in terms of accuracy, that is, percentage of correctly predicted values. This task is expected to be easier for event referents, because the tuples relative to this type of discourse referent almost always take part in the realization process. The results confirm this. For the classification of tuples relative to entities, the measured accuracy is 87.29 using the "structure" feature set, while for event nodes the accuracy is 98.18. Including the "semantic" features helps a little, raising the accuracy of the classification for entity-referents and event-referents respectively to 87.68 and 98.23. Since this binary classification sub-problem is not the main focus of this thesis and not particularly hard, in the rest of this section I will focus on the results of experiments conducted using gold-standard binary classification. Given the reported accuracy, the binary classification will affect only marginally the final outcome of the combined system.

The performance of the Learning to Rank algorithm is evaluated in terms of Kendall's Tau [Kendall, 1938], a measure of rank correlation. Kendall's Tau (τ) measures the similarity between two rankings (two ways of ordering the same list of elements) by counting how many pairs of elements are swapped with respect to the original ordering out of all possible pairs of elements. Kendall's Tau is computed by the following formula:

$$\tau = \frac{\#concordantpairs - \#discordantpairs}{1/2n(n-1)}$$

where n is the number of elements to order. Given its definition, Kendall's Tau ranges between -1 and 1, i.e., two identical rankings have $\tau = 1$ while two rankings where one is the inverse of the others have $\tau = -1$.

This method of evaluating the order of constituents and the local order of phrases is similar to that of Bohnet [2007], already mentioned in Chapter 2. Although the author does not explicitly use the Tau measure, in the cited work the evaluation accounts for the number of switched pairs in the target ordering.

Table 5.2: Rank correlation of reordering different types of referents.

Referent Type	**Feature Set**	**Tau**
Entity	Structural	0.698
Entity	Structural + Semantic	0.691
Event	Structural	0.542
Event	Structural + Semantic	0.555

For each given list of tuples of the input, its predicted order is compared against the gold standard to compute the average correlation (Table 5.2).

Noticeably, the classification relative to event nodes proves to be a harder problem, compared to entity nodes, probably due to their sparse nature. The structure of the logical forms provide most of the learning material. Lexical semantics helps increasing the performance only a little for events but not for entities (rather, noise is introduced.

5.3.2 BLEU-based Evaluation

One problem with Kendall's Tau as a measure of performance in this task is that not all pair swaps affect the final surface form the same way. This problem is not due to Kendall's Tau *per se*, but rather it rises from the use of a rank correlation measure to evaluate the task at hand. A mistake in the order of just two words has the same impact on Kendall's Tau (or similar metric) as a mistake in ordering two longer constituents. This justifies the need for a string-based metric for evaluation that captures this kind of errors while simple order correlation does not. To illustrate the problem consider these examples:

- [the][man][running in the street] (correct) vs. [the][running in the street][man] (incorrect);
- [the woman][bought][a flower] (correct) vs. [a flower][bought][the woman] (incorrect).

While both expressions on the right side are incorrect, the first one feels "less wrong", while the second one has a completely different meaning from the original, and yet the intrinsic evaluation method above would give them the same degree of accuracy.

An alternative way of evaluating the outcome of the model is to use the predictions of the models to re-rank the edges of gold standard LDRGs, then feed it to the realization algorithm and compare the resulting surface forms with the original texts associated with the LDRGs. There are in literature many metrics for string-to-string comparison, starting from the simplest *edit distance* — the count of the number of insertion, deletion or substitution needed to transform a string into the other. Of course, for the evaluation of an NLG-related task, one would want a measure that works at least at the word level, possible taking linguistic aspects of the string into account. A distance metric of strings that is widely popular in the fields of Machine Translation and NLG is the BLEU score [Papineni et al., 2002]. BLEU is an algorithm that computes the quality of a candidate translation (with respect to a reference translation) or a generation (with respect to some gold standard surface form) by counting the number of occurrences of gold standard n-grams. The length of the n-grams is a parameter of the algorithm, thus by varying it different aspects of the generation quality are captured. For instance, BLEU-1 counts the occurrences of single words, basically computing the *precision* of the generation, so it gives an estimate of the coverage of the generation (is all lexical material generated?), while BLEU-4 counts how many 4-grams in the gold standard are correctly generated **in the correct order**, thus giving a good indicator of the fluency.

I applied the models trained with varying the feature sets to a dataset made of 2,560 sentences for which the realization algorithm presented in Section 3.5.3 was capable of outputting the correct surface form (i.e., the original sentence). For this experiment, the focus is on the prediction of constituent order, thus the gold-standard data for the binary realization classification (whether an edge in the DRG is generated or not) is used.

Table 5.3: BLEU-4 scores of re-generated LDRGs.

Feature Set	**Entity**	**Event**	**Entity + Event**
Structural	0.653	0.627	0.333
Structural + Semantic	0.660	0.626	0.333

Separate experiments were run, considering only events, only entities, and both events and entities. I then computed the average BLEU-4 score using the original sentences as reference text and their respective re-generated versions as candidates.

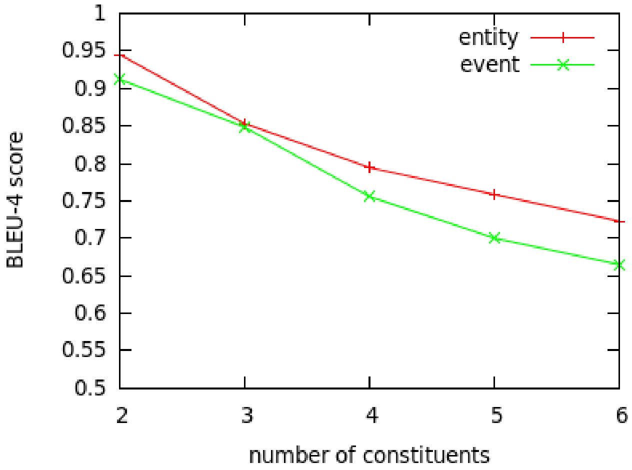


Figure 5.5: BLEU score effects caused by reordering the surface form of exactly one referent (of type entity and event), by number of constituents.

The results are summarized in Table 5.3, showing figures for the model with structural features and the one with both structural and semantic features.

From the results of this evaluation experiment, it can be seen how predicting word order for entities, as opposed to events, is generally an easier task. Intuitively, scrambling the order of tuples relative to an event referent has more chance to compromise the structure of a whole sentence. To get a better understanding of the effect of ordering errors on the BLEU score with respect to the number of constituents and type of discourse referent (entity or event), I ran an additional experiment. For each DRG in the test set the constituent order of exactly one discourse referent is predicted applying the models trained with the structural features and taking the gold-standard order for all remaining discourse referents. Then the sentences are re-generated with the realization algorithm. The computed BLEU-4 score is in Figure 5.5.

As expected, the more constituents are to be ordered, the more difficult the problem is, resulting in a lower BLEU score. More interestingly, on average, a single ordering mistake for an event has a higher impact than a single ordering mistake for an entity-type discourse referent. In the light of these findings, the results in Table 5.3 are indeed promising, the intuition being that small improvements in the algorithm could increase non linearly the quality of the output.

5.3.3 Error Analysis

Re-generation experiments and string-based metrics are efficient and automated ways to assess the performance of an NLG pipeline, but they do not provide the complete picture [Belz and Reiter, 2006]. Some errors in the prediction of word order are only apparent to the eye of the human judge. To reveal these errors, I manually inspected a hundred sentences from the test set containing at least one error, that is, incorrectly re-generated in the second experiment.

I found a total of 146 errors in the order prediction. Of these errors, roughly 36% happened when a relation is involved, in many cases a possessive (which is a very common phenomenon). This is probably due to the lack of lexical semantic information, that is, the WordNet-based features are not sufficient in terms of coverage, and there is no semantic feature that covers named entities. In 12% of the cases, the error was due to more than one attribute of an entity, e.g., two consequent adjectives, or to a noun-noun compound. Again, this type of error is bound to occur less when more lexical semantics is added, and often lead to marginally acceptable realizations as in the following example:

Original text
their government is bracing for a possible militant backlash

→

x_1 x_3 e_5 x_6

thing(x_1)
of(x_3, x_1)
government(x_3)
brace(e_5)
agent(e_5, x_3)
possible(x_6)
militant(x_6)
backlash(x_6)
for(e_5, x_6)

→

Re-generated Text
their government is bracing for a militant possible backlash

Some 28% of the incorrect re-generations involve “surface” tuples (17%), e.g., corresponding to adverbs in a noun phrase, or punctuation (11%). The rest of the errors occur inside a relative clause (14%) or a coordinated structure (10%). There are still cases in which a different word order is not necessarily ungrammatical or less fluent, such as:

Original Text
A joint Israeli-Egyptian team is investigating the incident.

→

$$\begin{array}{|l|} \hline x_1\ x_5\ e_6 \\ \hline \text{joint}(x_1) \\ \text{israeli-egyptian}(x_1) \\ \text{team}(x_1) \\ \text{incident}(x_5) \\ \text{investigate}(e_6) \\ \text{agent}(e_6, x_1) \\ \text{patient}(e_6, x_5) \\ \hline \end{array}$$

→

Re-generated Text
A Israeli-Egyptian joint team is investigating the incident.

Such alternative re-generations will be penalized by string-based metrics such as BLEU. However, based on the hundred examples that I looked at, they form just a small percentage of the whole dataset, and therefore it is my belief that they do not have a big effect on the actual results.

5.4 Alternative Approaches

The Unboxer's approach to word and constituent ordering during generation is in a way atypical. The emphasis on carrying out the two tasks of order prediction and lexical choice has the drawback of not being able to use the lexical information to drive the prediction of the word order. This is to say, this method uses only the structure of the semantic representation and gives it some form or ordering, without looking at the words themselves (but still leveraging the knowledge contained in the WordNet synsets).

The common approach in NLG follows from the traditional pipeline that sees the linearization task as part of the surface realization step, at the end of the generation process. In this framework the words, or at least the lemmata, are generally already in place and the task is that of giving them the right order. Starting from the earlier statistical NLG systems, this task has been approached by generating multiple candidate sentences, and subsequently ranking them according to their fluency — see for instance Langkilde and Knight [1998]. This is achieved typically by employing a *language model*, i.e., in its simplest form, a statistical model of the probabilities of sequences of words. A basic n-gram language model could consist in a long list of all the 3-grams or 4-grams found in a text corpus paired with the count of their occurrences, which gives an estimate of how probable a sequence of words is in a surface form. With such a resource, different surface forms generated by a system from the same abstract meaning representation can be ranked according to the probability of the occurrences of their n-grams. Finally, all the system has to do is to select the top ranked realization, deeming it as the most fluent. Consider the two following examples of realizations from the previous section:

“their government is bracing for a possible militant backlash” (correct)

“their government is bracing for a militant possible backlash” (wrong)

In a 3-gram model extracted from the GMB documents, the 3-gram *for a possible* is more than three times more likely to occur than *for a militant*, and the same goes for other 3-grams occurring in the two sentences. Using the 3-gram probabilities from the language model to compute the probability of the whole sentences, it is easy to verify that the first is more likely to be a natural language sentence than the second one, thus it should be selected as realization by a system that adopts the overgenerate-and-rank approach.

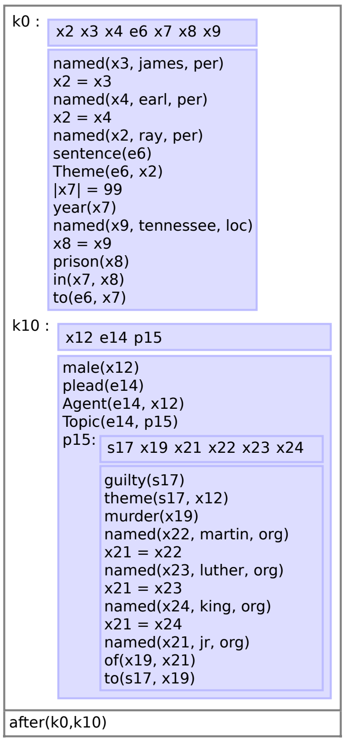
5.5 Discussion and Conclusion

Two important factors that contribute to lower the performance reported in this paper are the size of the training set and the quality of annotations. Although the numbers reported in Section 5.2 suggest a large enough dataset for the experiments, it must be noted that for a ranking problem the number of training examples required is typically larger, as the unit is represented by the lists of tuples to be ordered.

The Groningen Meaning Bank represents an effort towards a gold standard of linguistic annotation of the English language; a Web interface provides a tool to manually correct annotations produced by a toolchain of linguistic and semantic analysis systems. Even if the quality of the annotations is gradually improving, there are still many errors in the data, at several levels. This of course has an impact on the performance of the model(s), which assumes the analysis to be correct.

To make a complete surface realization system for DRSs, along with a component predicting word order (presented in this chapter), one also needs a lexical choice component that takes care of generating the actual (content) words. This will be the topic of the next chapter.

The approach presented in this paper can be extended to a third type of discourse referent, that is, discourse units. In DRT, discourse units have the hybrid nature of containers and units of information at the same time. The surface form of a complex sentence, or even a complete text, can be derived in a compositional way by generating surface forms for discourse units and linking them together following structural relations between them (e.g., subordination) or discourse relations (e.g., causation, continuation, implication, etc.). However, in most cases the lists of tuples



```
k0 :  x2 x3 x4 e6 x7 x8 x9

      named(x3, james, per)
      x2 = x3
      named(x4, earl, per)
      x2 = x4
      named(x2, ray, per)
      sentence(e6)
      Theme(e6, x2)
      |x7| = 99
      year(x7)
      named(x9, tennessee, loc)
      x8 = x9
      prison(x8)
      in(x7, x8)
      to(e6, x7)
k10 :  x12 e14 p15

       male(x12)
       plead(e14)
       Agent(e14, x12)
       Topic(e14, p15)
       p15:  s17 x19 x21 x22 x23 x24

             guilty(s17)
             theme(s17, x12)
             murder(x19)
             named(x22, martin, org)
             x21 = x22
             named(x23, luther, org)
             x21 = x23
             named(x24, king, org)
             x21 = x24
             named(x21, jr, org)
             of(x19, x21)
             to(s17, x19)
after(k0,k10)
```

Figure 5.6: DRS for “James Earl Ray was sentenced to 99 years in a Tennessee prison after he pleaded guilty to the murder of Martin Luther King Jr.”

relative to discourse unit referents only comprise one element, a fact that may lead to very sparse data for the task of learning their correct order. To understand the Unboxer ordering model applied to the discourse level, consider the following example sentence: “James Earl Ray was sentenced to 99 years in a Tennessee prison after he pleaded guilty to the murder of Martin Luther King Jr.”. For this sentence, the analysis provided by boxer is shown in Figure 5.6 as a segmented DRS.

From the picture it is easy to see that the semantic representation of the example sentence is made of two discourse units linked by the relation “after”. This is reflected in the corresponding DRG (not shown here for it would take too much space) where the tuple <k_0, after, k_{10}> appears, k_0 and k_{10} being the identifiers of the two discourse units respectively. With this information in the logical form, one can extend the approach to predicting the constituent order presented here to the prediction of the order of discourse units, perhaps with additional features such as discourse relations.

With respect to the evaluation method presented in this chapter, the BLEU score metric, although widely used to evaluate the fluency of automatically generated texts, is not optimal, as it penalizes alternative but permissible constituent orders. On the other hand, manual evaluation is expensive and time consuming. One promising way could be to devise a "game with a purpose", perhaps included in the Wordrobe collection of games (see Section 4.4.3). Such a judgment game could be implemented by presenting two version of a sentence (perhaps the gold standard and the generated one) to the players and ask them to rate their fluency.

Finally, it would be interesting to compare the performance of the ordering approach presented in this chapter with a traditional overgenerate-and-rank approach as described in Section 5.4. However, even though the quality of the final generation can be compared across the two methods, they presuppose different architectures of the respective NLG pipelines, so the comparison would be meaningful only up to a certain extent.

From the analysis of the results of the experiments presented in this chapter, I found that Discourse Representation Structures as an input format for NLG contain enough information to train a model that correctly predicts word order, to a sufficient extent. Within the framework of DRG-based generation, while the task of predicting whether or not a tuple takes part in the generation process is an easy one, to order them correctly is hard, especially with respect to event referents. I have also found that the method of evaluation of such ranking models based only on a correlation coefficient is incomplete, and an empirical, extrinsic evaluation, e.g., based on re-generation and BLEU-score, helps gaining new insights.

Chapter 6

Generating Words from Concepts

The word order aspect of Natural Language Generation constitutes a fundamental block of the whole process, as seen in the previous chapter. However, an equally important piece is still missing, that is, the production of the actual words that form the output of an NLG system.

The lexical ambiguity aspect of language makes it necessary for the designer of an NLG pipeline to take into account the problem of *lexical choice*, that is, the task of picking the most appropriated words to produce as part of the final surface output. Moreover, the words must be produced in the correct morphological *form*, as opposed to generate the base form (or *lemma*), a sub-problem that can be more or less easy to solve depending on the characteristics of the target natural language.

In this chapter the problem of lexical choice is instantiated as part of a more general NLG architecture, specifically the Unboxer NLG pipeline introduced in the previous chapters of this thesis.

The main questions the chapter will try to answer are the following four:

- Is the formalism employed in the Groningen Meaning Bank to represent semantics well-suited for linguistic generation, in particular with respect to the way it encodes concepts and events?
- Can the problem of lexical choice be solved by a knowledge-based, unsupervised algorithm producing generally acceptable surface forms?
- A supervised model based on aligned data is presented in Chapter 4. Can it also be used to approach the lexical choice task?
- Can the morphological information, i.e., word inflections, also be generated in a supervised fashion using similar features?

I will start with presenting the problem and its context (Sections 6.1 and 6.2) and

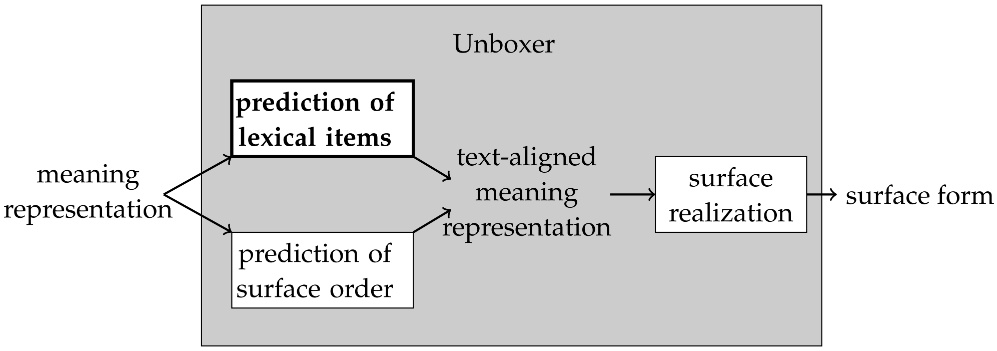

Unboxer
prediction of lexical items
meaning representation
text-aligned meaning representation
surface realization
surface form
prediction of surface order

Figure 6.1: A schematic view of the architecture of the Unboxer system with the lexicalization module object of this chapter highlighted.

motivating the choices made for the input format with respect to the representation of concepts (Section 6.4). I will then present a novel unsupervised algorithm to solve a significant part of the lexical choice problem in Section 6.5, and its supervised counterpart, trained on the Groningen Meaning Bank data in Section 6.6. The aspects related to the lexical morphology such as the generation of word inflections, are briefly treated in Section 6.7. Finally, Section 6.8 presents conclusions and future work.[1]

6.1 Introduction

The lexicalization module is responsible for the task of generating the lexical material from an abstract representation of meaning. In this chapter, this task is presented in the framework of the Unboxer pipeline for NLG, introduced in Chapter 3. The role of the lexicalization module in the system architecture is shown in Figure 6.1.

Many specialized Natural Language Generation systems are designed to convey messages relative to a given, self-contained domain. In such a context one may find very often a direct one-to-one mapping between concepts and words. After all, technical lingos evolve or are designed to be free of lexical ambiguity in situations where understanding each other with reliability is critical—in places like cruiser ships or operation rooms misunderstandings can have serious consequences and must be avoided.

[1] This chapter is partly based on, and is an extension of, the work published in Basile [2014].

k_1	unary	$\neg$		
$\neg$	scope	k_2		
k_2	referent	e_1		
k_2	referent	x_1	1	**A**
k_2	event	PAY		
k_2	concept	CUSTOMER		
k_2	role	$agent$		
CUSTOMER	instance	x_1	2	**customer**
PAY	instance	e_1	4	**pay**
$agent$	internal	e_1	1	
$agent$	external	x_1		
k_2	surface	e_1	2	**did**
k_2	surface	e_1	3	**not**
k_2	surface	e_1	5	**.**

Figure 6.2: Word-aligned DRG for "A customer did not pay." The word-alignment information is highlighted.

Having exactly one word to express one concept is of course handy from an NLG perspective, but obviously the constraint is not valid anymore once we move to an open-domain system whose intended output is natural language. There is in fact a many-to-many relation, in general, between concepts and the words that express them: a *polysemous* word is a word that has many senses, while a concept may be expressed, in general, by different *synonym* words. For example, the word "rock" is polysemous, as it can refer to a lump of mineral matter or to the popular musical genre. Conversely, the first of the two senses can also be expressed by the word "stone", which is a synonym of "rock".

To show an instance of this problem, consider the DRG already used as an example in Chapter 5 and shown again in Figure 6.2. Now, the focus is shifted from the fourth column, encoding the surface order as discussed in Chapter 5, to the fifth column, which in a gold standard text-aligned DRG contains the words that compose the surface form. The non-logical symbols CUSTOMER and PAY are predicates over entities in the meaning representation, and they need to be linked to some kind of world-knowledge in order for the whole structure to represent actual information. The symbol CUSTOMER, for instance, may refer to an entity defined as "someone who pays for goods or services", that can be realized in different ways depending on the circumstances (*customer* or *client* in English, *cliente* in Italian, and so on).

To define the problem, let's begin with defining what it is given (the input) and

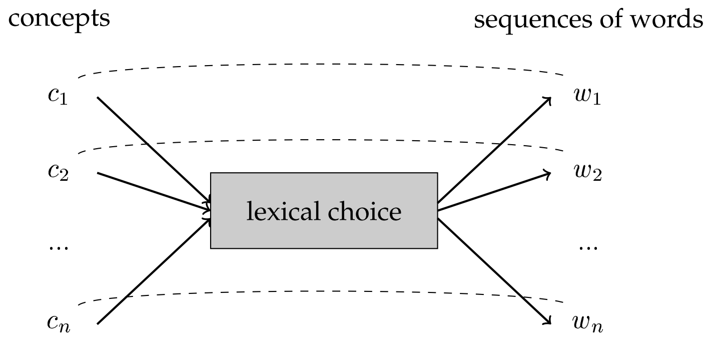


Figure 6.3: High-level depiction of the lexical choice problem with its input and output. The dashed lines represent the alignment between entities and words.

what a solution should look like (the output). As said in the first part of this thesis, the input to an NLG pipeline, regardless of difference in format and level of abstraction, is some kind of formal representation of information. Such structures contain, among other elements, items representing the *entities* involved in the piece of information to express in natural language. They can be already encoded as words, in which case the system does not have much work left to do for their realization, or they can be more sophisticated bits of information such as links to items in a knowledge base, e.g., a computational ontology. The set of concepts for which the NLG pipeline must produce words is the input to the lexical choice task. The output of the task is made by a set of natural language words, each being the expression of one of the concepts in the input set. Actually, a sequence made of more than one word can express a single concept, for instance with a multiword expression such as "domestic animal" or "body of water". A simplified representation of the lexical choice problem in terms of its input and output is depicted in Figure 6.3.

In general, the input to the lexical choice task is independent from a specific language, while the output is instead bound to one language. For instance, the entities linked to the non-logical symbols CUSTOMER and PAY in the example in Figure 6.2 are realized respectively by the words "customer" and "pay", an alignment that is only apparently trivial, considering that the non-logical symbols in the DRG are encoded as English words for the sake of readability. In fact, the representation of concepts is in principle independent of the target language of the generation process.

While this chapter focus on the realization of *concepts* specifically, the task of lexicalization in a NLG system extends to events, relations, attributes, and so on.

The solution to the lexicalization of every bit of an abstract meaning representation is left out of this thesis, but it is possible to speculate that the methods presented in the rest of this chapter can be adapted to components of a meaning representation other than concepts.

6.2 The Lexical Choice Problem

Many linguists argue that true synonyms don't exist [Bloomfield, 1933, Bolinger, 1968]. Yet, words with similar meanings do exist and they play an important role in language technology where lexical resources such as WordNet [Miller, 1995, Fellbaum, 1998] employ *synsets*, sets of synonyms that cluster words with the same or similar meaning. The internal structure of WordNet is detailed in the next section, along with an explanation of the reason why it constitutes a valuable resource for generation.

It would be wrong to think that any member of a synset would be an equally good candidate for every application. As an example to explain this phenomenon, consider the two sentences below extracted from the GMB (documents 01/0423 and 06/0358 respectively):

1. General Barno said he expects a small hard-core **remnant** of the Taleban to continue fighting even as the group's military strength fades away.

2. The conflict is also fueled by ethnic hatred **leftover** from the 1994 slaughter of Tutsis in neighboring Rwanda as well as Congo 's civil wars .

The two words highlighted in bold face are tagged in the corpus with the same WordNet synset, that is, they represent the same concept, defined as "a small part or portion that remains after the main part no longer exists". However, the reason why the concept is realized in different ways is not casual, but rather it depends on factors like linguistic context, topic, register, author style, age of the text, and possibly more. Without entering the domains of psycholinguistics and philology, it is simple to show with an informal experiment that a native speaker of English would find the sentence "General Barno said he expects a small hard-core **leftover*** of the Taleban to continue fighting" awkward and not natural. [2]

[2] I actually asked a number of native speakers of English their opinion. All eight of them preferred the version of the sentence with the word "remnant".

Here is another example concerning the synset {food, nutrient}, a concept whose gloss in WordNet is "any substance that can be metabolized by an animal to give energy and build tissue". In the two sentences below, extracted from the GMB (documents 00/0215 and 00/0697 respectively), this concept is realized as "food" in the first, but as "nutrient" in the second.

1. It said the loss was significant in a region where fishing provides a vital source of **food**.

2. The Kind-hearted Physician administered a stimulant, a tonic, and a **nutrient**, and went away.

Again, it is not immediate to pinpoint the exact reason for the choice of words. Intuitively, "food" feels related in meaning to "fishing" and "vital", while "nutrient" is in the same realm as "physician" and "tonic", despite the two words being used to indicate the same concept.

These examples show how the problem of picking the right word in the right context is not trivial, even when it is circumscribed to choosing between a fixed list of given items. However, in some cases there are linguistic clues that seem to be helpful in restricting the set of possible choices. For instance, in the second sentence above, "food" is a mass term and therefore cannot follow the indefinite article (putting aside for a moment the fact that typically the generation of articles depends on the generation of the relevant noun phrase first). The solution that comes to mind is that of leveraging the word n-gram frequencies extracted from existing corpora in order to assign some kind of "likelihood" score to every possible outcome of the generation process. Solutions of this type exploit statistical *language models*, created automatically by computing the probability of the occurrence of each word given the preceding words in the context. This way, the model can predict that, given for instance the sequence of words "please pass me the", the next word is more likely to be "salt" than "window". Langkilde and Knight [1998], for instance, employ n-gram language models to rank candidate realizations based on their plausibility according to the statistical model. In the example above *source of food* is arguably a more common English phrase than *source of nutrient*.

The language model approach, however, is only applicable when the final sentence is already given and the task is only that of generate the surface form of just one target synset. Given an abstract representation of meaning such as a DRG, a language model cannot be directly employed because the only information contained in the input structure is made of synsets, predicates, relations and logical symbols.

A system divided in modules like the Unboxer should solve a few other problems before the information needed to apply a language model is available:

1. Predict the surface order, as described in Chapter 5.
2. Predict the "surface" tuples to complete the alignment with semantically empty words and particles (see Section 3.4).
3. Generate all the possible combinations of words for all the concepts in the DRG, complete with their correct morphological inflections.

In other words, it would be necessary to generate the complete utterance first, to exploit features based on an n-gram model, and this is not always practical or even possible, at least not in an early stage of the pipeline. If the generation process is adapted to produce multiple generations for a given input, then without doubt statistical language models turn useful for ranking them according to their probability, thus selecting the version that resembles actually observed language the most. However, in the Unboxer architecture the tasks of predicting word order and predicting word forms are carried out independently from each other, so a language model cannot be applied as a part of the module for the generation of lexical items. A different solution is therefore needed that builds on the semantic content of a DRG alone.

The WordNet synsets that are linked to concept nodes in a DRG are good candidate representations of word meanings. WordNet could be seen as a dictionary, where each synset has its own definition in plain English. WordNet synsets are also well suited for lexical choice, because they consist in actual sets of lemmata, considered to be synonyms of each other in specific contexts. Thus, the problem is presented here in a form which is restricted to the *choice of lemmata from WordNet synsets*, although it will work with any other lexical resource with a similar structure.

The task of solving the lexical choice problem is an important one in the context of NLG from logical forms, but nevertheless is has not been broadly considered by the NLG community. One of the reasons is that it is hard to evaluate, because, as other lexical semantics phenomena, lexical choice is not always subject to crisp classification. In some cases, for instance, even human annotators cannot reach an agreement on exactly one lexicalization for a concept in a given context. Techniques based on information retrieval metrics (i.e., precision, recall and F-score) are often too strict, so they fail to capture not-so-wrong cases when a system produces a different lemma from the gold standard but still appropriate to the context. Not all instances of lexical choice are in fact problematic as the leftover|remnant example

```
minute(goal-1,34)
player(goal-1, player-1)
name(player-1, Ronaldinho)
minute(goal-2, 56)
player(goal-2,player-2)
name(player-2, Eto'o)
```

Figure 6.4: Example of the output of the content selection component of Bouayad-Agha et al. [2011].

above. In many cases the synonyms provided by a WordNet synset are effectively interchangeable in a given context. Despite the obstacles, however, there has been a number of studies and projects on the topic, briefly reviewed in the next section.

6.3 Related Work

Relevant literature on the representation of knowledge, can be found in Section 2.3 in the related work chapter of the thesis. In this section, I present previous work relevant to the lexical choice problem in the framework of NLG, in particular with respect to systems supported by computational ontologies.

Stede [1993] already pointed out the need to exploit semantic context, when investigating the criteria for lexical choice in NLG. Other authors try to solve the lexical choice problem by considering situational aspects of the communication process such as pragmatics [Hovy, 1987], argumentative intent [Elhadad, 1991] or the degree of salience of semantic elements [Wanner and Bateman, 1990]. In this thesis I favor the former approach over the latter, as it suits better the kind of structures from which the Unboxer generates natural language.

A whole line of research in NLG is focused on generation from ontologies, whether it be domain-specific applications or domain-independent generation systems like the Unboxer. Several works have underlined the benefits of a general concept hierarchy, such as the Upper Model [Bateman, 1997b] or the MIAKT ontology [Bontcheva and Wilks, 2004], to serve as pivot for different application-oriented systems. Bouayad-Agha et al. [2012a] employ a layered framework where an upper ontology is used together with a domain and a communication ontology for the purpose of robust NLG.

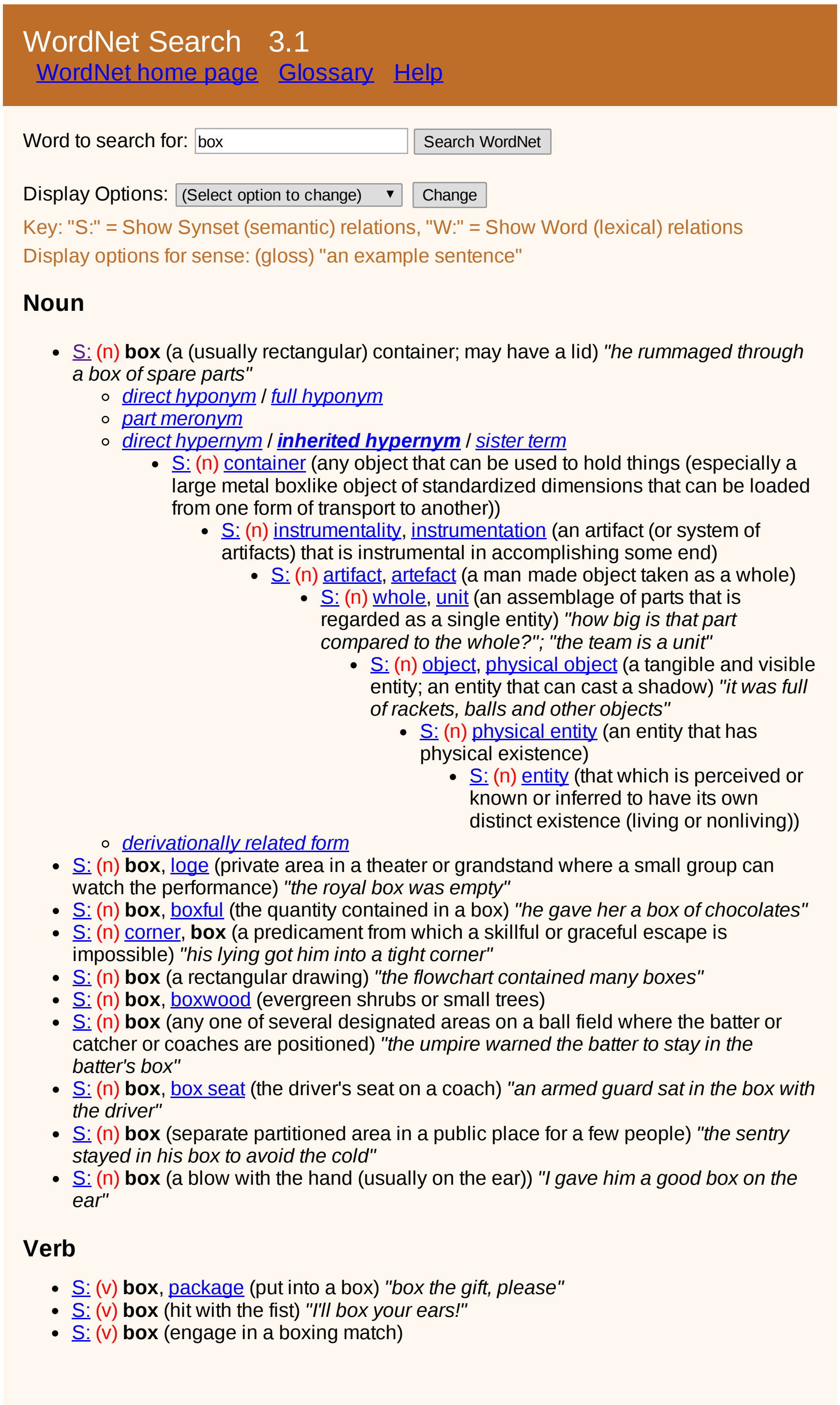
Many NLG systems make use of computational ontologies in the early stages of their generation process. Content planning modules typically produce a representation of the information to realize made by concepts taken from one or more ontologies. The concepts in the ontology provides the information items for the content planner, while the rules of the ontology specify how these can be put together in a coherent structure. As an example, Bouayad-Agha et al. [2011] use a computational ontology about the domain of the game of football as the basis of their content selection component. A sample output of the content selection is the set of six triples shown in Figure 6.4 describing the result of a match. The triples are then passed on to a discourse planner and finally to a surface realization component that will produce some natural language expression like "a goal by Ronaldinho in minute 34 and another goal by Etoó in minute 56". The concepts of *minute*, *goal*, *player* and so forth are specified in the underlying ontology.

WordNet can be seen as an upper ontology in itself (see Section 6.4), where the synsets are concepts and the hypernym/hyponym relation is akin to generalization/specialization. However, to the author's knowledge, WordNet has not been used so far as supporting ontology for a full-fledge generation pipeline, even though there exists work on the usefulness of such resource for NLG-related tasks such as domain adaptation and paraphrasing [Jing, 1998].

Statistical methods have also been employed to solve the lexicalization problem. Langkilde and Knight [1998] used a statistically derived n-gram model of the language in order to rank the possible output of their *Halogen* system. The candidate realizations are given a score based on how likely their n-grams are according to the language model, then the best candidate is picked by the system. Bangalore and Rambow [2000] improved on this approach by employing a tree model derived from a large corpus that is capable of predicting the likelihood of syntactic constructions.

6.4 WordNet as an Ontology for Generation

Discourse Representation Theory provides an excellent framework to encode the semantics of natural language from a formal logic standpoint (see Section 3.3.1). The formulas of DRT can be manipulated with the tools of formal logic, such as inference, and ultimately truth values can be computed Kamp [1984]. The DRG formalism introduced in Chapter 3 allows for a fine-grained alignment between the semantic structure and the surface structure, that in turn facilitates the processing with statistical methods. However, in order to encode knowledge about the real world, as opposed to a restricted domain, the need arises to ground the DRSs into



WordNet Search 3.1
WordNet home page Glossary Help

Word to search for: box [Search WordNet]

Display Options: (Select option to change) ▼ [Change]

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations
Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **box** (a (usually rectangular) container; may have a lid) *"he rummaged through a box of spare parts"*
  - *direct hyponym* / *full hyponym*
  - *part meronym*
  - *direct hypernym* / ***inherited hypernym*** / *sister term*
    - S: (n) container (any object that can be used to hold things (especially a large metal boxlike object of standardized dimensions that can be loaded from one form of transport to another))
      - S: (n) instrumentality, instrumentation (an artifact (or system of artifacts) that is instrumental in accomplishing some end)
        - S: (n) artifact, artefact (a man made object taken as a whole)
          - S: (n) whole, unit (an assemblage of parts that is regarded as a single entity) *"how big is that part compared to the whole?"; "the team is a unit"*
            - S: (n) object, physical object (a tangible and visible entity; an entity that can cast a shadow) *"it was full of rackets, balls and other objects"*
              - S: (n) physical entity (an entity that has physical existence)
                - S: (n) entity (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))
  - *derivationally related form*
- S: (n) **box**, loge (private area in a theater or grandstand where a small group can watch the performance) *"the royal box was empty"*
- S: (n) **box**, boxful (the quantity contained in a box) *"he gave her a box of chocolates"*
- S: (n) corner, **box** (a predicament from which a skillful or graceful escape is impossible) *"his lying got him into a tight corner"*
- S: (n) **box** (a rectangular drawing) *"the flowchart contained many boxes"*
- S: (n) **box**, boxwood (evergreen shrubs or small trees)
- S: (n) **box** (any one of several designated areas on a ball field where the batter or catcher or coaches are positioned) *"the umpire warned the batter to stay in the batter's box"*
- S: (n) **box**, box seat (the driver's seat on a coach) *"an armed guard sat in the box with the driver"*
- S: (n) **box** (separate partitioned area in a public place for a few people) *"the sentry stayed in his box to avoid the cold"*
- S: (n) **box** (a blow with the hand (usually on the ear)) *"I gave him a good box on the ear"*

Verb

- S: (v) **box**, package (put into a box) *"box the gift, please"*
- S: (v) **box** (hit with the fist) *"I'll box your ears!"*
- S: (v) **box** (engage in a boxing match)

Figure 6.5: The result of the search for *box* in the Wordnet 3.1 search Web interface.

some kind of database of facts and relations. In the GMB this is achieved by linking some of the predicates to WordNet synsets, a format that ultimately is the input format of the Unboxer.

WordNet was not created with the initial goal of being an ontology, but rather as an effort to prove psycholinguistic models about the mental organization of concepts. Nevertheless, the electronic lexical database has grown more and more popular among NLP scholars dealing with the meaning of words and their relations, and also among ontology experts. As a matter of fact, WordNet can be assimilated to an ontology by treating the hypernymy relation between synsets as *subsumption* between concepts, or in some cases as an *instantiation* relation between named entities (cities, countries, people, ...) and their hyponyms in WordNet. Gangemi et al. [2003b] went as far as defining a "complete formal specification of the conceptualizations expressed by means of Wordnet's synsets" in the OntoWordNet project.

In this section I argue that WordNet constitutes a solid choice as knowledge base for generation. The main argument is that some unique features of WordNet facilitate the NLG process as designed in the Unboxer pipeline, in particular the fact that concepts are represented in WordNet as sets of words, ready to be picked up for the generation of surface forms.

WordNet has been mentioned already in this chapter. Before going further, a few questions need to be answered. First, what is exactly WordNet? What are its components, its strengths and shortcomings? Second, is WordNet an ontology? If yes, what kind of ontology is it, and what are its peculiarities in that respect?

The book detailing the WordNet project is titled "WordNet: an Electronic Lexical Database" [Fellbaum, 1998], thus as a starting point the resource can be defined as a structured database of words in a format readable by electronic calculators. For each word in the database, WordNet provides a list of senses and their definition in plain English. The senses, besides having a inner identifier, are represented as *synsets*, i.e., sets of synonym words. Words in general belong to multiple synsets, as they have more than one sense, so the relation between words and synsets in WordNet is a many-to-many one. The synsets are grouped into four categories based on their parts of speech: noun, verb, adjective or adverb. WordNet is more than only an electronic dictionary though. As the "net" in the name suggests, WordNet not only contains words and their definitions, but also a whole set of relations defined among the word senses. In particular, the hyponymy relation between noun synsets induces a taxonomical structure of concepts.

```
<!-- Class:
  http://www.co-ode.org/ontologies/pizza/pizza.owl#QuattroFormaggi
-->

<owl:Class rdf:about="#QuattroFormaggi">
    <rdfs:label xml:lang="pt">QuatroQueijos</rdfs:label>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasTopping"/>
            <owl:someValuesFrom rdf:resource="#TomatoTopping"/>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Class rdf:about="#NamedPizza"/>
    </rdfs:subClassOf>
     <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasTopping"/>
            <owl:someValuesFrom rdf:resource="#FourCheesesTopping"/>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasTopping"/>
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:unionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#FourCheesesTopping"/>
                        <owl:Class rdf:about="#TomatoTopping"/>
                    </owl:unionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

Figure 6.6: A snippet of the Pizza ontology describing the class *QuattroFormaggi*.

Figure 6.5 shows a screenshot of the WordNet 3.1 search Web interface[3] used to search for the word *box*. Clicking on the *inherited hypernym* link under the first sense

[3] http://wordnetweb.princeton.edu/perl/webwn

of the word, the full hypernym chain is shown, all the way up to the root node *entity*.

An ontology is "an explicit specification of a conceptualization" [Gruber, 1993], a collection of facts about some domain of defined entities[4]. Ontologies vary on different dimensions, including their size, complexity, domain, and specificity. Some ontologies have complex logical formulas among their rules, while others are more shallow collections of classes. Rules in an ontology are if-then-like statements describing the logical inferences that can be drawn from assertions. Figure 6.6 shows a snippet of the Pizza ontology[5] developed at the University of Manchester for tutorial purposes, encoded in RDF/OWL, a popular XML-based format for computational ontologies. This portion of the ontology describes the class *QuattroFormaggi*, specifying that an instance of the class must have the properties *FourCheesesTopping* and *TomatoTopping*.

WordNet is then an ontology about words, senses, and a series of relations among them, while still being more ontology-like than, say, a machine-readable dictionary or a thesaurus. This last point is debatable, as technically WordNet is a lexical resource, and additional work is necessary to transform it into a formal ontology specified in some logic formalism [Gangemi et al., 2003a]. Nevertheless, for the purpose of designing an NLG system, the difference of definitions is not crucial.

In the Unboxer pipeline, the input (i.e., a DRG) is considered comparable to the output of a discourse planner. The elements of a DRG as a discourse plan come either from the theory (discourse units, symbols, roles, etc.) or from WordNet. The Unboxer approach, therefore, does not only employ WordNet as its supporting ontology, but rather WordNet plus the model given by Discourse Representation Theory together form the foundational ontology behind the Unboxer pipeline.

6.5 An Unsupervised Solution to Lexical Choice

In this section I introduce a method to tackle the lexical choice problem presented earlier in the chapter. This method comes in the form of an algorithm that makes use of an electronic lexical resource and deterministically decides what words to use to express concepts represented as synsets. A characteristic of the proposed method is that it is unsupervised, thus it can be applied to any new input without needing long and expensive training procedures or labeled data, but only some kind of WordNet-

[4] A more extensive definition of ontology in the field of computer science is given in the Encyclopedia of Database Systems [Liu and Özsu, 2009].

[5] `http://130.88.198.11/co-ode-files/ontologies/pizza.owl`

like electronic lexicon. Moreover, this method is in principle independent from the target language, provided that an appropriate lexical resource is available.

In a way, the problem of lexical choice for a concept can be seen as the inverse of word sense disambiguation [Weaver, 1949/1955]: instead of determining the right sense from a given inventory for a certain word, the problem is to decide which word of a synset is the best choice. In both problems, context is key, but while it is easy to see what the context looks like in the case of WSD (a window of words surrounding the target word, or perhaps the entire sentence in which the target word occurs), the idea of context in the case of lexical choice needs a precise definition. This section introduces a solution for the lexical choice problem based on an analogy with WSD where to generate the surface for a concept the set of synsets of the whole DRG is taken into account as semantic context.

6.5.1 Word Sense Disambiguation and Lexical Choice

Word sense disambiguation is the task of assigning senses to words, where the senses are usually taken from a fixed inventory such as a dictionary. This task is traditionally considered difficult for an automated system, for a number of reasons including the difficulty of representing word senses and the fact that a proper disambiguation of a word depends on the context where it occurs. WSD is a popular topic in NLP research, thus a plethora of methods of all kinds have been proposed in literature over the years. As often happens with other classification problems in NLP, it is fairly easy to achieve a decent performance on a WSD task by taking a simple or even trivial approach, while at the same time it is very difficult to improve significantly on such results.

As a case in point, the typical baseline employed to evaluate WSD algorithms consists of choosing the most frequent sense from the available inventory. In standard evaluation benchmarks, often systems do not achieve scores that are much higher than the baseline.

The *Lesk* algorithm is a classic solution to the WSD problem that, despite its simple scheme, achieves surprisingly good results by only relying on an external knowledge source, e.g., a dictionary [Lesk, 1986]. To disambiguate a word in a context, the algorithm considers all the definitions of the word itself as found in some dictionary, then counts how many words are there in common between the context of the word and each of the definitions. The sense corresponding to the definition that exhibits the greater word overlap with the context is chosen as the correct sense for the target word. Consider for instance the following sentence:

> I met her at a **party**, she was with a group of friends. Then we had dinner together.

WordNet provides five senses for the word **party**, with the following glosses:

1. an organization to gain political power "in 1992 Perot tried to organize a third party at the national level"
2. a group of people gathered together for pleasure "she joined the party after dinner"
3. a band of people associated temporarily in some activity "they organized a party to search for food"; "the company of cooks walked into the kitchen"
4. an occasion on which people can assemble for social interaction and entertainment "he planned a party to celebrate Bastille Day"
5. a person involved in legal proceedings "the party of the first part"

The words "group" and "together" occur in the context (the given sentence above) *and* in the second definition. The overlap is even greater if we consider the examples of usage too, i.e., the word "together" occurs in the second entry and also in the context. The three words overlap of the second gloss with the context is larger than that of all the other sense definitions, therefore, according to the Lesk algorithm, the second sense is selected as the correct one for this instance of "party".

Following the symmetry between lexical choice and word sense disambiguation stated in Section 6.2, the original idea for the lexical choice algorithm presented here comes from looking at this existing solutions for WSD and inverting it. The Lesk approach to WSD provided the inspiration to devise an algorithm that looks at the semantic similarity between candidate lemmata of a synset and its semantic context. The resulting algorithm for lexical choice from WordNet synsets is called *Ksel* (a word play on the name *Lesk*) and it is the topic of this section.

6.5.2 The Ksel Algorithm

Lesk computes the relatedness between the candidate senses for a lemma and the words in its proximity as a function of all the words in the sense definitions and the context itself. In the simplest case the aggregation is done by considering just word overlap. Over the years, refinements to the Lesk algorithm have surfaced, such as the approach by Banerjee and Pedersen [2003] that exploits not only the words in the sense definitions but also a set of related words extracted from WordNet relations. More recently, Basile, Caputo, and Semeraro [2014] proposed an enhancement to the Lesk approach where a word space model is used to overcome the need for exact word overlap of the original algorithm.

Similarly, Ksel computes a score for the candidate lemmata as a function of all the synsets they belong to and the *semantic context*[6]. Just as not every word in a synset gloss is relevant to the linguistic context, not every synset of a lemma will be related to the semantic context, but carefully choosing the aggregation function will weed out the unwanted elements. The intuition is that in most cases the synsets of a word in WordNet are related to each other, just as the words in a sense definition are often semantically related. For example, consider the seven senses of the noun "rock" according to WordNet:

1. *rock, stone* (a lump or mass of hard consolidated mineral matter) "he threw a rock at me"
2. *rock, stone* (material consisting of the aggregate of minerals like those making up the Earth's crust) "that mountain is solid rock"; "stone is abundant in New England and there are many quarries"
3. *Rock, John Rock* (United States gynecologist and devout Catholic who conducted the first clinical trials of the oral contraceptive pill (1890-1984))
4. *rock* ((figurative) someone who is strong and stable and dependable) "he was her rock during the crisis"; "Thou art Peter, and upon this rock I will build my church" — Gospel According to Matthew
5. *rock candy, rock* (hard bright-colored stick candy (typically flavored with peppermint))
6. *rock 'n' roll, rock'n'roll, rock-and-roll, rock and roll, rock, rock music* (a genre of popular music originating in the 1950s; a blend of black rhythm-and-blues with white country-and-western) "rock is a generic term for the range of styles that evolved out of rock'n'roll."
7. *rock, careen, sway, tilt* (pitching dangerously to one side)

[6] Here and in the rest of this chapter, the term *semantic context* refers to the set of WordNet synsets in the meaning representation, excluding the synset for which the algorithm is generating a lemma.

The first two senses are also the predominant ones and they refers to concepts that are similar in meaning. The third sense is actually a named entity. The fourth and fifth senses are somewhat related to the first two, one in a metaphorical kind of way (a person that is stable, unmovable as a rock) and the other as a simile (the candy is hard to chew just as a rock is hard to manipulate). Finally, the sixth and seventh senses are not related to the predominant senses, although they are arguably related to each other and to the *verb* senses of the word "rock":

1. *rock, sway, shake* (move back and forth or sideways) "the ship was rocking"; "the tall building swayed"; "She rocked back and forth on her feet"

2. *rock, sway* (cause to move back and forth) "rock the cradle"; "rock the baby"; "the wind swayed the trees gently"

The word "rock" is highly polysemous: in WordNet 3.0 only 699 out of 117,798 nouns (0.6%) has seven or more senses. Nevertheless, four out of its seven senses are linked by some kind of semantic relatedness.

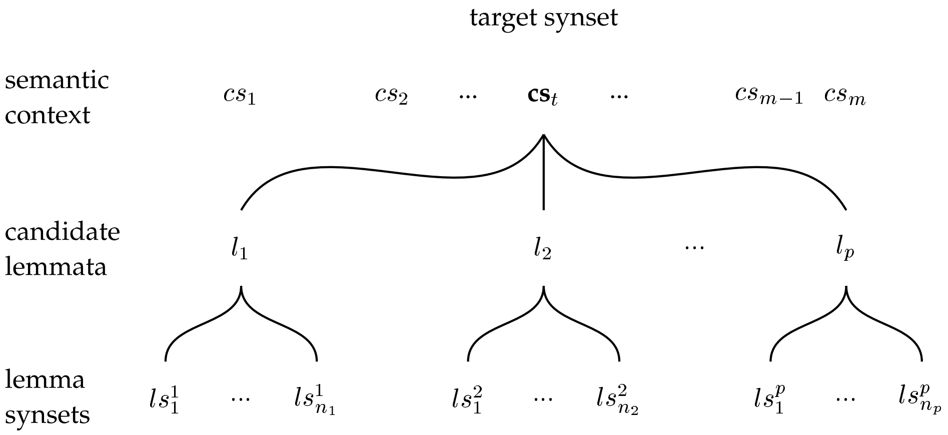


Figure 6.7: Elements of the Ksel algorithm.

The Ksel algorithm builds on this intuition about the average relatedness of the word senses. Here follows an explanation of how it works in practice. Referring to Figure 6.7, the task at hand is that of choosing the right lemma l_i among the candidates $l_1, l_2, ..., l_p$ for the target synset s_t. The other synsets given in input form the semantic context $C = s_1, ..., s_m, s_i \neq s_t$. The *lemma-synset* (LS) similarity between a lemma and a generic synset s_j is defined as a function of the similarities of all the synsets to which the lemma belongs and the synset under consideration:

$$s_{LS}(l_j, s_i) = f_1(sim(s_1, s_{j,k}) : 1 \leqslant k \leqslant n_j) \tag{6.1}$$

Using the lemma-synset similarity, the relatedness of a lemma to the semantic context (*lemma-context* similarity, LC) is defined as a function of the similarities of the lemma itself with the context synsets:

$$s_{LC}(l_j, C) = f_2(s_{LC}(l_j, s_i) : s_i \in C, 1 \leqslant i \leqslant m) \tag{6.2}$$

The three functions f_1, f_2 and sim are still not specified in the definitions above, because they are actually parameters of the algorithm. f_1 and f_2 are aggregation functions over a set of similarity scores, that is, they take a set of real numbers, typically limited to the $[-1, 1]$ interval representing similarity values, and return a value in the same interval. Sim is a similarity measure between WordNet synsets, like one of the many that have been proposed in literature — see Budanitsky and Hirst [2006] for a survey and an evaluation of WordNet-based similarity measures. A synset similarity measure is a function that takes two synsets and returns a score that indicates their similarity as a number between 0 (unrelated synsets) and 1 (identical synsets).The target lemma, according to the Ksel algorithm, is the one that maximizes the lemma-context similarity measure (6.2):

$$l_t = \arg\max_i s_{LC}(l_i, C) \tag{6.3}$$

Example To better clarify how Ksel works, here follows an example of lexical choice between two candidate lemmata given a semantic context. The example is based on the previously shown example about the sense-annotated sentence "The Kind-hearted Physician administered a stimulant, a tonic, and a food|nutrient, and went away.". The context C is the set of the synsets representing the meaning of the nouns "stimulant" (c_1 = {stimulant, stimulant drug, excitant}, "a drug that temporarily quickens some vital process"), "tonic" (c_2 = {tonic, restorative}, "a medicine that strengthens and invigorates") and "physician" (c_3 = {doctor, doc, physician, MD, Dr., medico}, "a licensed medical practitioner"). The target synset is cs_t = {food, nutrient} ("any substance that can be metabolized by an animal to give energy and build tissue"), for which the algorithm has to decide which lemma to generate between*food* and *nutrient*. *food* occurs in three synsets, while *nutrient* occurs in two:

- $s_{1,1}$: {food, nutrient} "any substance that can be metabolized by an animal to give energy and build tissue"

- $s_{1,2}$: {food, solid_food} "any solid substance (as opposed to liquid) that is used as a source of nourishment"
- $s_{1,3}$: {food, food_for_thought, intellectual_nourishment} "anything that provides mental stimulus for thinking"

- $s_{2,1}$: {food, nutrient} "any substance that can be metabolized by an animal to give energy and build tissue"
- $s_{2,2}$: {nutrient} "any substance (such as a chemical element or inorganic compound) that can be taken in by a green plant and used in organic synthesis"

Table 6.1: Running Ksel to select the best lemma between *food* and *nutrient* in a context composed of the three synsets s_1, s_2 and s_3.

lemma	**synset**	sim **(path similarity)**			**lemma-synset similarity (f_1 = mean)**
		c_1	c_2	c_3	
food	$s_{1,1}$	.200	.166	.090	.152
food	$s_{1,2}$	.142	.125	.090	.119
food	$s_{1,3}$	.090	.083	.071	.081
lemma-context similarity (f_2 = mean):					**.117**
nutrient	$s_{2,1}$	.200	.166	.090	.152
nutrient	$s_{2,2}$	.200	.166	.090	.152
lemma-context similarity (f_2 = mean):					**.152**

For the sake of the example, the basic WordNet path similarity measure is used, that is, the inverse of the length of the shortest path between two synsets in the WordNet hierarchy. Other similarity measures yield similar results in this case. For each synset of *food*, the algorithm computes the mean of its path similarity with all the context synsets. This represents an aggregate measure of the semantic relatedness between a lemma (i.e., all of its possible synsets) and the semantic context under consideration. Again, the choice of the mean as aggregation function is arbitrary here, made just for the sake of the readability of the example. Then the process is repeated with *nutrient*, and finally the algorithm chooses the lemma with the highest aggregate similarity score. The whole process and the intermediate results are summarized in Table 6.1. Since 0.152 is greater than 0.117, the algorithm picks *nutrient* as the best candidate for this semantic context.

6.5.3 Empirical Evaluation

In this section I describe the outcome of a test conducted in order to investigate how the parameters of the Ksel algorithm influence the performance on a dataset taken from the Groningen Meaning Bank. Recall that, in the GMB, concepts are linked to WordNet synsets. The test consists of generating a lemma for each concept of a DRG, comparing it to the correspondent gold standard lemma, and computing the average precision and recall over the set of documents.

Ksel has three parameters functions. For the two aggregating functions, I experimented with mean, median and maximum. For the WordNet similarity measures between synsets, I took advantage of the Python NLTK library[7] that provides implementation for six different measures on WordNet 3.0 data:

- Path similarity (path), based on the shortest path that connects the synsets in the hypernym/hypnoym taxonomy.
- Leakcock & Chodorow's measure (LCH), which takes into account the maximum depth of the taxonomy tree [Leacock and Chodorow, 1998].
- Wu & Palmer's measure (WUP), where the distances are computed between the target synsets and their most specific common ancestor [Wu and Palmer, 1994].
- Three methods based on Information Content: Resnik's measure (RES) [Resnik, 1995], Jiang's measure (JCN) [Jiang and Conrath, 1997] and Lin's measure (LIN) [Lin, 1998].

In the case of WSD, a typical baseline consists of taking the most frequent sense of the target word. The most frequent sense baseline in WSD works well (see for instance Navigli et al. [2007]), due to the highly skewed distribution of word senses, i.e., typically very few instances of a given word deviate from its most common meaning. I investigated if the intuition behind the most frequent sense baseline is applicable to the the lexical choice problem by reversing its mechanics, that is, the baseline looks at the frequency distribution of the target synset's lemmata among the synset occurrences in the data and selects the one that occurs more often.

The implementation of Ksel is run on a dataset comprising 1,000 randomly chosen DRGs from the GMB (32,764 concepts in total), with the goal of finding the best combination of parameters. Three alternatives for the aggregation functions and six

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

Table 6.2: Comparison of the performance of the Ksel algorithm with two baselines.

Method	**Accuracy**
Random	0.552
Most Frequent Lemma	0.748
Ksel (median, median, RES)	0.777

Table 6.3: Accuracy scores of the Ksel algorithm measured by the experiment from Section 6.5.3 with all possible combinations of parameters.

f_1	f_2	sim	**Accuracy**	f_1	f_2	sim	**Accuracy**
mean	mean	path	0,735	mean	mean	RES	0,625
mean	median	path	0,731	mean	median	RES	0,741
mean	max	path	0,734	mean	max	RES	0,584
median	mean	path	0,746	median	mean	RES	0,628
median	median	path	0,754	median	median	RES	0,777
median	max	path	0,748	median	max	RES	0,624
max	mean	path	0,775	max	mean	RES	0,705
max	median	path	0,775	max	median	RES	0,761
max	max	path	0,777	max	max	RES	0,727
mean	mean	LCH	0,585	mean	mean	JCN	0,757
mean	median	LCH	0,604	mean	median	JCN	0,759
mean	max	LCH	0,551	mean	max	JCN	0,758
median	mean	LCH	0,599	median	mean	JCN	0,757
median	median	LCH	0,644	median	median	JCN	0,759
median	max	LCH	0,595	median	max	JCN	0,758
max	mean	LCH	0,684	max	mean	JCN	0,775
max	median	LCH	0,705	max	median	JCN	0,777
max	max	LCH	0,730	max	max	JCN	0,776
mean	mean	WUP	0,611	mean	mean	LIN	0,601
mean	median	WUP	0,621	mean	median	LIN	0,662
mean	max	WUP	0,562	mean	max	LIN	0,606
median	mean	WUP	0,617	median	mean	LIN	0,601
median	median	WUP	0,671	median	median	LIN	0,671
median	max	WUP	0,600	median	max	LIN	0,610
max	mean	WUP	0,709	max	mean	LIN	0,732
max	median	WUP	0,714	max	median	LIN	0,755
max	max	WUP	0,728	max	max	LIN	0,746

different similarity measures result in 54 possible combination of parameters. For each possibility, the accuracy relative to the gold standard lemmata in the data set corresponding to the concepts is reported. The best choice of parameters resulted to be the median for both aggregation functions and the Resnik's measure for synset similarity.

The next step is a comparisons between Ksel (with best-performing parameters), a baseline that selects one uniformly random lemma among the set of synonyms, and the most frequent lemma baseline described earlier. The results of the experiment are presented in Table 6.2, showing how Ksel outperforms the baseline, even though perhaps not in a significant way.

The results of the experiment carried out with all possible combinations of parameters are listed in Table 6.3. From the figures, it seems that the interactions between the parameters are not linear, for instance the combination $f_1 = max, f_2 = median$ is the one that on average achieves the best result, although if $sim = RES$ then $f_1 = median, f_2 = median$ works slightly better. However, the sample on which the algorithm has been tested is perhaps too small to draw definitive conclusions on this matter.

The experimental test presented in this section shows the potential of the Ksel algorithm. The main strength of this method for lexical choice is that of being completely unsupervised. In fact, it does not require a manually labeled dataset, whereas other methods do, including the most frequent lemma baseline against which the performance of Ksel is tested. However, the performance itself is not much better than the baseline, and the difference in the scores is likely not significant.

To conclude, the novel algorithm for lexical choice introduced in this section works well as a bootstrap method when no annotated data is available. In the next section I present an alternative method based on a supervised statistical model that exploits labeled data to learn the best lexical items to realize concepts, encoded as synsets in a given semantic context.

6.6 Supervised approach to Lexical Choice

The previous section introduced a self-contained solution for lexical choice, or at least the particular flavor of lexical choice that takes place in the NLG pipeline proposed in this thesis (see Chapter 3). However, we know from Chapters 4 and 5 that the proposed system is supervised, and that a corpus providing enough data to train the system does exists. Following these considerations, it appears natural to investigate whether a supervised approach to the lexical choice task is feasible.

For each entity to generate, the DRGs in the GMB provide the link to a synset in WordNet and the alignment with the surface form provides the actual words. For now, and for the rest of this section, let's assume that the alignment contains lemmata rather than word forms, for the sake of the simplicity of the model. This assumption is not restrictive with respect to the goal of natural language generation, as will be clear later in the chapter (see Section 6.7).

In Chapter 5 a series of classifiers are employed to predict part of the alignment, specifically the information about word and phrase order, by learning it from the gold standard data. In a similar way, a set of features can be extracted for each entity in the corpus (thus for each WordNet synset), encoded, and fed to a Machine Learning algorithm with the goal of predicting the most suitable lemmata for each concept in the DRG. This section presents the first steps carried out towards this direction. By the end of this section it will be clear that the supervised approach to lexical choice is technically feasible, although hindered by data sparseness.

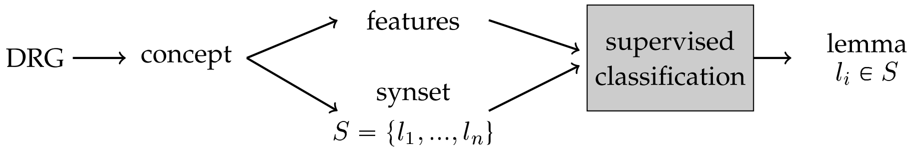


Figure 6.8: Supervised model to predict lemmata for the concepts in a DRG.

A rough design of the proposed solution is sketched in Figure 6.8.

The supervised statistical model for lexical choice is a component of the Unboxer NLG pipeline that is able to take a DRG and a target concept (which is part of the DRG itself) and produce the appropriate lemma to use in later steps of the generation process. Being a supervised method, it relies on an annotated dataset of DRGs to train its statistical model. In the rest of this section the method is presented in details, along with the result of an experiment designed to test the method.

6.6.1 A Supervised Model for Lexical Choice

A significant difference with respect to the supervised classifiers for predicting word order presented in the previous chapter is that in this case the possible outputs do not form a consistent class. This is to say that while the output of a ranking classifier is always a sequence of integer numbers, the set of possible values for a lexical choice component varies based on the input synset — each different synset corresponds to its specific set of lemmata. The consequence of this consideration is that either a dedicated classifier is needed for each particular type of input, or some form of abstraction has to be implemented. The method presented here explores the former approach, that is, specialized statistical models are built for each concept for which a lemma has to be generated.

The idea behind the supervised approach to lexical choice presented in this section is that the information contained in the meaning representation surrounding a concept can help to predict how such a concept is expressed in natural language. This hypothesis is also at the basis of the unsupervised method presented in the previous section. However, while the Ksel algorithm only exploits *shallow* semantic features, i.e., word senses and their place in the WordNet taxonomy, a supervised classifier for lexical choice would take as input features extracted from a *deep* semantic analysis. For example, consider the synset {rock, stone}, defined by WordNet 3.1 as "a lump or mass of hard consolidated mineral matter". The working hypothesis of the supervised method for lexical choice is that the concept encoded by the synset {rock, stone} is expressed by either "rock" or "stone" based on what other concepts, events, attributes are in the same abstract meaning representation structure, what relations they are involved in and what role they play in them, and other features extracted from the *semantic* context, as opposed to the *linguistic* context that at this point of the generation pipeline is not known.

6.6.2 Data and Features

In order to test the feasibility of the supervised model for lexical choice, I collected a dataset made of text-aligned DRGs taken from the GMB. The dataset consists of 252,214 concepts extracted from 9,161 DRGs. Each concept corresponds to one of 6,348 synsets. This results in an average of about 39 occurrences of each synset, although this measure is not meaningful considering the skew of the distribution of the number of the occurrences. To give an idea, one third of the synsets occur ten times or less in the dataset while one fourth only occur once. On the other hand, the most frequent concept in the dataset, corresponding to the synset {official, functionary}, occurs 4,209 times. [8]

The datasets corresponding to the synsets contain a variable number of rows, one

[8] Due to the nature of the documents that are collected in the GMB, a great deal of content is about the U.S. government and administration.

for each instance of the synsets found in the corpus, and a fixed number of columns representing the values of the features extracted for each instance, plus the gold standard class, i.e., the lemma as found in the data. The features are extracted from the "neighborhood" of the concept node in the DRG network structure, in a way similar to the supervised method for learning word order presented in Chapter 5. Here is a list of the features extracted for each concept found in the DRGs:

- The *categories* of all the synsets corresponding to concepts in the DRG. Following WordNet's hypernym/hyponym taxonomy, the category of a synset is considered as the synset at the third-level from the top of the taxonomy that appears in the hypernym chain going from the target synset to the root.
- The type of each inward edge attached to the concept node in the DRG.
- The binary relations in which the concept is involved.
- The type of discourse unit in which the concept occurs.
- The types of the other discourse referents that participate in relations with the target concept.
- Logical operators, if the target concept is in their scope.

These features capture the structural information local to the discourse referent in the meaning representation for which the system has to generate a lexical expression. It is difficult to represent the whole structure, recursive and varying in size by design, in a vectorial format of fixed dimensionality suitable for machine learning, therefore some information is necessarily lost. In particular, the features listed here do not capture the information attached to the discourse referents that are not in a direct relationship with the target discourse referent. Nevertheless, in the next section an experiment is presented that has been carried out in order to test the feasibility of the supervised method for lexical choice introduced here.

6.6.3 Experimental Setup

The experiment has been carried out using the Weka framework, an integrated environment for machine learning that includes many types of classifiers, options and related tool [Witten et al., 1999]. For this experiment, a SVM classifier is created for each synset, trained on the correspondent dataset, and the performance of the classification is tested by means of ten-fold cross-validation.

As a first step, the dataset of DRGs is processed and the features are extracted, put into numeric vector format and exactly one file is constructed for each synset found in the dataset. Next, a script loops over the synset datasets and feed each of them in turn into Weka.

The result of a run of the experiment is computed by training the model on a portion of the dataset (90% of the instances, in this case), applying the model to predict the gold standard labels of the remaining part of the dataset, and finally computing the accuracy of the prediction. This process is repeated ten times, each time changing the parts of synset dataset that is used for training and testing (ten-fold cross-validation) and then repeated for each synset in the global dataset to get the final result as the average of the accuracy of each classifier.

In the experiment carried out in the setting described so far, the classifiers created for each synset were not able to correctly predict lemmata other than the most common ones. Specifically, they predict always the most common lemma for each synset, thus performing just as good as the baseline used in the experiments in Section 6.5.3.

6.6.4 Results and Discussion

The result of the experiment presented in this section is a negative one, that is, the statistical approach to lexical choice as formulated here does not give satisfactory results. This outcome could be caused by several factors. First and foremost, data sparsity is an issue, i.e., there is simply not enough data for a large number of synsets to train a lexical choice classifiers. Other characteristics of the dataset that hinder the success of a statistical approach are the skewed distribution of the number of occurrences of the synsets and the skewed distribution of lemma occurrences for most synsets.

Clearly, the shape of the data is an issue that must be addressed in order for a statistic approach to lexical choice to work properly. To give an idea of the phenomena that one has to deal with when exploring senses and lemmata, in the present dataset 5,761 out of 6,348 synsets (more than 90%) are realized by one lemma only. Ignoring these mono-lemma synsets, the realized lemma corresponds to the most frequent lemma in 77% of the occurrences, on average. These figures do not leave much space to learn the correct lexicalization from the data.

Data sparsity is not likely to be the only issue that prevents this method from working properly. The use of a silver standard in terms of word senses causes a number of lemmata in the dataset to be annotated with the synset corresponding to their most frequent sense, thus a contributing to the skew of the sense distribution, and in general degrading the performance of a supervised method.

Finally, it is not clear whether the features extracted from the semantic context of the concepts, that is, their surrounding nodes and edges in the DRG, contain enough information to predict their realization. In some cases, information found farther away from the discourse referent node in the DRG might contribute to inform the model. The exploration of such features and their implementation in the supervised model is left as future work.

6.7 Generation of Word Inflections

Up to this point in this chapter, the difference between *word* and *lemma* has not been the focus of discussion. Technically, the basic unit at the lexical level is the *lexeme*, that is, the set of forms that can be assumed by a word. One of such word forms is considered the "base form" and called lemma. Commonly used dictionaries, for instance, are indexed by lemmata.

In the previous sections, the problem of lexical choice has been considered solved if the solution provides a suitable lemma for the each concept to generate. Obviously this approach is incomplete and would lead to clunky, ungrammatical sentences. Take for instance the following Chinese proverb:

> "Kindness in word**s** create**s** confidence. Kindness in think**ing** create**s** profoundness. Kindness in giv**ing** create**s** love" —Lao Tzu

In this example, the suffixes -ing and -s are highlighted, underlining how some of the words (e.g., giv-ing) deviate from their base form ("give", in this case). The grammar of the English language specifies that some words in certain cases must be *inflected*, i.e., their *form* is different from the lemma. Languages other than English are even more inflected, with their words assuming a variety of morphological forms. Examples of highly inflected languages are Czech, Turkish or the Romance family of languages (French, Italian, Spanish, Portuguese). For instance, in Italian, "la bella casa" (the nice home) is singular, while "le belle case" (the nice homes) is plural. Notice how all three words change their inflection to reflect the change in number.

It is now clear that, if one of the characteristics of good natural language generation is being fluent, then the problem of morphological generation has to be addressed. In the framework of the Unboxer NLG pipeline (see Chapter 3) this task is solved by constructing a supervised model similar to the ones employed for predicting surface order (Section 5.1) and word lemmata (Section 6.6).

The prediction of word inflections given a lemma is then cast as a closed-class classification problem. At least for English, there is a finite set of inflections to choose from, and features such as the part-of-speech of the word and its context help predict the correct inflection.

6.7.1 Related Work

The prediction of morphological information is not a new task in the panorama of NLP and generation in particular. The standard NLG pipeline described in Chapter 2 inspired several NLG systems to follow a similar architecture, including the one proposed in this thesis. The *StuMaBa* NLG system [Bohnet et al., 2011a], for instance, is based on a pipeline architecture. One of the modules at the end of the generation pipeline is responsible for the generation of inflected forms starting from lemmas. This is carried out by computing edit scripts between lemmas and forms during training, and using the edit distance [Levenshtein, 1966] to score the candidate forms. For practical applications, the *SimpleNLG* software library Gatt and Reiter [2009] also includes a series of rules for the generation of inflectional morphology of English.

The other strand of work in this area is in the context of machine translation. Minkov et al. [2007], employ statistical methods to predict the inflections of words in the generation process in the framework of machine translation, showing how synctactic features improve the accuracy of the prediction. Toutanova et al. [2008], following the work of Minkov et al. [2007], successfully applied morphology prediction models to an actual machine translation task.

6.7.2 Generation of Inflectional Morphology

The English language has a fairly regular morphology. In English, usage of the base forms of words is common, and the inflections belong to a closed class of limited cardinality. With this consideration in mind, Minnen et al. [2001] have developed a system for morphological analysis and generation of English that operates on four possible word inflections: *-s* for plural of nouns and third person of verbs, *-ing* for the present participle, *-ed* for the past tense and *-en* for the past principle. These four cases alone cover most of the inflectional morphology of English, while the comparative and superlative form of adjectives, i.e., the suffixes -er and -est, are left out of the model. However, the choice of this particular model of morphology of English, rather than a richer one, does not interfere with the main point of this section of the thesis, that is, the prediction of word inflections can be treated as a classification task.

k_1	unary	$\neg$		
$\neg$	scope	k_2		
k_2	referent	e_1		
k_2	referent	x_1	1	**<A, - >**
k_2	event	pay		
k_2	concept	$customer$		
k_2	role	$agent$		
$customer$	instance	x_1	2	**<customer, - >**
pay	instance	e_1	4	**<pay, - >**
$agent$	internal	e_1	1	
$agent$	external	x_1		
k_2	surface	e_1	2	**<do, -ed>**
k_2	surface	e_1	3	**<not, - >**
k_2	surface	e_1	5	**<., - >**

Figure 6.9: Word-aligned DRG for "A customer did not pay." Instead of the words, in the last column there are <lemma, inflection> pairs.

The system created by Minnen et al. [2001] consists of two software packages. One package, called *morpha*, is the morphological analyzer and it has been described briefly in Section 4.3.2. It takes as input a word and optionally its POS tag and it outputs a lemma and its inflection. Since morpha is already part of the pipeline of tools used for the linguistic analysis of the documents in the GMB, with little modification it is possible to bring the morphological information into the DRG-surface alignment described in Chapter 3. In particular, the words in the alignment have to be replaced by a pair <lemma, inflection>. For instance, the example text-aligned DRG shown earlier in Figure 6.2 would now look like the one in Figure 6.9. Note the pair *<do, -ed>* corresponding to the word *did*. Here are a couple of examples of using morpha interactively on the command line:

```
$ echo 'John saw the rising sun' | ./morpha -au
john see+ed the rise+ing sun

$ echo 'Drinks are getting cheaper' | ./morpha -au
```

```
drink+s be+ get+ing cheaper
```

The other package included in the software distribution, *morphg* is the counterpart of morpha aimed at generation and it also works in a straightforward way by taking in input a lemma and an inflection and producing the word form. A POS tag can be optionally passed to morphg in order to increase its accuracy. For instance, passing the lemma *give*, (optionally) the POS tag *VBZ*, and the inflection *-ing*, morphg will output the word form *giving*. Here is an example of interactive usage of morphg:

```
$ echo 'John see+ed the rise+ing sun' | ./morphg -u
john saw the rising sun

$ echo 'Drink+s be+ get+ing cheaper' | ./morphg -u
drinks am getting cheaper

$ echo 'Drink+s_NNS be+_VBR get+ing_VBG cheaper_JJR' | ./morphg
drinks are getting cheaper
```

Note how the POS tag is necessary in some cases to correctly associate word forms to lemmata, otherwise the system does not have access to enough information to make a decision. For instance, in the second sentence above, morphg has to know that "drinks" is a plural noun and not the third person of the verb "to drink" in order to conjugate "to be" as "are".

By integrating morphg in the Unboxer pipeline the problem of generating inflectional morphology is reduced to a classification problem. The system associates to each entity (concepts and events) in the abstract meaning representation the inflection that characterizes the lemma that expresses such entity. The case where no inflection is needed is also included as one of the possible classification outcome. While another component is responsible of generating the correct lemma for each concept in the DRG (see sections 6.5 and 6.6), all that is left is predicting which form the final word will take.

6.7.3 Predicting Word Inflections: Pilot Study

Now that the task at hand is cast as a classification problem, input and output should be identified before proceeding. The input for the word inflection component is a concept in a DRG. Actually, the same set of features extracted to form the input for the lexical choice model presented in 6.6.1 can be reused here, as the input is the same in both tasks.

In addition to such features, two kind of features are added, that is, the cardinality of entities and the tense information related to events. These features are expected to be quite informative for the generation of concepts and events respectively, in particular in the *-s* cases for concepts — a concept with cardinality greater than one is usually realized by a plural noun, unless a mass noun is used.

The output of the statistical component is one out of the four classes contemplated by morpha/morphg plus the case where no inflection is present, that is, the set {s, ed, en, ing, ∅}, where the empty set symbol indicates the null inflection, i.e., that the word is generated in its base form.

Table 6.4: Confusion matrix of the prediction of morphological inflections for concepts.

class	en	ed	s	ing	∅
en	0	0	2	0	0
ed	0	0	22	0	297
s	0	0	1,065	0	474
ing	0	0	18	0	179
∅	0	0	552	0	11,971

Table 6.5: Confusion matrix of the prediction of morphological inflections for events.

class	en	ed	s	ing	∅
en	0	15	0	8	12
ed	0	559	0	42	298
s	0	173	0	14	178
ing	0	1	0	60	250
∅	0	376	0	18	8,633

A pilot study to assess the feasibility of this method has been carried out. A SVM-based classifier is trained on a dataset extracted from the same set of DRGs used for the experiment in Section 6.6 (252,214 concepts extracted from 9,161 DRGs out of the GMB). For each concept and each event in the dataset, the features extracted are the same as listed in Section 6.6.2, while the gold standard class is the inflection as produced by the morpha lemmatizer integrated in the GMB toolchain. A ten-fold cross-validation experiment resulted in the confusion matrices shown in Tables 6.4

and 6.5. 89.4% of the concepts in the experiment were classified correctly by the model, versus the 86.9% of the events.

Each row of the confusion matrices represents a class to be predicted and on the columns there are the classes that are actually predicted by the model. For example, in the experiment on concepts the inflection *-s* is predicted correctly 1,065 times, while 474 times the concept is misclassified as null inflection.

6.7.4 Discussion

The results that emerge from this pilot study, indicates that the model works, yet not in a satisfactory way. In particular, in many instances there is misclassification of null inflection cases and wrong classification of inflected forms as base forms.

Clearly the model misses some of the information needed to predict the right inflection in some cases. In the case of events, for instance, the English third person inflection -s is never predicted by the model. This may be due to the small number of examples in the training data, but also to the lack of certain features. The model is not able to decide, for instance, that the subject of a verb is a singular noun, and therefore the event is expressed by a third person verb, because it does not have access to the information about the cardinality of the concept that is the agent of the event. The implementation of new features capable of capturing this kind of information is left for future work, as is a comprehensive error analysis in order to find out exactly what additional information is missing in the model.

Regardless of the success and the issues of the simple supervised model for morphology prediction presented in this section, its main drawback is that it is strongly dependent on the language being English. A natural evolution would be that of generalizing the approach of treating the generation of inflections as a classification problem, so that it can be applied to other languages as well.

6.7.5 Alternatives

The five-class rigid classification model presented in this section is perhaps too simplistic to account for all the morphological variation of the English language, let alone that of other languages. Moreover, the method relies on an existing lexicon that maps lemmata to all their possible forms, including irregular ones such as *go+ed* = *went*. A valid alternative would be to switch to an open-class classification, where the set of possible predictions for a given concept is not restricted to the set {s, ed, en, ing, ∅} but actually covers all the possibilities given by the language.

Morphological variation can be modeled as sequences of editing operations on the base form of the target word. For instance, to go from the base form *drink* to the inflected form *drinks* the only operation needed is appending the letter *s* at the end of the word. For a slightly more complicated case, the editing operations to transform the base form *rise* into *rising* are, in sequence, a deletion (the letter *e*) and three additions (the letters *i*, *n* and *g*). With this scheme one can represent every possible morphological variation from the base form as a string of editing operations encoded in some unique way (*As* for the first example, *DAiAnAg* for the second example, and so on), including the irregular cases such as *see* becoming *saw* (*DDAaAw*). The advantages of this method are that it accounts for every possible combination of editing operations and it is independent from the target language and manually compiled lists of word forms. On the other hand, to implement such scheme in a supervised model for the prediction of inflections (like the one presented in this chapter) would require a larger dataset, as some of the instances to learn are rather sparse, in particular the cases of irregular morphology.

Finally, another alternative is to generate every possible inflection for every generated lemma and then employ a statistical language model to sort out the best combination by computing ngram frequencies of ngrams of inflected words. As noted in Section 6.2, this method is not really compatible with the architecture of the system proposed in this thesis, although it remains a useful way of evaluating the results of the system.

6.8 Discussion and Conclusion

In this chapter I introduced the problem of lexical choice in the framework of the Unboxer NLG pipeline presented in Chapter 3. The problem is tackled by dividing it into two separate tasks, namely lemma prediction and inflection generation (for the second task a pilot study has been conducted).

For the first task, I presented an unsupervised algorithm for lexical choice from WordNet synsets called Ksel that exploits the WordNet hierarchy of hypernyms to produce the most appropriate lemma for a given synset. Ksel performs better than a baseline based on the frequency of lemmata in an annotated corpus. A supervised alternative to Ksel is also tested in this chapter, although its performance is hindered by several factors.

The second task is treated as a classification problem, where the lemma and other features extracted from the semantic structure are fed to a supervised classifier to produce the word inflection out of a fixed set of choices.

The combination of the two solutions makes creates a lexical choice component that is able to generate the content words that express concepts in a DRG.

6.8.1 Error Analysis

At first glance, it is surprising that the unsupervised WordNet-based method performs well, considering that every possible synset of the candidate lemma takes part in the computation. Obviously, the aggregation functions play a big role in ruling out irrelevant senses from the picture (e.g., the third sense of *food* in the example in Section 6.5 has nothing to do with the semantic context), but seemingly something in the underlying structure of WordNet helps too. The intuition is that the intra-relatedness of different synsets associated with the same words is generally high, that is, different meanings of polysemous words are typically related to each other (with not-so-common exceptions).

WordNet synsets usually provide good quality synonyms for English lemmata. However, this is not always the case, for instance in some cases there are lemmata (or sequences of lemmata) that are not frequent in common language. As an example, the first synset of the English noun *month* is made of the two lemmata *month* and *calendar_month*. The latter occurs very seldom outside specific domains but Ksel produced it in 177 out of 181 cases in the experiment in Section 6.5.3. Cases like this result in awkward realizations such as "Authorities blame Azahari bin Husin for orchestrating last *calendar month*'s attacks in Bali." (example from the test set). Fortunately, only a very small number of synsets are affected by this phenomenon. Finally, it must be noted that Ksel is a totally unsupervised algorithm that requires only an external lexical knowledge base such as WordNet. This is not the case for other methods, including the Most Frequent Lemma baseline.

6.8.2 Future Work

The lexical choice module of the Unboxer pipeline is perhaps the most problematic, party because the difficulty of the problem itself, and partly because it is actually formed by two parts that cooperates towards the solution.

Regarding the unsupervised method for lexical choice presented in Section 6.5, being based purely on a lexical resource the Ksel approach lends itself nicely to be applied to different languages by leveraging multi-lingual resources like BabelNet [Navigli and Ponzetto, 2012], that have not been tried yet.

Both Ksel and its supervised alternative have only been tested on the generation of concepts, leaving the generation of events out of the picture. Although the syntax of DRGs allows for the same system to be easily adapted to events, it would be interesting to investigate the differences in the outcome.

Chrupala et al. [2008] have developed a statistical morphological analyzer that is independent from the specific language. *Morfette*, this is the name of the system, treats the analysis of morphological variation as a classification problem, where the classes are the possible sequences of edits at the character level that transform the base form into the inflected form. By reversing it, the Morfette system could be used to generate word inflections from lemmata, in a similar fashion to the model proposed in Section 6.7, but with the advantage of being language independent.

Chapter 7

Discussion and Conclusions

This is the final chapter of this thesis, and as such it is written with two main goals in mind: to look behind at the work presented so far, so to try and distill it into a meaningful conclusion, but also to look forward to new directions of research in NLG and related fields and to possible applications of the work presented in this thesis.

At the beginning of the thesis, in the introductory chapter, I wrote down four research questions to serve as a guide to understand the direction of the work presented in the thesis. Now those questions can finally have an answer motivated by the content of the previous chapters.

What logical formalism can represent the meaning of natural language expressions in a way that facilitates fine-grained alignment with the surface form? The Discourse Representation Graphs introduced in Chapter 3 are capable of representing the meaning of natural language with the same expressive power and flexibility of Discourse Representation Structures, from which they are derived. At the same time, they are a flat, tuple-based formalism, as opposed to the recursive DRSs, and as such they can be effectively aligned with the surface form at the word level.

How can we produce natural language from a logical form, provided that its alignment with the surface form is known? An aligned DRG contains all the information to reconstruct the surface form. The algorithm presented in Section 3.5.3 does that by producing the surface forms for individual discourse referents, and subsequently the complete surface form for the whole structure by composition.

Given an arbitrary logical form, what methods are the most effective at predicting the alignment with the surface form? The supervised statistical method introduced in Chapter 5 is capable of predicting the surface order encoded locally for each discourse referent by learning the alignment from a corpus of DRGs. The lexicalization can instead be solved by applying the unsupervised Ksel algorithm introduced in Chapter 6, and predicting the morphological inflections in a supervised fashion (see 6.7).

In the case of supervised statistical methods for natural language generation, resources like annotated corpora are needed. What characteristics should such a resource have, and how is it possible to build one that is rich enough to be employed to train statistical models? The Groningen Meaning Bank contains a large number of DRGs aligned with text, thus forming an appropriate base for supervised methods. Such a resource is built automatically by employing a pipeline of NLP analysis tools, and then corrected manually via crowdsourcing, as detailed in Chapter 4.

In the rest of this section, I present a few ideas and speculations about alternative approaches that have not been explored (Section 7.1). Next, in Section 7.2 I present a catalog of open problems in NLG in general and with respect to the approach of this thesis in particular. Finally, Section 7.3 presents some ideas for future work.

7.1 A Retrospective Look

The automatic generation of natural language, as many related problems, can be approached from two different directions. One is the study of the processes that bring human beings from the representation of information in their brain to the expression of such information by the use of a language suitable for communication with other humans. This way of representing knowledge, language, and the processes that transform and manipulate them, is typical of *Computational Linguistics*, where the focus is on construct plausible and effective models of language and communication, and an important byproduct (when not the main goal) is a deeper understanding of such processes.

The *Natural Language Processing* approach, on the contrary, shifts the attention on the development of techniques and resources to successfully achieve the task at hand. As a consequence, the solutions provided by NLP efforts are often harder to interpret from a linguistic perspective, even if they are well engineered and efficient ones.

Both approaches, and a mixture of the two, have been experimented with in the field of statistical NLG, as highlighted in Chapter 2, although the work found in literature seems to exhibit an historical trend going from the modeling approach, i.e., learning and applying a bidirectional grammar, towards more application-driven approaches.

Regarding the system proposed in this thesis, the Unboxer, it started out in a way that contains a mix of modeling and engineering elements. Specifically, the inspiration and motivation came originally from the study of the modeling process that had led to the creation of the Groningen Meaning Bank. A modification of the analysis pipeline that translates English texts into abstract meaning representation has made possible to create a large number of text aligned with the formal representation of their meaning, laying the foundation for the statistical NLG pipeline of the Unboxer system. From there on, the core of the system employs machine learning techniques to learn such alignment and generalize its representation in order to reconstruct it for a given unaligned abstract meaning representation. This is done by the two components described in Chapters 5 and 6.

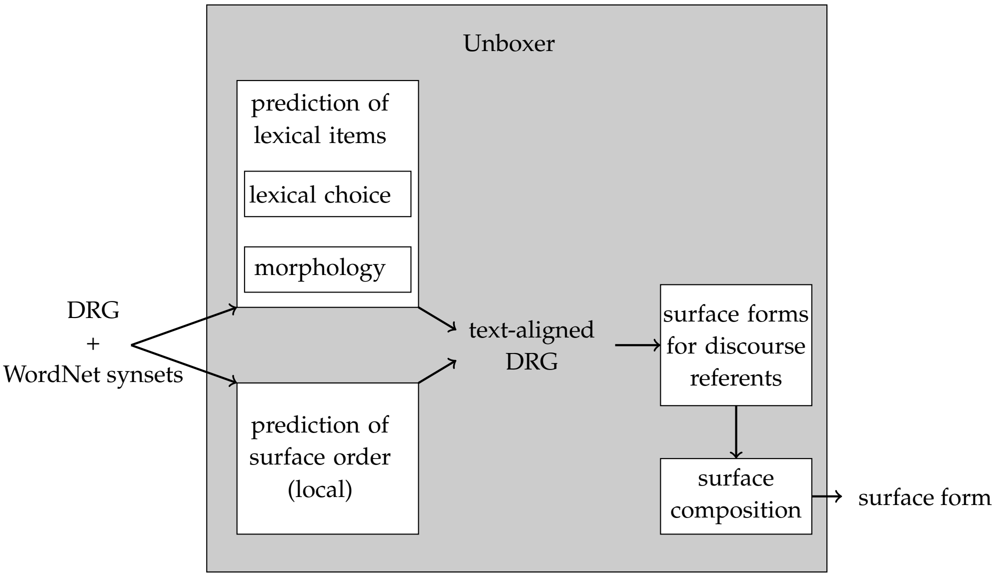

Unboxer
prediction of lexical items
lexical choice
morphology
DRG + WordNet synsets
text-aligned DRG
surface forms for discourse referents
prediction of surface order (local)
surface composition
surface form

Figure 7.1: A more detailed schematic view of the architecture of the Unboxer system.

Considering all the information presented throughout the central part of the thesis, after the idea of the Unboxer system is first introduced, an updated, richer pictorial representation of the architecture looks like Figure 7.1.

The modular structure of the Unboxer pipeline allows for a more efficient treatment of isolated tasks, but it comes at the cost of a increased difficulty in reading the results from a linguistic perspective. The experimental section of Chapter 5 is an example of such problem: it is not trivial to understand clearly what features of the alignment are more or less informative for the final surface order, or what information is missing.

Stepping into an hypothetical time machine and traveling back a few years, the development of the Unboxer could have followed a different route. Since the goal would have been to build a supervised system for generating natural language from logical forms, a resource like the GMB would have still been part of the design. The alignment of the formulas with the text, however, would be probably different, for instance experimenting with the global alignment strategy rather than the local alignment strategy that has been implemented, as discussed in Chapter 3. The rules that govern the alignment at the word level could also be better defined, and cover the entirety of the text, avoiding to leave out surface-specific words. With such an alignment in place, machine learning techniques would have been used to learn the alignment from scratch, without the need to learn different pieces of information at a time and then rebuilding the output in a separate step. This is not an easy task though, because of two main reasons. First, the structure of the input to the NLG pipeline is complex and sparse, that is, it is impossible to encode a whole DRG as a vector of features without hitting the data sparseness problem. Even if an algorithm was created to transform each possible DRG into a fixed-dimensionality vector, the number of feature required would be so large that each instance would be different from all the others and a learning algorithm could not infer any kind of generalized knowledge from a dataset like this. The other problem has again to do with the sparseness of data, but this time regarding the possible outputs, i.e., all the valid natural language expressions.

Perhaps these problems render the whole approach completely unfeasible, or perhaps it is only a matter of adapting and tweaking the appropriate representation formalism and learning machinery, for instance using modern techniques based on deep and unsupervised learning. Anyway, an approach like the one sketched here would learn a bidirectional semantic grammar that maps a language independent formal meaning representation to expressions in a natural language. Such result would be an important step towards general-purpose NLG and consequently semantically informed machine translation.

7.2 Known Issues and Challenges

An approach based on a bidirectional grammar would result in a more elegant solution to the NLG problem, although the question remains open whether such an approach is feasible at all. On the other hand, an engineering-driven solution like the one proposed by this thesis has an advantage among others, that is, it helps focusing on smaller subproblems during the development. This process has indeed highlighted some issues in several steps of the pipeline, that are discussed in this section.

This section addresses some of the issues that arise in the typical tasks in the field of statistical Natural Language Generation as they are formulated under the framework presented throughout the thesis. Traditionally, many models and components have been developed to solve separate sub-problems of the generation pipeline, often requiring different input formats and assumptions. Among such tasks are the problems of *generating referring expression*, *aggregation*, and the treatment of *rhetorical relations*, which are ofter dealt with by micro-planning components. Another problem one has to solve when dealing with supervised settings is the creation of a *gold standard* dataset, a task often too demanding in terms of time and resources. Finally, the *evaluation* of natural language generation output is known to be a hard task, similar to evaluating the quality of automatic translations, that proves to be particularly difficult to solve in an automatic way.

Part of the traditional NLG pipeline, namely the document-planning component, responsible for , among other things, structuring the output and connecting its parts via their discourse relations, is not taken care of by the Unboxer pipeline. Rather, the input to the Unboxer is assumed to be already available as a DRG, thus the issues related to discourse planning, content determination and document structuring are not discussed here.

7.2.1 Lack of a Gold Standard

Throughout the thesis, a problem has consistently shown up in several places, namely the need for good quality annotated data. In NLP, having a gold standard means the availability of a sufficiently large set of natural language segments paired with linguistic knowledge encoded by *human experts*. Manual annotation is so important for the quality of the data that a whole community of researchers is dedicated to develop new methods, interfaces and evaluation strategies, with their events and publications.

Likewise, a good deal of the work done for this thesis went into the creation of a gold standard semantically annotated corpus of English and methods to facilitate the creation of quality annotation. In fact, most of Chapter 4 is devoted at describing the process of creating the Groningen Meaning Bank, and then using the GMB Explorer and Wordrobe to provide gold standard annotation on many levels of linguistic analysis.

While both the Explorer and the Wordrobe game proved successful for their own goals, the GMB as a whole is far from being a gold standard resource. At the time of this writing, the GMB consists of 30,788 documents, 2.7 sentence long on average, for a total of 1,575,487 tokens. The single annotations collected from human experts are only 42,007, most of which are part-of-speech tags (10,789) and named entity tags (17,790). This means that only 0.68% of the POS tags of the GMB are manually corrected. While this number should not let down the reader, because the accuracy of the POS-tagger used to provide the automatic annotation is still very high, it is clear that the POS tag layer of the GMB annotation is not at a gold standard level, and similar conclusions can be drawn for the other layers. The situation is not better when it comes to the corrections extracted from Wordrobe. From the tens of thousands of answers (a number that is continuously growing) that the game collected in almost three years of activity, 6,982 single corrections were distilled.

Besides the "horizontal" problem of annotating a large number of tokens, the other problematic aspect of making a gold standard resource out of the GMB is the "vertical" nature of its parallel layers of linguistic annotation. Even if all the tokens in a text are checked by human experts with respect to one or more facet, this does not guarantee that the annotation is correct for every level. The shortest document in the GMB at the moment is the single nine-token sentence "Officials have warned opposition activists not to hold demonstrations". Nine tokens times ten layers of annotation (POS, lemmata, namex, animacy, senses, roles, relations, scope, reference, syntax) result in up to ninety tags to check for the expert linguist. Moreover, the chance of potential disagreement between experts increases with the size of the document and when looking into typically hard items to analyze such as word senses or named entity classes.

All these figures and considerations show that the GMB is a promising resource, and a useful one for many tasks, but there is still work to be done before being able to use it confidently for tasks that require large-scale gold standard datasets, such as statistical ones based on supervised machine learning techniques.

But why is it so important to have a gold standard-quality dataset? The obvious answer is that if one wants a computer system that learns from examples, such examples should be correct. But the reason why constructing a gold standard semantically annotated corpus is a delicate job lies in the skewed nature of the distributions of many natural language phenomena. Consider for instance the problem of word sense disambiguation, and conversely that of lexical choice as presented in Chapter 6. The popular baseline method for WSD consists in taking the most common sense for each word, and this method is known to work surprisingly well, due to most words having only one sense or a predominant sense that is the intended one most of the times. The most frequent lemma baseline against which the Ksel algorithm is tested behaved in a similar way, showing how most concepts have a predominant way of being expressed. As an exercise in manual annotation, I took the time to check one by one every noun and verb in thirty short documents from the GMB and confirm or correct their assigned WordNet synset. The process took several weeks and the quality cannot be confirmed to be perfect, both because there was just one annotator and because he was not a native speaker of the English language. As a result, most of the GMB is sense-annotated according to the most frequent sense baseline, which in turn means that a machine learning algorithm that tries to learn the surface forms corresponding to concepts in a DRG based on features of the semantic structure is more likely to fail, i.e., picking consistently the most frequent occurrence. Unfortunately this kind of issues are ubiquitous in a silver standard, automatically annotated dataset, sometimes undermining the chance of running reliable, meaningful experimental tests.

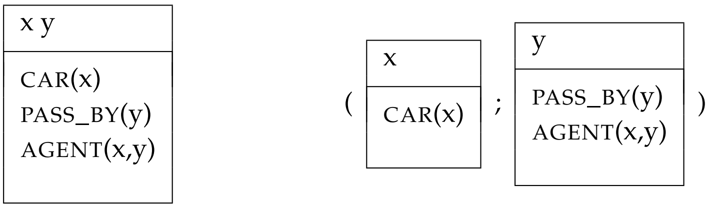


Figure 7.2: DRSs representing the meaning of the two sentences “A car passes by” (left) and “The car passes by” (right).

7.2.2 Generating Referring Expressions

In DRT, anaphoric expressions are resolved to a suitable antecedent discourse referent. Proper names and definite descriptions are too, but if finding a suitable antecedent fails then a process usually referred to as *presuppositional accommodation* introduces the semantic material of the anaphoric expression on an accessible level of DRS. The result of this process yields a DRS in which all presupposed information is explicitly distinguished from asserted information.

Consider for instance the two DRSs in Figure 7.2. Here the DRS on the left, representing the text “A car passes by” do not presuppose any material, since its entities (the car, in the example) are introduced for the first time in the sentence. The DRS on the right instead explicitly represents that CAR is part of the presupposed semantic material, reflecting the fact that in "The car passes by" the definite expression "the car" arguably refers to something that has been introduced elsewhere in the discourse (or it is to be found somewhere in the world knowledge of the hearer).

This gives rise to an interesting challenge for NLG, in particular when the task is that of generating a suitable expression for a given entity in an abstract meaning representation. A system informed by this kind of representation of presupposition might be better, for instance, at generating definite vs. indefinite determiners.

While in principle a system like the Unboxer has access to this kind of information, in practice it is not always easy to encode presupposition as local features of discourse referents, as seen for instance in the experimental section of Chapter 5. A possible direction for improvement the quality of the alignment is the new Projective DRT formalism, an extension to DRT that accounts for presupposition and other projective phenomena [Venhuizen et al., 2013b]. It is also interesting to explore the insights from approaches dedicated to generating referring expressions using logical methods van Deemter [2006], Gardent et al. [2004] with robust surface realization systems.

7.2.3 Aggregation

Coordinated noun phrases are known to be potentially ambiguous between distributive and collective interpretations. A simple DRT analysis for the distributive interpretation yields two possible ways to generate strings: one where the noun phrases are coordinated within one sentence, and one where the noun phrases involved are generated in separate sentences. For instance, the DRSs corresponding to "Deep Purple and Pink Floyd played at a charity show" (with a distributive interpretation) and "Deep Purple played at a charity show, and Pink Floyd played at a charity show", would be equivalent. This is due to copying semantic material in the compositional process of computing the meaning of the coordinated noun phrase "Deep Purple and Pink Floyd". The *collective* reading, as in "Deep Purple and Pink Floyd played together at a charity show" would not involve copying semantic material, and would result in a different DRS, with a different interpretation. It is the task of the aggregation process to pick one of these realizations, as discussed by White [2006]. Doing this from the level of DRS poses an interesting challenge, because one would need to recognize that such an aggregation choice is possible in the first place. Alternatively, instead of copying, one could use an explicit operator that signals a distributive reading of a plural noun phrase, for instance as suggested by

Kamp [1984]. Arguably, this is required anyway to adequately represent sentences such as “Both Deep Purple and Pink Floyd played at a charity show”.

7.2.4 Discourse Relations

In the classic NLG architecture, a document planning component is responsible for determining *what to say*, as opposed to the micro-planner which tells us *how to say it*. The document planner has to determine what is the topic of the generation and how to structure it in a way that is accessible to a human recipient. In (written) natural language, we decide what it is appropriate to say, then we subdivide it in documents, paragraphs, sentences and clauses. These divisions also have relations binding them together, for instance in “Max had a great evening last night. He ate salmon.” the second sentence is the *continuation* of the first one, while in “the population fled abroad because the volcano was erupting” the eruption event clause is the *cause* of the main event.

Asher and Lascarides [2003] nicely integrate relations at the discourse level into the DRT framework, and thanks to the reification mechanism shown in 3.3.2 they can be easily represented in a DRG. Even though the generation of discourse markers has been left out of the work presented in this thesis, while focusing on the generation of single sentences, an simple extension to the alignment strategy presented in 3.4 would allow a supervised system like the Unboxer to generate whole documents made of multiple clauses and sentences.

7.2.5 Evaluation of NLG

The evaluation of tasks related to the generation of language is typically harder than for tasks of analysis. Simply put, the output of a language analysis model is usually expected to be unique, unambiguous. For instance, to one English sentence should correspond exactly one syntactic parse tree, or at least only one should be considered the correct one. This is not the case when the direction of the analysis is reversed and one deals with a generation component because in general the output consists of a range of expressions, phrases, words, etc. some of which can be equally satisfying for the task at hand. The situation is analogous to the evaluation of automatic translations, where often times the “best” candidate for a translation (as in “most meaning preserving”) is not the one that sounds more natural or appealing to the hearer.

Traditionally, NLG systems are evaluated by means of experts *manually* checking the output and comparing it to some gold standard text. For some studies, researchers have resorted to *extrinsic* evaluation, where the output of a system is evaluated in the context of a larger system, or as the basis for another task. An example of the latter type of evaluation is the GIVE challenge [Byron et al., 2007] where instructions generated by the participant systems are given to human subjects with the goal of navigating a virtual 3D environment.

Recently, the interest has increased towards *automatic* methods of evaluation. These methods are typically based on some computable metric of "distance" between the candidate generations and a gold standard. However, the quality of the evaluation carried out with automatic measures is still not high enough to completely substitute manual checking [Belz and Reiter, 2006].

In the next section I sketch the idea for a new method of evaluating the output of natural language generation based on gamification and crowdsourcing.

7.3 Future Work

Any non-trivial research project is bound to leave certain options unexplored. Time and resources are limited, sometimes perhaps the technical means to develop a particular solution are not available at the time of the research. It may be nonetheless interesting for the reader to have some plans for future work sketched out, therefore the present section.

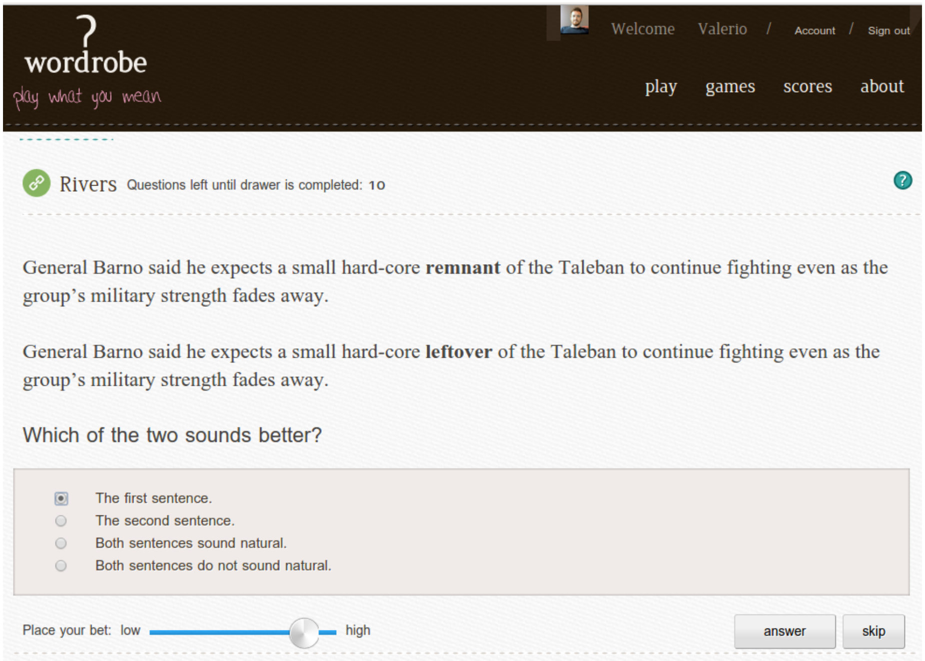
Throughout the thesis, usually at the end of each chapter, there is a section presenting plans for future work in the area of interest of the relevant chapter. In the rest of this section, instead, I present the ideas for work that is relevant to this thesis in a more global sense, or it represent an extension of it, but it has not been carried out yet.

7.3.1 Gamification for the Evaluation of Generated Sentences

In Chapter 4, I introduced Wordrobe, the collection of online games that was used (and still is, at the time of this writing) to collect gold standard annotation for the GMB. By design principle, the games of Wordrobe all share a common structure, that is, a fixed question, a variable short text with some optional highlighting of words, and a set of possible choices for the answer. The upside of this simple architecture is that creating a new game to be included in Wordrobe is a pretty straightforward process, consisting in the production of two tabular files, one for the question texts and another for the multiple choices.

The declared goal of Wordrobe is that of bettering the quality of the annotation of the GMB, therefore the linguistic material for the questions is generally taken from the GMB itself. This is not a hard costraint though, the two systems being virtually independent from each other. In fact, in one case — the game *Viittaukset*, a clone of *Pointers* for Finnish — the questions were generated separately, along with their possible choices, and subsequently integrated into a Wordrobe game.

With this premise, it is easy to see that the creation of a gamification-based approach to the evaluation of NLG is a feasible task. Here follows the plan for a game called *Rivers*, designed to leverage the Wordrobe crowdsourcing environment in order to collect human evaluation of the Unboxer output.


wordrobe
play what you mean
Welcome Valerio / Account / Sign out
play
games
scores
about
Rivers Questions left until drawer is completed: 10
Place your bet: low high
answer
skip

General Barno said he expects a small hard-core **remnant** of the Taleban to continue fighting even as the group's military strength fades away.

General Barno said he expects a small hard-core **leftover** of the Taleban to continue fighting even as the group's military strength fades away.

Which of the two sounds better?

● The first sentence.
○ The second sentence.
○ Both sentences sound natural.
○ Both sentences do not sound natural.

Figure 7.3: A mock-up example of what the Rivers game will look like.

The player of Rivers is shown pairs of sentences at a time. One of the sentences is taken from the GMB, while the second one is re-generated following the same process used for the experiments in 5.3. The question asked to the player is to give a judgment on the fluency of the text on screen, by saying which sentence sounds more natural. The list of possible choices is fixed:

1. The first sentence.
2. The second sentence.
3. Both sentences sound natural.
4. Both sentences do not sound natural.

A screenshot of the game as it is imagined here is provided in Figure 7.3. Note that in the example the words "leftover" and "remnant" are highlighted, being the example based on different lexicalizations of a concept (see Chapter 6. However, the game structure is applicable also to the evaluation of different aspects of the generation output, for instance the surface order.

With enough answers from different players, it is possible to measure the quality of the generated sentences for instance by computing the disagreement with respect to the two first options. The assumption is that a highly fluent generated sentence different on the surface from the gold standard would cause players to prefer it as much as the gold standard sentence. Other strategies and sets of possible choice can also be subject of experiments.

An approach to evaluation of NLG like the one proposed in this section would retain the desirable properties of traditional human judgment, including the graded kind of metrics capable of distinguish between a wrong result and a good but not perfect one. On the other hand, as proved by the other applications of Wordrobe, gamification techniques applied to NLP tasks can be powerful tools to speed up the collection of large amount of data.

7.3.2 Global Order Alignment

At the end of Section 3.5.1 it is hinted that there is an alternative way to align the text to the meaning representation, with respect to the information encoding the order of words an phrases. To recap, the strategy followed in the method of alignment proposed in this thesis exploits the network structure of the meaning representation and encodes the surface order *locally* to each discourse referent. This kind of alignment has been shown to be sufficiently informative to produce complete surface forms by means of the algorithm introduced in Section 3.5.3.

The alternative way of encoding order information in the alignment is to give a *global* order to the tuples composing the meaning representation based on the order

k_1	unary	$\neg$			
$\neg$	scope	k_2			
k_2	referent	e_1			
k_2	referent	x_1	**1**	**1**	A
k_2	event	PAY			
k_2	concept	CUSTOMER			
k_2	role	$agent$			
CUSTOMER	instance	x_1	**2**	**2**	customer
PAY	instance	e_1	**4**	**5**	pay
$agent$	internal	e_1	**1**		
$agent$	external	x_1			
k_2	surface	e_1	**2**	**3**	did
k_2	surface	e_1	**3**	**4**	not
k_2	surface	e_1	**5**	**6**	.

Figure 7.4: Word-aligned DRG for "A customer did not pay." Note the comparison between local ordering (fourth column) and global ordering (fifth column).

of the surface form. The difference between the two strategies of alignment is shown in the example in Figure 7.4.

While only the local order alignment strategy has been explored in this thesis, both strategies have their pros and cons. From an alignment based on local order a composition-based algorithm is capable of producing partial surface forms for the single discourse referents. In fact, this is the way the realization algorithm of the Unboxer constructs the final, complete surface form, that is, by building surface forms for each discourse referent and incrementally put them together. This behavior has desirable properties, e.g., for the purpose of error analysis, and it would be difficult to replicate it in the presence of an alignment using global ordering. Local order information also facilitates the supervised learning procedure, because it limits the size of the input to the problem, that is, the number of tuples to be ranked (see Chapter 5).

On the other hand, global order alignment is easier to produce, being based trivially on the order of the words in the surface form. While the local order information in the meaning representations of the GMB is produced by the linguistic analysis pipeline, the global order is inherent in the surface, thus it reduces the complexity of the procedure to create the alignment. Moreover, the realization algorithm would be trivial with an alignment based on global order.

Following these consideration, it is interesting to explore the alternative path of an NLG pipeline based on an alignment that follows a global order strategy. The main challenge would be the design of a new machine learning framework capable of learning such alignment. Nonetheless, once this problem is solved, a subsequent step is no longer necessary, potentially leading to an improved quality of the realizations.

7.3.3 Challenge: Generating Concepts from Logical Forms

A *shared task* is an open challenge to solve a specific problem on a given data set, often used as a mean of evaluating the state-of-the-art in a research area. Even though there have been various shared tasks on application-driven NLG tasks such as Question Generation [Rus et al., 2011], the GIVE challenge [Striegnitz et al., 2011] and specific NLG-related subtasks [Belz and Kow, 2010], the production of fluent text from abstract, deep semantic representations has not been fully explored. Since the sophistication and efficiency of syntactic and semantic parsers has increased considerably the last years [Butler and Yoshimoto, 2012, Bos, 2008, Zettlemoyer and Collins, 2012], it would be natural to be able to reverse this process, i.e., generating text from semantic representations. In this section, I propose a shared task aimed at the generation of natural language expressions for target concepts in a logical form. This task would be a step in the aforementioned direction, the inverse of semantic parsing.

This task has much in common with the Surface Realization task [Eugenio et al., 2012], for which an effort was made to agree on a standardized meaning representation format that could serve as the input of surface realization systems. However, finding a common ground in terms of input representation format for NLG has proven to be a challenge, thus the resulting input representation formats still contain traces of language-specific syntactic properties (based on dependency tree structures) and are designed with the goal of generation in mind, rather than being semantic representations with a model-theoretic interpretation. Rather than proposing new semantic representations to generate natural language from, the proposal is to take well-established formal semantic representations

The aim of this shared task is to generate text strings that accurately and fluently describe concepts expressed in a logical representation. More concretely, participants are given a logical form, one or more concept identifiers within this logical form, and are asked to generate their English surface string. A simplified example would be:

LF:	|x|=80,000
	BRITAIN-N-1(y)
	OF(x,y)
	TROOP-N-2(x)
Concept:	x
Output:	80,000 British troops

In this example, the logical form contains two variables referring to concepts: x and y. Here the task is to generate a description for one of them, namely x, where the gold standard description would be “80,000 British troops”. Other correct descriptions would be thinkable, such as “80K troops from Britain” (slightly different from the gold standard, but using more words). Informative but incomplete descriptions would be “80,000" troops”, or just “troops”. In contrast, “Britain” would be incorrect descriptions.

The syntax of logical forms includes operators for negation, disjunction and implication, and atomic formulas made of non-logical symbols with one (predicates) or two arguments (relations), formulas for named entities, cardinalities, and time expressions. There are links to Wordnet synsets (for symbols derived from nouns, adjectives, adverbs and verbs), thematic roles from VerbNet, preposition symbols (a closed class of English prepositions or possibly links to the Preposition Project database [Litkowski and Hargraves, 2005]).

Naturally, a concept can be expressed by various surface forms, but in the datasets for the shared task only one gold standard output is given. Also possible is to have the same logical form but ask to generate two or more concepts from it, as in the following couple of examples:

LF:	PRESIDENT-N-2(x)
	OF(x,y)
	NAMED(y, United_States)
	NAMED(x, Barack_Obama)
Concept:	x
Output:	President of the United States Barack Obama

LF:	PRESIDENT-N-2(x)
	OF(x,y)
	NAMED(y, United_States)
	NAMED(x, Barack_Obama)

Concept:	y
Output:	the United States

The examples above are just for illustration. The actual data will be formatted as DRGs (see Chapter 3 for a description of the formalism).

DRG	**variable**	**surface form**
101	x	80,000 British troops
101	y	Great Britain
102	x	President Barack Obama
103	x	capital of Helmand province
...	...	...

Figure 7.5: Format of the training data set

The training data for the task is a large set of triples consisting of a logical form of a sentence, the concept variable for which the surface has to be generated, and the gold standard surface form. The test data would be in a similar format but, obviously, without the expected surface form. The logical forms are derived from the Groningen Meaning Bank. A text file is provided to the participants containing, on each tab-separated line, the identifier of a DRG, the variable name of the concept to generate, and the expected output surface form An example of the format of the training data set is shown in Figure 7.5. The algorithm for surface realization of aligned DRGs presented in Section 3.5.3 is used to produce the gold standard, that is, for each marked concept a surface form is automatically produced, then checked manually for correctness.

By targeting the generation of natural language expression for concepts in logical forms, the focus of the shared task becomes that of exploring the common issues, abstracting away, as much as possible, from specific representations and languages. One problem that a candidate system should be able to solve is that of determining the order of the elements to be generated (see Chapter 5). Looking at the example above, “troops from Britain” is clearly different from “Britain of troops”. This is not simply a matter of applying an n-gram language model though, since some of the elements of the logical form could not generate surface structure at all, e.g., $of(x, y)$ within the surface form “80,000 British troops”. Other issues include the ones typical of the surface realization pipeline, such as the generation of correct word forms (gender, number, capitalization, see also Chapter 6) and the generation of connectives and other particles that are not explicit parts of the logical form. The information needed to produce the surface forms for the given concepts is contained, in theory, in the lexical resources linked to the symbols in the logical forms. However, the full logical form is given as input, so that systems could exploit contextual information in order to help the lexicalization process.

Logical forms are in principle language-neutral, but the lexical resources used in this task to link them to world knowledge are not. However, there exist many examples of cross-language alignment, e.g., for WordNet [Pianta et al., 2002]. By exploiting such resources, in the future the scope of the task could be extended to multi-language surface realization from logical forms.

7.4 Final Words

The automatic generation of natural language is a problem that has been on the radar of computer scientists almost since the beginning of the computing era. The research community then moved on from the utopia of a computer system capable of communicating with its human interlocutor using exclusively natural language. At the same time, many applications of Natural Language Generation have surfaced, and the field has partially evolved to accommodate particular applications, and to divide the problem into more manageable tasks.

This thesis reprises the original goal of NLG of being general, agnostic with respect to its application, and language-neutral. The system described in this thesis aims at generating full natural language expressions from formal structures based on logical formulas that represent meaning in an way that abstracts away from words and syntax. Indeed the emphasis of this thesis on the representation formalism is necessary, as it constitutes the motivation for all the choices and the engineering steps that have been taken to complete the project of the system.

Obviously, this is an ambitious project, and the experimental tests conducted on its several modules show that additional work needs to be done before the performance of the system reaches industry-level standards of quality. Yet, the hope is that the reader will grasp the message that the generation of natural language expressions from logical formulas is a job as feasible as it is important for the future of man and machine communication.

As a final note, I believe that Computational Linguistics should be treated with no less rigor than any empirical science. This entails that the models developed by the researchers should be accessible to the public and the experiments should be reproducible with as little extra effort as possible, as also pointed out in a recent study by Fokkens et al. [2013]. In this spirit, the software developed during the course of this study, the datasets, and the instructions on how to replicate the experiments are all publicly available at the URL `https://github.com/valeriobasile/unboxer`.

Bibliography

Kazimierz Adjukiewicz. Die syntaktische Konnexität. *Studia Philosophica*, 1:1–27, 1935. English translation “Syntactic Connexion" by H. Weber in McCall, S. (Ed.) *Polish Logic*, pp. 207–231, Oxford University Press, Oxford, 1967.

C. Akkaya, A. Conrad, J. Wiebe, and R. Mihalcea. Amazon mechanical turk for subjectivity word sense disambiguation. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 195–203. Association for Computational Linguistics, 2010.

Douglas E. Appelt. Bidirectional grammars and the design of natural language generation systems. In *Proceedings of the 1987 Workshop on Theoretical Issues in Natural Language Processing*, TINLAP '87, pages 206–212, Stroudsburg, PA, USA, 1987. Association for Computational Linguistics. doi: 10.3115/980304.980352. URL `http://dx.doi.org/10.3115/980304.980352`.

Guillaume Artignan, Mountaz Hascoët, and Mathieu Lafourcade. Multiscale visual analysis of lexical networks. In *13th International Conference on Information Visualisation*, pages 685–690, Barcelona, Spain, 2009.

N. Asher and A. Lascarides. *Logics of conversation*. Studies in natural language processing. Cambridge University Press, 2003. ISBN 9780521650588. URL `http://books.google.com.au/books?id=VD-8yisFhBwC`.

Miguel Ballesteros, Simon Mille, and Leo Wanner. Classifiers for data-driven deep sentence generation. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 108–112, Philadelphia, Pennsylvania, U.S.A., June

2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W14-4416.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W13-2322.

Satanjeev Banerjee and Ted Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 805–810, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1630659.1630775.

Srinivas Bangalore and Owen Rambow. Exploiting a probabilistic hierarchical model for generation. In *Proceedings of the 18th Conference on Computational Linguistics - Volume 1*, COLING '00, pages 42–48, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics. ISBN 1-55860-717-X. doi: 10.3115/990820.990827. URL http://dx.doi.org/10.3115/990820.990827.

Yehoshua Bar-Hillel. A quasi-arithmetical notation for syntactic description. *Language*, 29:47–58, 1953.

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. An enhanced lesk word sense disambiguation algorithm through a distributional semantic model. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 1591–1600, 2014. URL http://aclweb.org/anthology/C/C14/C14-1151.pdf.

Valerio Basile. A lesk-inspired unsupervised algorithm for lexical choice from wordnet synsets. *The First Italian Conference on Computational Linguistics CLiC-it 2014*, page 48, 2014.

Valerio Basile and Johan Bos. Towards generating text from discourse representation structures. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 145–150. Association for Computational Linguistics, 2011.

Valerio Basile and Johan Bos. Aligning formal meaning representations with surface strings for wide-coverage text generation. *ENLG 2013*, page 1, 2013a.

Valerio Basile and Johan Bos. Learning word order for surface realisation from logical forms. 2013b.

Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. Developing a large semantically annotated corpus. In *LREC*, Istanbul, 2012a.

Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. A platform for collaborative semantic annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 92–96, Avignon, France, 2012b.

J. Bateman. Sentence generation and systemic grammar: an introduction, 1997a. URL `http://www.fb10.uni-bremen.de/anglistik/langpro/kpml/Doc/lgsci-chap.pdf`.

John A. Bateman. Enabling technology for multilingual natural language generation: the kpml development environment. *Natural Language Engineering*, 3(1):15–55, 1997b. URL `http://dblp.uni-trier.de/db/journals/nle/nle3.html#Bateman97`.

Anja Belz and Eric Kow. The grec challenges 2010: overview and evaluation results. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 219–229, 2010.

Anja Belz and Ehud Reiter. Comparing automatic and human evaluation of nlg systems. In *In Proc. EACLâĂŹ06*, pages 313–320, 2006.

L. Bloomfield. *Language*. University of Chicago Press, 1933. ISBN 9780226060675. URL `http://books.google.nl/books?id=87BCDVsmFE4C`.

Bernd Bohnet. The induction and evaluation of word order rules using corpora based on the two concepts of topological models. 2007.

Bernd Bohnet, Simon Mille, Benoît Favre, and Leo Wanner. «stumaba»: From deep representation to surface. In *Proceedings of the 13th European Workshop on Natural Language Generation*, ENLG '11, pages 232–235, Stroudsburg, PA, USA, 2011a. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2187681.2187722`.

Bernd Bohnet, Simon Mille, and Leo Wanner. Statistical language generation from semantic structures. In *In Proceedings of the International Conference on Dependency Linguistics*, 2011b.

Dwight Bolinger. Entailment and the meaning of structures. *Glossa*, 2(2):119–127, 1968.

Kalina Bontcheva and Yorick Wilks. Automatic report generation from ontologies: The miakt approach. In *In Proceedings of the 9th International Conference on Applications of Natural Language to Information Systems*, pages 324–335, 2004.

Johan Bos. Implementing the binding and accommodation theory for anaphora resolution and presupposition projection. *COMPUTATIONAL LINGUISTICS*, 2003.

Johan Bos. Wide-Coverage Semantic Analysis with Boxer. In J. Bos and R. Delmonte, editors, *Semantics in Text Processing. STEP 2008 Conference Proceedings*, volume 1 of *Research in Computational Semantics*, pages 277–286. College Publications, 2008.

Nadjet Bouayad-Agha, Gerard Casamayor, and Leo Wanner. Content selection from an ontology-based knowledge base for the generation of football summaries. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 72–81. Association for Computational Linguistics, 2011.

Nadjet Bouayad-Agha, Gerard Casamayor, Simon Mille, Marco Rospocher, Horacio Saggion, Luciano Serafini, and Leo Wanner. From ontology to nl: Generation of multilingual user-oriented environmental reports. In Gosse Bouma, Ashwin Ittoo, Elisabeth Métais, and Hans Wortmann, editors, *Natural Language Processing and Information Systems*, volume 7337 of *Lecture Notes in Computer Science*, pages 216–221. Springer Berlin Heidelberg, 2012a. ISBN 978-3-642-31177-2. doi: 10.1007/978-3-642-31178-9_24. URL `http://dx.doi.org/10.1007/978-3-642-31178-9_24`.

Nadjet Bouayad-Agha, Gerard Casamayor, Simon Mille, Marco Rospocher, Horacio Saggion, Luciano Serafini, and Leo Wanner. From ontology to nl: Generation of multilingual user-oriented environmental reports. In *Natural Language Processing and Information Systems*, pages 216–221. Springer, 2012b.

Alexander Budanitsky and Graeme Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 89–96, New York, NY, USA, 2005. ACM. ISBN 1-59593-180-5. doi: 10.1145/1102351.1102363. URL `http://doi.acm.org/10.1145/1102351.1102363`.

Alastair Butler and Kei Yoshimoto. Banking meaning representations from treebanks. *Linguistic Issues in Language Technology - LiLT*, 7(1):1–22, 2012.

Donna Byron, Alexander Koller, Jon Oberlander, Laura Stoia, and Kristina Striegnitz*. Generating instructions in virtual environments (GIVE): A challenge and an evaluation testbed for NLG. In Michael White and Robert Dale, editors, *Proceedings of the Workshop on Shared Tasks and Comparative Evaluation in Natural Language Generation*, Arlington, VA, 2007.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 129–136, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: http://doi.acm.org/10.1145/1273496.1273513.

John Chamberlain, Massimo Poesio, and Udo Kruschwitz. Addressing the Resource Bottleneck to Create Large-Scale Annotated Texts. In Johan Bos and Rodolfo Delmonte, editors, *Semantics in Text Processing. STEP 2008 Conference Proceedings*, volume 1 of *Research in Computational Semantics*, pages 375–380. College Publications, 2008. URL `http://www.aclweb.org/anthology/W08-2230`.

Nancy Chinchor and Patty Robinson. MUC-7 Named Entity Task Definition, 1997. URL `http://www-nlpir.nist.gov/related_projects/muc/proceedings/ne_task.html`.

Noam Chomsky. *Syntactic Structures*. Mouton, The Hague, 1957.

Grzegorz Chrupala, Georgiana Dinu, and Josef van Genabith. Learning morphology with morfette. In Bente Maegaard Joseph Mariani Jan Odijk Stelios Piperidis Daniel Tapias Nicoletta Calzolari (Conference Chair), Khalid Choukri, editor, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may 2008. European Language Resources Association (ELRA). ISBN 2-9517408-4-0. http://www.lrec-conf.org/proceedings/lrec2008/.

Stephen Clark and James R. Curran. Parsing the WSJ using CCG and Log-Linear Models. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04)*, pages 104–111, Barcelona, Spain, 2004.

Ann Copestake, Dan Flickinger, Ivan Sag, and Carl Pollard. Minimal recursion semantics: An introduction. *Journal of Research on Language and Computation*, 3(2–3): 281–332, 2005.

Elizabeth Coppock and David Baxter. A translation from logic to english with dynamic semantics. In Kumiyo Nakakoji, Yohei Murakami, and Eric McCready, editors, *New Frontiers in Artificial Intelligence*, volume 6284 of *Lecture Notes in Computer Science*, pages 197–216. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-14887-3. doi: 10.1007/978-3-642-14888-0_18. URL `http://dx.doi.org/10.1007/978-3-642-14888-0_18`.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

James R. Curran and Stephen Clark. Language Independent NER using a Maximum Entropy Tagger. In *CONLL '03 Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, pages 164–167, 2003.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie Strassel, and Ralph M. Weischedel. The automatic content extraction (ace) program - tasks, data, and evaluation. In *LREC*. European Language Resources Association, 2004. URL `http://dblp.uni-trier.de/db/conf/lrec/lrec2004.html#DoddingtonMPRSW04`.

Rebecca Dridan and Stephan Oepen. Tokenization: Returning to a long solved problem. a survey, contrastive experiment, recommendations, and toolkit. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 378–382, 2012.

Pablo A. Duboue and Kathleen R. McKeown. Statistical acquisition of content selection rules for natural language generation. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, pages 121–128, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1119355.1119371. URL `http://dx.doi.org/10.3115/1119355.1119371`.

Michael Elhadad. Generating adjectives to express the speaker's argumentative intent. In *In Proceedings of the 9th Annual Conference on Artificial Intelligence. AAAI*, pages 98–104, 1991.

Jeffrey L. Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Jeffrey L. Elman. Distributed representations, simple recurrent networks, and grammatical structure. *Machine learning*, 7(2):195–225, 1991.

Barbara Di Eugenio, Susan McRoy, Albert Gatt, Anja Belz, Alexander Koller, and Kristina Striegnitz, editors. *INLG 2012 - Proceedings of the Seventh International*

Natural Language Generation Conference, 30 May 2012 - 1 June 2012, Starved Rock State Park, Utica, IL, USA, 2012. ISBN 978-1-937284-23-7.

Kilian Evang, Valerio Basile, Grzegorz Chrupała, and Johan Bos. Elephant: Sequence labeling for word and sentence segmentation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1426, 2013.

Roger Evans, Paul Piwek, and Lynne Cahill. What is nlg? In *IN PROCEEDINGS OF THE SECOND INTERNATIONAL CONFERENCE ON NATURAL LANGUAGE GENERATION*, pages 144–151, 2002.

Christiane Fellbaum, editor. *WordNet: an electronic lexical database*. MIT Press, 1998.

Antske Fokkens, Marieke van Erp, Marten Postma, Ted Pedersen, Piek Vossen, and Nuno Freire. Offspring from reproduction problems: What replication failure teaches us. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1691–1701, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P13-1166.

Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Alessandro Oltramari. Sweetening wordnet with dolce. *AI Mag.*, 24(3):13–24, September 2003a. ISSN 0738-4602. URL http://dl.acm.org/citation.cfm?id=958671.958673.

Aldo Gangemi, Roberto Navigli, and Paola Velardi. The ontowordnet project: extension and axiomatization of conceptual relations in wordnet. In *in WordNet, Meersman*, pages 3–7. Springer, 2003b.

Claire Gardent, Hélène Manuélian, Kristina Striegnitz, and Marilisa Amoia. Generating definite descriptions: Non-incrementality, inference and data. In Thomas Pechmann and Christopher Habel, editors, *Multidisciplinary approaches to language production*, pages 53–85. Walter de Gruyter, Berlin, 2004. URL http://hal.inria.fr/inria-00099806/en/.

Albert Gatt and Ehud Reiter. Simplenlg: A realisation engine for practical applications. In *Proceedings of the 12th European Workshop on Natural Language Generation*, ENLG '09, pages 90–93, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1610195.1610208.

Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowl. Acquis.*, 5(2):199–220, June 1993. ISSN 1042-8143. doi: 10.1006/knac.1993.1008. URL http://dx.doi.org/10.1006/knac.1993.1008.

Bikash Gyawali and Claire Gardent. Surface realisation from knowledge-bases. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 424–434, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P14-1040.

J. Hockenmaier and M. Steedman. CCGbank: a corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396, 2007.

Ian Horrocks and Ulrike Sattler. A description logic with transitive and inverse roles and role hierarchies. *Journal of logic and computation*, 9(3):385–410, 1999.

Eduard Hovy. Generating natural language under pragmatic constraints. *Journal of Pragmatics*, 11(6):689 – 719, 1987. ISSN 0378-2166. doi: http://dx.doi.org/10.1016/0378-2166(87)90109-3. URL http://www.sciencedirect.com/science/article/pii/0378216687901093.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. Ontonotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, NAACL-Short '06, pages 57–60, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1614049.1614064.

J.J. Jiang and D.W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of the Int'l. Conf. on Research in Computational Linguistics*, pages 19–33, 1997. URL http://www.cse.iitb.ac.in/~cs626-449/Papers/WordSimilarity/4.pdf.

Hongyan Jing. Usage of wordnet in natural language generation. In *Proceedings of the Joint 17th International Conference on Computational Linguistics 36th Annual Meeting of the Association for Computational Linguistics (COLING-ACL'98) workshop on âĂIJUsage of WordNet in Natural Language Processing SystemsâĂİ*, pages 128–134, 1998. URL http://www.aclweb.org/anthology/W98-0718.

T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA, 1999.

Hans Kamp. A Theory of Truth and Semantic Representation. In Jeroen Groenendijk, Theo M.V. Janssen, and Martin Stokhof, editors, *Truth, Interpretation* *and Information*, pages 1–41. FORIS, Dordrecht – Holland/Cinnaminson – U.S.A., 1984.

M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938. doi: 10.1093/biomet/30.1-2.81. URL `http://biomet.oxfordjournals.org/content/30/1-2/81.short`.

Tracy Holloway King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. The parc 700 dependency bank. In *In Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03*, pages 1–8, 2003.

Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. A large-scale classification of English verbs. *Language Resources and Evaluation*, 42(1):21–40, 2008.

Tibor Kiss and Jan Strunk. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525, 2006.

Daniël De Kok, Barbara Plank, and Gertjan Van Noord. Reversible stochastic attribute-value grammars.

Alexander Koller and Matthew Stone*. Sentence generation as planning. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 336–343, Prague, 2007.

Ioannis Konstas and Mirella Lapata. Concept-to-text generation via discriminative reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 369–378, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2390524.2390576`.

Ivana Korbayova, John Bateman, and Geert-Jan M. Kruijff. Generation of contextually appropriate word order. In Rodger Kibble and Kees van Deemter, editors, *Information Sharing: Reference and Presupposition in Language Generation and Interpretation*, volume 143 of *CSLI Lecture Notes*, pages 193–222. CSLI Publications, Stanford, 2002.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001. URL `citeseer.ist.psu.edu/lafferty01conditional.html`.

Irene Langkilde and Kevin Knight. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1*, COLING '98, pages 704–710, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics. doi: 10.3115/980451.980963. URL `http://dx.doi.org/10.3115/980451.980963`.

Thomas Lavergne, Olivier Cappé, and François Yvon. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 504–513. Association for Computational Linguistics, July 2010. URL `http://www.aclweb.org/anthology/P10-1052`.

Claudia Leacock and Martin Chodorow. Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2): 265–283, 1998.

Douglas B. Lenat. Cyc: A large-scale investment in knowledge infrastructure. *Commun. ACM*, 38(11):33–38, November 1995. ISSN 0001-0782. doi: 10.1145/219717.219745. URL `http://doi.acm.org/10.1145/219717.219745`.

Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, SIGDOC '86, pages 24–26, New York, NY, USA, 1986. ACM. ISBN 0-89791-224-1. doi: 10.1145/318723.318728. URL `http://doi.acm.org/10.1145/318723.318728`.

V. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics-Doklady*, 10(8):707–710, 1966.

Dekang Lin. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-556-8. URL `http://dl.acm.org/citation.cfm?id=645527.657297`.

Feiyu Lin and Kurt Sandkuhl. A survey of exploiting wordnet in ontology matching. In Max Bramer, editor, *Artificial Intelligence in Theory and Practice II*, volume 276 of *IFIP âĂŞ The International Federation for Information Processing*, pages 341–350. Springer US, 2008. ISBN 978-0-387-09694-0. doi: 10.1007/978-0-387-09695-7_33. URL `http://dx.doi.org/10.1007/978-0-387-09695-7_33`.

Kenneth C Litkowski and Orin Hargraves. The preposition project. *Proceedings of the Second ACL-SIGSEM Workshop on the Linguistic Dimensions of Prepositions and their Use in Computational Linguistics Formalisms and Applications*, pages 171–179, 2005.

Ling Liu and M. Tamer Özsu, editors. *Encyclopedia of Database Systems*. Springer US, 2009. ISBN 978-0-387-35544-3. doi: 10.1007/978-0-387-39940-9. URL `http://dx.doi.org/10.1007/978-0-387-39940-9`.

Francois Mairesse, Milica Gasic, Filip Jurcicek, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. Phrase-based statistical language generation using graphical models and active learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1552–1561, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P10-1157`.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *COMPUTATIONAL LINGUISTICS*, 19(2):313–330, 1993.

I. A Mel'cuk. *Dependency Syntax: Theory and Practice*. SUNY Press, Albany, 1988.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, 2010.

G. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38 (11):39–41, 1995.

Einat Minkov, Kristina Toutanova, and Hisami Suzuki. Generating complex morphology for machine translation. In *Proceedings of ACL*. Association for Computational Linguistics, June 2007. URL `http://research.microsoft.com/apps/pubs/default.aspx?id=69479`.

Guido Minnen, John Carroll, and Darren Pearce. Applied morphological processing of english. *Journal of Natural Language Engineering*, 7(3):207–223, 2001.

Reinhard Muskens. Combining Montague Semantics and Discourse Representation. *Linguistics and Philosophy*, 19:143–186, 1996.

Roberto Navigli and Simone Paolo Ponzetto. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193(0):217 – 250, 2012. ISSN 0004-3702. doi: 10.1016/j.artint.2012.07.001. URL `http://www.sciencedirect.com/science/article/pii/S0004370212000793`.

Roberto Navigli, Kenneth C. Litkowski, and Orin Hargraves. Semeval-2007 task 07: Coarse-grained english all-words task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 30–35, Stroudsburg, PA, USA,

2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1621474.1621480.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL http://dx.doi.org/10.3115/1073083.1073135.

Terence Parsons. *Events in the Semantics of English: A study in subatomic semantics*. MIT Press, Cambridge, MA, 1990.

JohnD. Phillips. Generation of text from logical formulae. *Machine Translation*, 8 (4):209–235, 1993. ISSN 0922-6567. doi: 10.1007/BF00981757. URL http://dx.doi.org/10.1007/BF00981757.

Emanuele Pianta, Luisa Bentivogli, and Christian Girardi. Developing an aligned multilingual database. In *Proc. 1st IntâĂŹl Conference on Global WordNet*, 2002.

R. Power. Controlling logical scope in text generation. In *Proceedings of the 7th European Workshop on Natural Language Generation*, Toulouse, France, 1999.

Judita Preiss, Jon Dehdari, Josh King, and Dennis Mehay. Refining the most frequent sense baseline. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, DEW '09, pages 10–18, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-31-2. URL http://dl.acm.org/citation.cfm?id=1621969.1621973.

Valentina Presutti, Francesco Draicchio, and Aldo Gangemi. Knowledge extraction based on discourse representation theory and linguistic frames. In *Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management*, EKAW'12, pages 114–129, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33875-5. doi: 10.1007/978-3-642-33876-2_12. URL http://dx.doi.org/10.1007/978-3-642-33876-2_12.

Ehud Reiter and Robert Dale. *Building Natural Language Generation Systems*. Cambridge University Press, New York, NY, USA, 2000. ISBN 0-521-62036-8.

Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *In Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, 1995.

Uwe Reyle. Dealing with ambiguities by underspecification: Construction, representation and deduction. *Journal of Semantics"*, 10:123–179, 1993.

A. Rumshisky, N. Botchan, S. Kushkuley, and J. Pustejovsky. Word sense inventories by non-experts. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, 2012. European Language Resources Association (ELRA).

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Cristian Moldovan. Question generation shared task and evaluation challenge – status report. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 318–320, Nancy, France, September 2011. URL `http://www.aclweb.org/anthology/W11-2853`.

S. Sekine, K. Sudo, and C. Nobata. Extended named entity hierarchy. In *Proceedings of LREC*, volume 2, 2002.

Stuart M. Shieber. A uniform architecture for parsing and generation. In *Proceedings of the 12th conference on Computational linguistics - Volume 2*, COLING '88, pages 614–619, Stroudsburg, PA, USA, 1988. Association for Computational Linguistics. ISBN 963 8431 56 3. doi: 10.3115/991719.991764. URL `http://dx.doi.org/10.3115/991719.991764`.

Stuart M. Shieber. The problem of logical-form equivalence. *Comput. Linguist.*, 19(1): 179–190, March 1993. ISSN 0891-2017. URL `http://dl.acm.org/citation.cfm?id=972450.972460`.

Manfred Stede. Lexical choice criteria in language generation. In *Proceedings of the Sixth Conference on European Chapter of the Association for Computational Linguistics*, EACL '93, pages 454–459, Stroudsburg, PA, USA, 1993. Association for Computational Linguistics. ISBN 90-5434-014-2. doi: 10.3115/976744.976799. URL `http://dx.doi.org/10.3115/976744.976799`.

Mark Steedman. *The Syntactic Process*. The MIT Press, 2001.

Kristina Striegnitz, Alexandre Denis, Andrew Gargett, Konstantina Garoufi, Alexander Koller, and Mariët Theune. Report on the second second challenge on generating instructions in virtual environments (give-2.5). In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 270–279, Nancy, France, September 2011. URL `http://www.aclweb.org/anthology/W11-2845`.

Lucien Tesnière. *Eléments de Syntaxe Structurale*. Klincksieck, Paris, 1959.

Juen tin Wang. On computational sentence generation from logical form. In *COLING*, pages 405–411, 1980. URL http://dblp.uni-trier.de/db/conf/coling/coling1980.html#Wang80.

Kristina Toutanova, Hisami Suzuki, and Achim Ruopp. Applying morphology generation models to machine translation. In *Proceedings of ACL-08: HLT*, pages 514–522, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P08/P08-1059.

Kees van Deemter. Generating referring expressions that involve gradable properties. *Computational Linguistics*, 32(2):195–222, 2006.

Kees van Deemter. What game theory can do for nlg: The case of vague language. In *Proceedings of the 12th European Workshop on Natural Language Generation*, ENLG '09, pages 154–161, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1610195.1610221.

Rob A. Van der Sandt. Presupposition Projection as Anaphora Resolution. *Journal of Semantics*, 9:333–377, 1992a.

Rob A. Van der Sandt. Presupposition projection as anaphora resolution. *Journal of Semantics*, 9:333–377, 1992b.

Noortje Venhuizen, Valerio Basile, Kilian Evang, and Johan Bos. Gamification for word sense labeling. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Short Papers*, pages 397–403, 2013a.

Noortje J. Venhuizen, Johan Bos, and Harm Brouwer. Parsimonious semantic representations with projection pointers. In Katrin Erk and Alexande Koller, editors, *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Long Papers*, pages 252–263, Potsdam, Germany, March 2013b. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W13-0122.

Leo Wanner and John A. Bateman. A collocational based approach to salience-sensitive lexical selection, 1990.

Warren Weaver. Translation. In William N. Locke and A. Donald Boothe, editors, *Machine Translation of Languages*, pages 15–23. MIT Press, Cambridge, MA, 1949/1955. Reprinted from a memorandum written by Weaver in 1949.

Michael White. Efficient realization of coordinate structures in combinatory categorial grammar. *Research on Language & Computation*, 4:39–75, 2006. ISSN 1570-7075.

Michael White, Rajakrishnan Rajkumar, and Scott Martin. Towards broad coverage surface realization with CCG. In *Proceedings of the Workshop on Using Corpora for NLG: Language Generation and Machine Translation (UCNLG+MT)*, 2007.

Ian H. Witten, Eibe Frank, Len Trigg, Mark Hall, Geoffrey Holmes, and Sally Jo Cunningham. Weka: Practical machine learning tools and techniques with java implementations, 1999.

Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics. doi: 10.3115/981732.981751. URL `http://dx.doi.org/10.3115/981732.981751`.

Annie Zaenen, Jean Carletta, Gregory Garretson, Joan Bresnan, Andrew Koontz-Garboden, Tatiana Nikitina, M Catherine O'Connor, and Tom Wasow. Animacy encoding in english: why and how. In *Proceedings of the 2004 ACL Workshop on Discourse Annotation*, pages 118–125. Association for Computational Linguistics, 2004.

Luke S Zettlemoyer and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *arXiv preprint arXiv:1207.1420*, 2012.

ChengXiang Zhai. Statistical language models for information retrieval a critical review. *Found. Trends Inf. Retr.*, 2(3):137–213, March 2008. ISSN 1554-0669. doi: 10.1561/1500000008. URL `http://dx.doi.org/10.1561/1500000008`.

Bowen Zhou, Yuqing Gao, Jeffrey S. Sorensen, Zijian Diao, and Michael Picheny. Statistical natural language generation for speech-to-speech machine translation systems. In *7th International Conference on Spoken Language Processing, ICSLP2002 - INTERSPEECH 2002, Denver, Colorado, USA, September 16-20, 2002*, 2002.

List of Acronyms

BOW Bit Of Wisdom.

CCG Combinatory Categorial Grammar.

DRG Discourse Representation Graph.

DRS Discourse Representation Structure.

DRT Discourse Representation Theory.

GMB Groningen Meaning Bank.

NLG Natural Language Generation.

NLP Natural Language Processing.

POS part-of-speech.

SVM Support Vector Machine.

WSD Word Sense Disambiguation.

English Summary

The central theme of this thesis is the generation of natural language from a formal representation of meaning. In a nutshell, the problem we want to solve is to go from a logical formula such as $\forall x(man(x) \rightarrow \exists y(woman(y) \wedge loves(x, y))$ to the sentence "every man loves a woman". This is achieved by employing several computer algorithms and statistical techniques. Moreover, not all representation formalisms are equal, some being better than others for the purpose of generation.

The first chapter of this thesis starts by presenting to the reader the problem of Natural Language Generation in a general way. This chapter introduces the formal representation of the meaning of natural language, in particular with formalisms based on logic, followed by the relationship between NLG and Machine Translation, and the motivation behind the approach to NLG undertaken in this thesis, that is, robustness and theoretical soundness. Finally, the research questions that drive the work presented in the rest of the thesis are formulated.

Chapter 2 contains a review of the literature in the field of Natural Language Generation, with particular focus on methods and problems relevant to several aspects of the work presented in the rest of this thesis. The chapter begins with presenting the traditional architectural organization of NLG tasks, then presents a review of previous work on statistical generation, generation from knowledge bases, and prediction of surface order, that is, the order of the words in the final output. The second part of the chapter shows how the representations of meaning found in the literature do not support well approaches to generation like the one proposed in this thesis. The chapter ends with a review of software packages for NLG.

Chapter 3 introduces the plan for the architecture of a novel system that generates natural language expressions given formal representations of meaning as input. Two of the modules, namely the surface order prediction module and the lexicalization module are covered in greater detail in the central chapters of the thesis. The surface realization module, positioned at the end of the pipeline, is also described in this chapter. It takes the output of the previous modules and produces the complete surface form that expresses the meaning encoded in the original meaning representation. This chapter also introduces a novel formalism for the representation of meaning, based on formal logic. The crucial feature of this formalism, called Discourse Representation Graphs, is that it favors the alignment between the abstract meaning representation and the text at the level of words. Thanks to this schema of alignment, several tasks of the NLG pipeline can be treated by supervised machine learning approaches.

Chapter 4 presents a semantically annotated resource called the Groningen Meaning Bank. This resource is used to train the models introduced in the previous chapter, as well as to extract data to test the proposed approaches through experimental trials. A number of design choices have been made for the creation and the annotation of the resource, and several software tools were employed to automatically analyze large quantity of text. Moreover, crowdsourcing methods were applied to gather linguistic annotations from the public, through a Web interface for experts and a Game With A Purpose called Wordrobe.

Chapter 5 covers the module of the system responsible for the prediction of the order of words and phrases composing the surface form. This problem is solved by leveraging a dataset of text-aligned meaning representations and building statistical models for learning to rank to predict the order of small sets of items local to each concept.

Chapter 6 presents the other central module, that is, the module responsible for the production of content words for the concepts contained in the original abstract meaning representation. This module actually solves two problems: the choice of the correct lemma from a closed set of options, based on the semantic content to convey, and the production of the correct morphological inflection. For the first task, two alternative methods are proposed: an unsupervised one and a supervised one, trained on GMB data. For the second task, a pilot study is presented in which the problem is solved by a supervised model of inflectional morphology of English.

The final chapter contains a series of reflections to conclude the thesis. A look *a posteriori* highlights the decisions that have been made for the design of the NLG system, thus inviting to speculate about alternative directions. Several problems are still open and it is important to consider how they affect the performance of the system. While these issues need to be addressed in order to obtain better results, the approach to NLG presented in this thesis is a step forward in improving existing approached to generation from logical forms.

Nederlandse Samenvatting

Het centrale thema van dit proefschrift is het genereren van natuurlijke taal vanuit een formele betekenisrepresentatie. Kort gezegd, willen we het volgende probleem oplossen: gegeven de logische formule $\forall x(man(x) \rightarrow \exists y(vrouw(y) \wedge liefhebben(x,y))$, moet de zin "alle mannen hebben een vrouw lief" worden afgeleid. Om dit te bereiken, worden verscheidene computeralgoritmes en statistische technieken toegepast. Naast de gebruikte methode, is ook het representationele formalisme van invloed op de kwaliteit van de gegenereerde zin; sommige formalismen zijn hiervoor meer geschikt dan andere.

Het eerste hoofdstuk van dit proefschrift start met een algemene beschrijving van het probleem van 'Natural Language Generation' (*Natuurlijke Taal Generatie*; NLG). Dit hoofdstuk introduceert formele representaties van de betekenis van natuurlijke taal, met een nadruk op logische formalismen, en de relatie tussen NLG en automatisch vertalen. Vervolgens wordt de motivatie achter de in dit proefschrift genomen aanpak tot NLG beschreven: het ontwikkelen van een robuuste, breed toepasbare en theoretisch onderbouwde NLG methode. Tenslotte worden in dit hoofdstuk de onderzoeksvragen die in dit proefschrift aan bod komen gepresenteerd.

Hoofdstuk 2 bevat een overzicht van de literatuur op het gebied van Natural Language Generation, met een focus op de methoden en problemen die relevant zijn met betrekking tot de aanpak tot NLG in de overige hoofdstukken van dit proefschrift. Het hoofdstuk begint met een beschrijving van de traditionele hiërarchische organisatie van NLG taken, en presenteert vervolgens een overzicht van de vakliteratuur over statistische generatie, generatie vanuit zogenaamde 'knowledge bases', en het voorspellen van woordvolgorde. In dit hoofdstuk wordt ook beschreven welke verschillende betekenisrepresentaties gebruikt zijn in eerdere NLG systemen. Aan het einde van het hoofdstuk wordt een overzicht van de verschillende bestaande softwarepakketten voor NLG gegeven.

Hoofdstuk 3 introduceert de architectuur van een nieuw systeem dat natuurlijke taal genereert vanuit formele betekenisrepresentaties. De twee belangrijkste modules van dit systeem, namelijk de module voor het voorspellen van de woordvolgorde en de lexicalisatiemodule, worden in Hoofdstukken 5 en 6 nader behandeld. De derde en laatste module van het systeem, welke op basis van de voorgaande modules de uiteindelijke expressie in natuurlijke taal levert, wordt in dit hoofdstuk beschreven. Daarnaast wordt in dit hoofdstuk een nieuw formalisme voor het representeren van betekenis geïntroduceerd, gebaseerd op formele logica. Het cruciale aspect van deze nieuwe representaties, genaamd Discourse Representation Graphs, is dat deze de coördinatie tussen de abstracte betekenisrepresentaties en de woorden van een tekst faciliteren. Dankzij dit nieuwe formalisme kunnen verschillende onderdelen van de NLG procedure uitgevoerd worden door middel van 'supervised machine learning'.

Hoofdstuk 4 presenteert een semantisch geannoteerde verzameling teksten, genaamd de Groningen Meaning Bank (GMB). Een dergelijk corpus kan gebruikt worden om de modellen die in het vorige hoofdstuk zijn gepresenteerd te trainen. Daarnaast kan de data worden gebruikt om de voorgestelde NLG methode experimenteel te testen. In dit hoofdstuk worden de ontwerpkeuzes met betrekking tot de ontwikkeling en annotatie van het corpus gemotiveerd en de softwareapplicaties beschreven die gebruikt zijn om de grote hoeveelheid teksten automatisch te analyseren. Om deze analyses te optimaliseren, verzamelen we taalkundige annotaties via verschillende bronnen, zoals een Web interface voor expert taalkundigen en een online taalkundig spel genaamd 'Wordrobe'.

Hoofdstuk 5 beschrijft de module van het NLG systeem dat verantwoordelijk is voor het voorspellen van de volgorde van de woorden waaruit de te genereren zin bestaat. Dit probleem wordt opgelost door het gebruik van een verzameling teksten geassocieerd met betekenisrepresentaties. Door middel van de statistische methode 'Learning to Rank' wordt van deze associaties geleerd om de volgorde van bepaalde woordgroepen te voorspellen.

Hoofdstuk 6 beschrijft de andere centrale module van het NLG systeem: de module die verantwoordelijk is voor het produceren van inhoudswoorden voor de concepten beschreven in de originele abstracte betekenisrepresentaties. De taak van deze module bestaat uit twee onderdelen: het selecteren van het juiste lemma gegeven een beperkte verzameling opties, gebaseerd op de betekenis in de huidige context, en het produceren van de juiste morfologische inflectie. Voor het eerste onderdeel worden twee alternatieve methoden voorgesteld: een 'unsupervised' en een 'supervised' methode, beide getraind op de data van de GMB. Voor het tweede onderdeel wordt een proefonderzoek gepresenteerd waarin het probleem wordt opgelost door middel van een 'supervised' model van inflectie in het Engels.

Het laatste hoofdstuk bestaat uit een reeks beschouwingen om het proefschrift af te ronden. Het eerste deel van het hoofdstuk bevat een terugblik op het werk dat in dit proefschrift is gepresenteerd, waarin de gemaakte keuzes worden geëvalueerd en wordt gespeculeerd over alternatieve benaderingen. In het derde en laatste deel van dit hoofdstuk worden een aantal openstaande problemen beschreven, en wordt nagegaan hoe deze de resultaten van het voorgestelde NLG systeem affecteren. Hoewel deze problemen in toekomstig werk geadresseerd dienen te worden om nog betere resultaten te verkrijgen, vormt de NLG methode die in dit proefschrift is gepresenteerd een belangrijke toevoeging op bestaande methoden voor het genereren van taal vanuit logische representaties.

Groningen Dissertations in Linguistics (GRODIL)

1. Henriëtte de Swart (1991). *Adverbs of Quantification: A Generalized Quantifier Approach.*
2. Eric Hoekstra (1991). *Licensing Conditions on Phrase Structure.*
3. Dicky Gilbers (1992). *Phonological Networks. A Theory of Segment Representation.*
4. Helen de Hoop (1992). *Case Configuration and Noun Phrase Interpretation.*
5. Gosse Bouma (1993). *Nonmonotonicity and Categorial Unification Grammar.*
6. Peter Blok (1993). *The Interpretation of Focus: an epistemic approach to pragmatics.*
7. Roelien Bastiaanse (1993). *Studies in Aphasia.*
8. Bert Bos (1993). *Rapid User Interface Development with the Script Language Gist.*
9. Wim Kosmeijer (1993). *Barriers and Licensing.*
10. Jan-Wouter Zwart (1993). *Dutch Syntax: A Minimalist Approach.*
11. Mark Kas (1993). *Essays on Boolean Functions and Negative Polarity.*
12. Ton van der Wouden (1994). *Negative Contexts.*
13. Joop Houtman (1994). *Coordination and Constituency: A Study in Categorial Grammar.*
14. Petra Hendriks (1995). *Comparatives and Categorial Grammar.*
15. Maarten de Wind (1995). *Inversion in French.*
16. Jelly Julia de Jong (1996). *The Case of Bound Pronouns in Peripheral Romance.*
17. Sjoukje van der Wal (1996). *Negative Polarity Items and Negation: Tandem Acquisition.*
18. Anastasia Giannakidou (1997). *The Landscape of Polarity Items.*
19. Karen Lattewitz (1997). *Adjacency in Dutch and German.*
20. Edith Kaan (1997). *Processing Subject-Object Ambiguities in Dutch.*
21. Henny Klein (1997). *Adverbs of Degree in Dutch.*
22. Leonie Bosveld-de Smet (1998). *On Mass and Plural Quantification: The Case of French 'des'/'du'-NPs.*

23. Rita Landeweerd (1998). *Discourse Semantics of Perspective and Temporal Structure.*
24. Mettina Veenstra (1998). *Formalizing the Minimalist Program.*
25. Roel Jonkers (1998). *Comprehension and Production of Verbs in Aphasic Speakers.*
26. Erik F. Tjong Kim Sang (1998). *Machine Learning of Phonotactics.*
27. Paulien Rijkhoek (1998). *On Degree Phrases and Result Clauses.*
28. Jan de Jong (1999). *Specific Language Impairment in Dutch: Inflectional Morphology and Argument Structure.*
29. Hae-Kyung Wee (1999). *Definite Focus.*
30. Eun-Hee Lee (2000). *Dynamic and Stative Information in Temporal Reasoning: Korean Tense and Aspect in Discourse.*
31. Ivilin Stoianov (2001). *Connectionist Lexical Processing.*
32. Klarien van der Linde (2001). *Sonority Substitutions.*
33. Monique Lamers (2001). *Sentence Processing: Using Syntactic, Semantic, and Thematic Information.*
34. Shalom Zuckerman (2001). *The Acquisition of "Optional" Movement.*
35. Rob Koeling (2001). *Dialogue-Based Disambiguation: Using Dialogue Status to Improve Speech Understanding.*
36. Esther Ruigendijk (2002). *Case Assignment in Agrammatism: a Cross-linguistic Study.*
37. Tony Mullen (2002). *An Investigation into Compositional Features and Feature Merging for Maximum Entropy-Based Parse Selection.*
38. Nanette Bienfait (2002). *Grammatica-onderwijs aan allochtone jongeren.*
39. Dirk-Bart den Ouden (2002). *Phonology in Aphasia: Syllables and Segments in Level-specific Deficits.*
40. Rienk Withaar (2002). *The Role of the Phonological Loop in Sentence Comprehension.*
41. Kim Sauter (2002). *Transfer and Access to Universal Grammar in Adult Second Language Acquisition.*
42. Laura Sabourin (2003). *Grammatical Gender and Second Language Processing: An ERP Study.*
43. Hein van Schie (2003). *Visual Semantics.*
44. Lilia Schürcks-Grozeva (2003). *Binding and Bulgarian.*
45. Stasinos Konstantopoulos (2003). *Using ILP to Learn Local Linguistic Structures.*
46. Wilbert Heeringa (2004). *Measuring Dialect Pronunciation Differences using Levenshtein Distance.*
47. Wouter Jansen (2004). *Laryngeal Contrast and Phonetic Voicing: A Laboratory Phonology Approach to English, Hungarian and Dutch.*
48. Judith Rispens (2004). *Syntactic and Phonological Processing in Developmental Dyslexia.*
49. Danielle Bougaïré (2004). *L'approche communicative des campagnes de sensibilisation en santé publique au Burkina Faso: les cas de la planification familiale, du sida et de l'excision.*
50. Tanja Gaustad (2004). *Linguistic Knowledge and Word Sense Disambiguation.*
51. Susanne Schoof (2004). *An HPSG Account of Nonfinite Verbal Complements in Latin.*
52. M. Begoña Villada Moirón (2005). *Data-driven identification of fixed expressions and their modifiability.*

53. Robbert Prins (2005). *Finite-State Pre-Processing for Natural Language Analysis.*
54. Leonoor van der Beek (2005). *Topics in Corpus-Based Dutch Syntax.*
55. Keiko Yoshioka (2005). *Linguistic and gestural introduction and tracking of referents in L1 and L2 discourse.*
56. Sible Andringa (2005). *Form-focused instruction and the development of second language proficiency.*
57. Joanneke Prenger (2005). *Taal telt! Een onderzoek naar de rol van taalvaardigheid en tekstbegrip in het realistisch wiskundeonderwijs.*
58. Neslihan Kansu-Yetkiner (2006). *Blood, Shame and Fear: Self-Presentation Strategies of Turkish Women's Talk about their Health and Sexuality.*
59. Mónika Z. Zempléni (2006). *Functional imaging of the hemispheric contribution to language processing.*
60. Maartje Schreuder (2006). *Prosodic Processes in Language and Music.*
61. Hidetoshi Shiraishi (2006). *Topics in Nivkh Phonology.*
62. Tamás Biró (2006). *Finding the Right Words: Implementing Optimality Theory with Simulated Annealing.*
63. Dieuwke de Goede (2006). *Verbs in Spoken Sentence Processing: Unraveling the Activation Pattern of the Matrix Verb.*
64. Eleonora Rossi (2007). *Clitic production in Italian agrammatism.*
65. Holger Hopp (2007). *Ultimate Attainment at the Interfaces in Second Language Acquisition: Grammar and Processing.*
66. Gerlof Bouma (2008). *Starting a Sentence in Dutch: A corpus study of subject- and object-fronting.*
67. Julia Klitsch (2008). *Open your eyes and listen carefully. Auditory and audiovisual speech perception and the McGurk effect in Dutch speakers with and without aphasia.*
68. Janneke ter Beek (2008). *Restructuring and Infinitival Complements in Dutch.*
69. Jori Mur (2008). *Off-line Answer Extraction for Question Answering.*
70. Lonneke van der Plas (2008). *Automatic Lexico-Semantic Acquisition for Question Answering.*
71. Arjen Versloot (2008). *Mechanisms of Language Change: Vowel reduction in 15th century West Frisian.*
72. Ismail Fahmi (2009). *Automatic term and Relation Extraction for Medical Question Answering System.*
73. Tuba Yarbay Duman (2009). *Turkish Agrammatic Aphasia: Word Order, Time Reference and Case.*
74. Maria Trofimova (2009). *Case Assignment by Prepositions in Russian Aphasia.*
75. Rasmus Steinkrauss (2009). *Frequency and Function in WH Question Acquisition. A Usage-Based Case Study of German L1 Acquisition.*
76. Marjolein Deunk (2009). *Discourse Practices in Preschool. Young Children's Participation in Everyday Classroom Activities.*
77. Sake Jager (2009). *Towards ICT-Integrated Language Learning: Developing an Implementation Framework in terms of Pedagogy, Technology and Environment.*

78. Francisco Dellatorre Borges (2010). *Parse Selection with Support Vector Machines.*
79. Geoffrey Andogah (2010). *Geographically Constrained Information Retrieval.*
80. Jacqueline van Kruiningen (2010). *Onderwijsontwerp als conversatie. Probleemoplossing in interprofessioneel overleg.*
81. Robert G. Shackleton (2010). *Quantitative Assessment of English-American Speech Relationships.*
82. Tim Van de Cruys (2010). *Mining for Meaning: The Extraction of Lexico-semantic Knowledge from Text.*
83. Therese Leinonen (2010). *An Acoustic Analysis of Vowel Pronunciation in Swedish Dialects.*
84. Erik-Jan Smits (2010). *Acquiring Quantification. How Children Use Semantics and Pragmatics to Constrain Meaning.*
85. Tal Caspi (2010). *A Dynamic Perspective on Second Language Development.*
86. Teodora Mehotcheva (2010). *After the fiesta is over. Foreign language attrition of Spanish in Dutch and German Erasmus Students.*
87. Xiaoyan Xu (2010). *English language attrition and retention in Chinese and Dutch university students.*
88. Jelena Prokić (2010). *Families and Resemblances.*
89. Radek Šimík (2011). *Modal existential wh-constructions.*
90. Katrien Colman (2011). *Behavioral and neuroimaging studies on language processing in Dutch speakers with Parkinson's disease.*
91. Siti Mina Tamah (2011). *A Study on Student Interaction in the Implementation of the Jigsaw Technique in Language Teaching.*
92. Aletta Kwant (2011). *Geraakt door prentenboeken. Effecten van het gebruik van prentenboeken op de sociaal-emotionele ontwikkeling van kleuters.*
93. Marlies Kluck (2011). *Sentence amalgamation.*
94. Anja Schüppert (2011). *Origin of asymmetry: Mutual intelligibility of spoken Danish and Swedish.*
95. Peter Nabende (2011). *Applying Dynamic Bayesian Networks in Transliteration Detection and Generation.*
96. Barbara Plank (2011). *Domain Adaptation for Parsing.*
97. Çağrı Çöltekin (2011). *Catching Words in a Stream of Speech: Computational simulations of segmenting transcribed child-directed speech.*
98. Dörte Hessler (2011). *Audiovisual Processing in Aphasic and Non-Brain-Damaged Listeners: The Whole is More than the Sum of its Parts.*
99. Herman Heringa (2012). *Appositional constructions.*
100. Diana Dimitrova (2012). *Neural Correlates of Prosody and Information Structure.*
101. Harwintha Anjarningsih (2012). *Time Reference in Standard Indonesian Agrammatic Aphasia.*
102. Myrte Gosen (2012). *Tracing learning in interaction. An analysis of shared reading of picture books at kindergarten.*
103. Martijn Wieling (2012). *A Quantitative Approach to Social and Geographical Dialect Variation.*

104. Gisi Cannizzaro (2012). *Early word order and animacy.*
105. Kostadin Cholakov (2012). *Lexical Acquisition for Computational Grammars. A Unified Model.*
106. Karin Beijering (2012). *Expressions of epistemic modality in Mainland Scandinavian. A study into the lexicalization-grammaticalization-pragmaticalization interface.*
107. Veerle Baaijen (2012). *The development of understanding through writing.*
108. Jacolien van Rij (2012). *Pronoun processing: Computational, behavioral, and psychophysiological studies in children and adults.*
109. Ankelien Schippers (2012). *Variation and change in Germanic long-distance dependencies.*
110. Hanneke Loerts (2012). *Uncommon gender: Eyes and brains, native and second language learners, & grammatical gender.*
111. Marjoleine Sloos (2013). *Frequency and phonological grammar: An integrated approach. Evidence from German, Indonesian, and Japanese.*
112. Aysa Arylova (2013). *Possession in the Russian clause. Towards dynamicity in syntax.*
113. Daniël de Kok (2013). *Reversible Stochastic Attribute-Value Grammars.*
114. Gideon Kotzé (2013). *Complementary approaches to tree alignment: Combining statistical and rule-based methods.*
115. Fridah Katushemererwe (2013). *Computational Morphology and Bantu Language Learning: an Implementation for Runyakitara.*
116. Ryan C. Taylor (2013). *Tracking Referents: Markedness, World Knowledge and Pronoun Resolution.*
117. Hana Smiskova-Gustafsson (2013). *Chunks in L2 Development: A Usage-Based Perspective.*
118. Milada Walková (2013). *The aspectual function of particles in phrasal verbs.*
119. Tom O. Abuom (2013). *Verb and Word Order Deficits in Swahili-English bilingual agrammatic speakers.*
120. Gülsen Yılmaz (2013). *Bilingual Language Development among the First Generation Turkish Immigrants in the Netherlands.*
121. Trevor Benjamin (2013). *Signaling Trouble: On the linguistic design of other-initiation of repair in English Conversation.*
122. Nguyen Hong Thi Phuong (2013). *A Dynamic Usage-based Approach to Second Language Teaching.*
123. Harm Brouwer (2014). *The Electrophysiology of Language Comprehension: A Neurocomputational Model.*
124. Kendall Decker (2014). *Orthography Development for Creole Languages.*
125. Laura S. Bos (2015). *The Brain, Verbs, and the Past: Neurolinguistic Studies on Time Reference.*
126. Rimke Groenewold (2015). *Direct and indirect speech in aphasia: Studies of spoken discourse production and comprehension.*
127. Huiping Chan (2015). *A Dynamic Approach to the Development of Lexicon and Syntax in a Second Language.*
128. James Griffiths (2015). *On appositives.*

129. Pavel Rudnev (2015). *Dependency and discourse-configurationality: A study of Avar.*
130. Kirsten Kolstrup (2015). *Opportunities to speak. A qualitative study of a second language in use.*
131. Güliz Güneş (2015). *Deriving Prosodic structures.*
132. Cornelia Lahmann (2015). *Beyond barriers. Complexity, accuracy, and fluency in long-term L2 speakers' speech.*
133. Sri Wachyunni (2015). *Scaffolding and Cooperative Learning: Effects on Reading Comprehension and Vocabulary Knowledge in English as a Foreign Language.*
134. Albert Walsweer (2015). *Ruimte voor leren. Een etnogafisch onderzoek naar het verloop van een interventie gericht op versterking van het taalgebruik in een knowledge building environment op kleine Friese basisscholen.*
135. Aleyda Lizeth Linares Calix (2015). *Raising Metacognitive Genre Awareness in L2 Academic Readers and Writers.*
136. Fathima Mufeeda Irshad (2015). *Second Language Development through the Lens of a Dynamic Usage-Based Approach.*
137. Oscar Strik (2015). *Modelling analogical change. A history of Swedish and Frisian verb inflection.*
138. He Sun (2015). *Predictors and stages of very young child EFL learners' English development in China.*
139. Marieke Haan (2015). *Mode Matters. Effects of survey modes on participation and answering behavior.*
140. Nienke Houtzagers (2015). *Bilingual advantages in middle-aged and elderly populations.*
141. Noortje Joost Venhuizen (2015). *Projection in Discourse: A data-driven formal semantic analysis.*

GRODIL
Secretary of the Department of General Linguistics
Postbus 716
9700 AS Groningen
The Netherlands