Event-Based Modelling in Question Answering

Diploma Thesis

LSV – Spoken Language Systems
Saarland University

Michael Wiegand

February 7, 2007

Statement

This is to certify that this thesis comprises only my original work except where indicated, and due acknowledgement has been made in the text to all other material use.

Saarbrücken, February 2007

Contents

List of Tables

List of Figures

Zusammenfassung in deutscher Sprache (Summary in German)

In der natürlichen Sprachverarbeitung haben Frage-Antwort-Systeme in der letzten Dekade stark an Bedeutung gewonnen. Vor allem durch robuste Werkzeuge wie statistische Syntax-Parser und Eigennamenerkenner ist es möglich geworden, linguistisch strukturierte Informationen aus unannotierten Textkorpora zu gewinnen. Zusätzlich werden durch die *Text REtrieval Conference (TREC)* jährlich Maßstäbe für allgemeine domänen-unabhängige Frage-Antwort-Szenarien def niert.

In der Regel funktionieren Frage-Antwort-Systeme nur gut, wenn sie robuste Verfahren für die unterschiedlichen Fragetypen, die in einer Fragemenge vorkommen, implementieren. Ein charakteristischer Fragetyp sind die sogenenannten *Ereignisfragen*. Obwohl *Ereignisse* schon seit Mitte des vorigen Jahrhunderts in der theoretischen Linguistik, vor allem in der Satzsemantik, Gegenstand intensiver Forschung sind, so blieben sie bislang im Bezug auf Frage-Antwort-Systeme weitgehend unerforscht. Deshalb widmet sich diese Diplomarbeit diesem Problem.

Ziel dieser Arbeit ist zum Einen eine Charakterisierung von Ereignisstruktur in Frage-Antwort Systemen, die unter Berücksichtigung der theoretischen Linguistik sowie einer Analyse der TREC 2005 Fragemenge entstehen soll. Zum Anderen soll ein Ereignis-basiertes Antwort-Extraktionsverfahren entworfen und implementiert werden, das sich auf den Ergebnissen dieser Analyse stützt. Informationen von diversen linguistischen Ebenen sollen daten-getrieben in einem uniformen Modell integriert werden. Spezielle linguistische Ressourcen, wie z.B. WordNet und Subkategorisierungslexika werden dabei eine zentrale Rolle einnehmen. Ferner soll eine Ereignisstruktur vorgestellt werden, die das Abpassen von Ereignissen unabhängig davon, ob sie von Vollverben oder Nominalisierungen evoziert werden, erlaubt.
Mit der Implementierung eines Ereignis-basierten Antwort-Extraktionsmoduls soll letztendlich auch die Frage beantwortet werden, ob eine explizite Ereignismodellierung die Performanz eines Frage-Antwort-Systems verbessern kann.

Chapter 1

Introduction

Within the last decade the task of Question Answering (QA) has become one of the most prominent research tasks in the area of Information Retrieval (IR) and Natural Language Processing (NLP). The need for processing large amounts of documents has arisen from the expansion and increasing popularity of the *World Wide Web* in the 1990s. Technical advances in hardware engineering provided new means of processing large amounts of data. With the development of robust NLP systems, such as part-of-speech (POS) taggers, named-entity (NE) taggers or statistical parsers, more ambitious tasks than the one of information retrieval (which is basically the retrieval of documents from a corpus on the basis of matching terms of a query with terms of the documents of a corpus) have been formulated. The need for a more f exible and user friendly interface for search-engines additionally promotes the development of such systems. One of the most diff cult task to date is QA which allows the user to formulate his/her query in natural language. Unlike conventional retrieval systems the output is not an entire document or passage but a text snippet which - in the ideal case - does not contain anything but the answer to a question posed.

The complexity of QA systems varies due to the extent of linguistic processing. The role of linguistic processing mainly distinguishes QA from IR. In QA, the query is not simply an unordered set of terms but a question formulated in natural language. On the one hand, this makes processing more diff cult due to the high ambiguity of natural language but, on the other hand, the query contains much more (structural) information, i.e. the syntax and semantics. One particular aspect of this additional information which combines both syntax and semantics is *event structure*. The inf uence of this aspect in QA, or more precisely answer extraction, will be explored in this thesis.

1.1 Motivation

In many conventional QA systems questions and answer sentences are represented by a set of terms, also known as *bag of words*. Such a representation originates from *information retrieval (IR)* which is mainly concerned with retrieving data, mostly documents, from a large data collection. In all QA systems which deal with unstructured or at most semi-structured data (such as the *world wide web*), such a retrieval task is embedded into the system. The transformation of the question into a query for the retrieval system and the answer extraction from the set of retrieved documents or passages are additional tasks. In simple systems the modelling of the question and candidate answer sentences remains term-based. Queries are constructed by converting questions to bag of words (usually by removing all functional words and stemming the remaining content words). As far as answer extraction is concerned, a common method is to match the terms appearing in a question with the terms in a candidate answer passage or sentence. A passage or sentence is deemed relevant if the ratio of matching terms is high. An answer is identified as a term situated in the vicinity of an area with a high density of matching terms which additionally conforms to some constraints, such as having an appropriate POS and/or NE tag[1]. This kind of answer extraction is illustrated by Figure 1.1. The advantage of this type is that this makes processing very efficient and a uniform representation is maintained throughout the pipeline of the QA system.
The power of such term-based models is, however, rather limited. A term-based representation certainly guarantees a reasonable *recall* but this often goes at the expense of the *precision* (Rijsbergen, 1979). Such an approach is likely to fail on the two Question-Answer Pairs[2] (1.1)-(1.2) and (1.3)-(1.4):

(1.1) Who was killed in the attack?

(1.2) The terrorists killed twenty three people who were working in the factory.

(1.3) Who has supported the new UN resolution?

(1.4) The British Prime Minister, who currently spends his holidays in Barbados on an invitation of veteran singer Sir Cliff Richard, has emphatically supported the new UN resolution.

In the first case, either the correct answer, i.e. *the twenty three people who were working in the factory*, or the *two terrorists* are returned as an answer. Without more structured knowledge concerning the *kill* event (for example: labelling the former entity as the *agent* and the second as the *patient*) a QA system cannot reasonably decide between those two candidates. In the second example, a standard system would favour *Sir Cliff Richard* to the *British Prime Minister* since the entity

[1] (Shen & Klakow, 2006), for example, use such a method, which they call *density-based answer extraction*, as a baseline to test their more advanced method against.

[2] Note that in this thesis, *question-answer pair* means a pair of question and answer sentence.

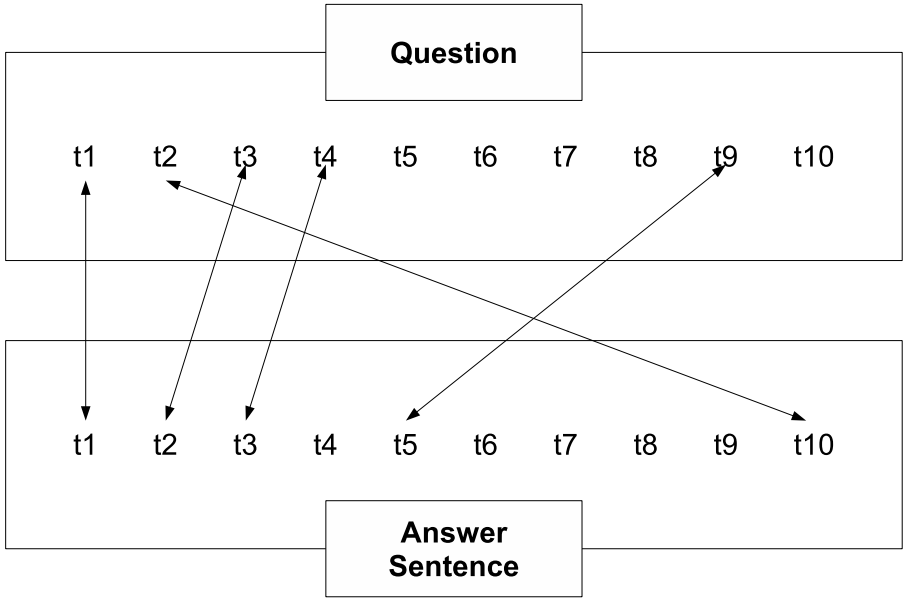

Figure 1.1: Term-based Comparison in QA.

is far closer to the terms of the question. The fact that the entity is deeply embedded in a relative clause and thus is not related to the *support* event cannot be modelled. Fortunately, the tasks of question analysis and answer extraction is different from the retrieval of information from large data. Due to the fact that those modules require less data to be processed the amount of processing can be increased. Thus, a more sophisticated form of linguistic processing should be attempted. Syntactical and semantic parsing should be used in order to represent questions and answer sentences. The question that arises is what linguistic unit should be chosen to represent them. Since many questions deal with *events* (a detailed def nition of that term will be given in the next chapter) this might be a suitable way of representation. It should be intuitive that a model which represents event structures as a group of entities which have a particular role in this event, be it expressing the spatial or temporal setting or other participating roles, such as agent, patient or theme, is a more appropriate way of representation than an unstructured set of terms. Event-based comparison in QA is illustrated by Figure 1.2[3]. Note that *ede* stands for *event denoting expression*[4] and *arg* for the *argument*.

[3]Some readers may have noted that the question contains two events. This is no misprint. It may be the case that short questions (like the majority of the TREC questions) only refer to a single event, but this does not have to be the case. The longer the questions become the more events the question may contain. This issue will be discussed further in the forthcoming chapters along some examples.

[4]This is a predicate evoking events.

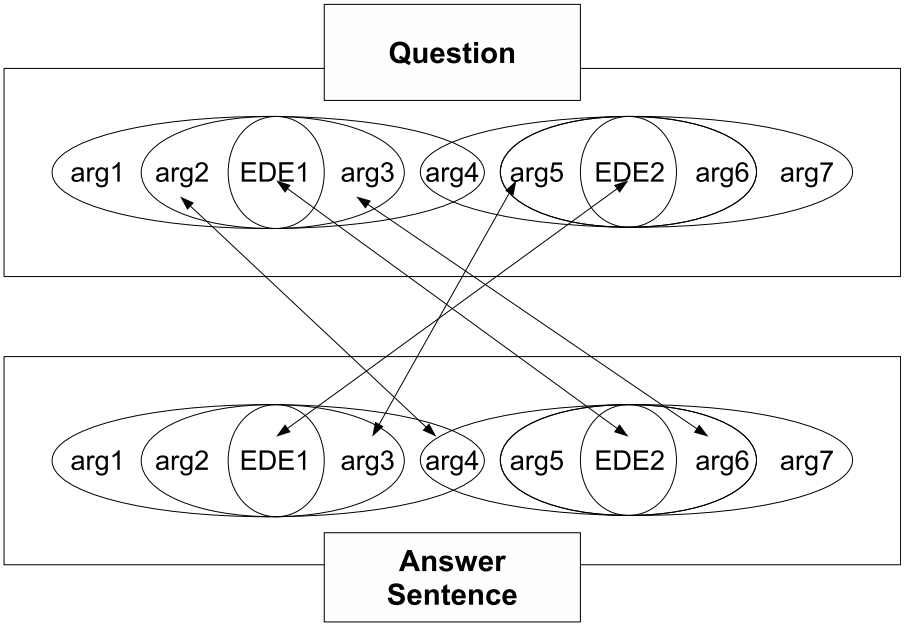

Figure 1.2: Event-based Comparison in QA.

1.2 Goals of this Thesis

The main goal of this diploma-thesis is to explore in how far *event structure* can contribute to better processing in QA systems. In order to do so I intend to implement an *answer extraction module* which is exclusively designed to tackle event questions.
Following questions have to be answered in the course of this thesis:

- What is an event?
- What does it comprise?
- What is its scope?
- How can events be modelled in QA?
- Can the performance of a QA system be improved by employing some form of event modelling?

Since there is no commonly accepted def nition of the term *event* one has to f nd a def nition at f rst which suits the context of open-domain QA best. Once there is a concrete notion of this term, one has to explore what methods this modelling requires. Concretely speaking, this means that one has to evaluate empirically potential tools for the module in advance, check whether they work as predicted

in this particular context and select the most appropriate ones. The design of the module should not merely be event-based but the model should also outperform other non-event-based answer extraction methods. Finally, the module which is to be developed should run in a reasonable time-span, i.e. the complexity of the module has to be adjusted to the practical needs of its usage.

1.3 Outline

This thesis will be structured as follows: Chapter 2 will first give a brief overview of the task of QA. Then, I will try to illustrate the competing concepts of events. Different QA systems which use some kind of event-based modelling will also be looked at. I will explain which particular notion is best suited for QA and how it can be used in theory to enhance the performance of a baseline QA system. Chapter 3 will discuss the insights gained by an analysis of the TREC 2005 question set, which is the set on which my module is going to be developed. Additionally, eligible tools that are available for a possible implementation will be described. The final design of the model will be explained in Chapter 4. In Chapter 5, I will carry out an evaluation of the module by testing its performance on both some artificial test set and the output of the retrieval component of an existing QA system in order to examine its viability in state-of-the-art applications. After a discussion of these results in Chapter 6, I will summarize my findings and also suggest possible directions of further research in Chapter 7.

Chapter 2

Background

This chapter tries to establish the foundations for the following chapters. I will start with giving a generic overview of QA. Then, I will discuss the different linguistic notions of *events*. After this, I will look at existing QA systems which perform event modelling or some similar form of processing. I will assess the concepts of linguistic theory and those to be found in practical systems with respect to their usability in an open-domain event-based answer extraction module. Finally, I will give a (preliminary) def nition of event structure which suits the task of answer extraction best.

2.1 A Brief Overview of QA

Giving an overview of QA is quite diff cult since there are different forms of QA tasks and that is why the corresponding architecture of such systems varies considerably. I will try to describe a fairly generic architecture, i.e. only those components of a QA system will be mentioned that are present in most types. This description will mainly follow (Hirschman & Gaizauskas, 2001).
The task in QA is to f nd out automatically whether an answer for a question is hidden in a data collection. This data collection may have different forms. It may consist of unstructured data, such as a corpus of newspaper articles, semi-structured data, such as the *World Wide Web*, or structured data, such as a database. In the following, I will focus on the f rst case since this thesis will only deal with this form of QA. It comprises following steps:

- **Question Analysis:** Once a question has been entered into the system it must be analysed. The aim of the *question analysis* is to convert the text into a structured query for the document retrieval component. Most systems also employ some *question typing* which map a question onto an element of a set of predef ned types. This question type determines how the retrieved information from the corpus has to be processed further in order to f nd an appropriate answer for the question. This component may become even more complex if the QA system is a component in a dialogue system. In this

case, the question should be analysed with the help of the context of the preceding dialogue and can be ref ned by asking the user if his/her question needs further clarif cation.

- **Document Collection Pre-processing:** In most situations, the corpus from which answers are to be extracted is too large to process exclusively on-line, i.e. when a question has been entered as a query. Some pre-processing is required. This often means that the corpus is converted to a representation which is more appropriate for fast data access. This conversion process is commonly referred to as *indexing*.

- **Candidate Document Selection:** A query is matched against an indexed representation of documents in order to retrieve a list of ranked candidate documents. The techniques applied in this step are IR methods.

- **Candidate Document Analysis:** This is an intermediate step in which the collection of retrieved documents is analysed further in order to restrict the set of potential answer documents. (The fewer documents are returned the more detailed answer extraction can be performed.) Sometimes, this even involves dividing the set of potential documents into passages.

- **Answer Extraction:** The potential answer documents or passages are further processed. Since this module covers a fairly small set of data, more complex processing, i.e. advanced NLP, is possible. This processing again re-ranks the list of retrieved documents or passages.

- **Answer Selection:** There are two ways how to obtain an answer from the data. The easiest way is to take the best retrieved document or passage and return the text snippet from these data that matches the criteria imposed by the question analysis most. The alternative is to embed this text snippet into an appropriate utterance. This task is also known as *Natural Language Generation (NLG)*. The generation of an answer (sentence) is particularly more appropriate in case of a dialogue system since the answer (sentence) can be tailored to the context in which the question was posed. (Note that NLG will not be part of this thesis.)

Event-based modelling is some form of linguistic processing. This restricts its application to only a subset of the QA modules. All those modules which process large amounts of data should only consist of eff cient IR methods. The exclusion of *candidate document selection* and *candidate document analysis* are therefore inevitable. On the other hand, typical modules which benef t from NLP are question analysis, answer extraction and answer selection. (Note that the latter two often appear as one step in literature. It is also called *answer extraction*. In order to be consistent with the majority of QA publications, I will stick to this convention.)
An important issue in QA are the resources in terms of data collections that are currently available. The most prominent collection is provided by *Text REtrieval*

Conference (TREC) (Voorhees & Harman, 2005) which is an ongoing series of workshops and competitions focusing on various IR research areas. The *Question Answering Track* takes place on an annual basis and provides both a set of questions and a text corpus. (Note that in this thesis for reasons of simplicity I always refer to the *Question Answering Track* when using the term *TREC*.) The corpus that is currently used in TREC is the *AQUAINT* corpus (Voorhees & Tice, 2000). The results of participating systems are evaluated manually. These evaluations are, however, later made available publicly and can thus be used for system development in subsequent years.

2.2 Concepts of Events in Linguistic Theory for QA

A general definition of *event* according to (Hornby, 1995) is

> a thing that happens, especially something important, an incident.

(Pustejovsky et al., 2003) describe it as

> a cover term for situations that occur. Events can be punctual or last for a period of time.

(Papka & Allan, 1998) call it

> something happening in a certain place at a certain time.

These are very broad definitions. For the current task a more linguistic notion is required. In linguistics, the notion of the term depends, however, on the particular branch of discipline one considers. It does not necessarily mean that these concepts are completely disjoint but at least they consider the term from a different point of view. There are two main areas in which this term plays a crucial role (which are also relevant for QA). These are aspectual classification[1] and sentence semantics.

2.2.1 The Scope of Events

Before I will discuss the different linguistic notions of *event* I should define what its *scope* is from a linguistic point of view. In some literature, like (Parsons, 1990), one assumes that sentences are the atomic units to denote events. This view, however, is a simplification. It is more the semantic counterpart of a sentence, i.e. the proposition[2], that is the atomic unit of an event. But not all propositions are realized as sentences. This is even not true if one regards syntactic clauses, such as relative clauses, adverbial clauses or verbal phrases as sentences. This is due

[1] According to (*Linguistics in SIL*, n.d.) *aspect* is a grammatical category associated with verbs that expresses a temporal view of the event or state expressed by the verb.

[2] According to (*Wikipedia - The Free Encyclopedia*, n.d.) propositions are assertions whose content might be taken as either true or false.

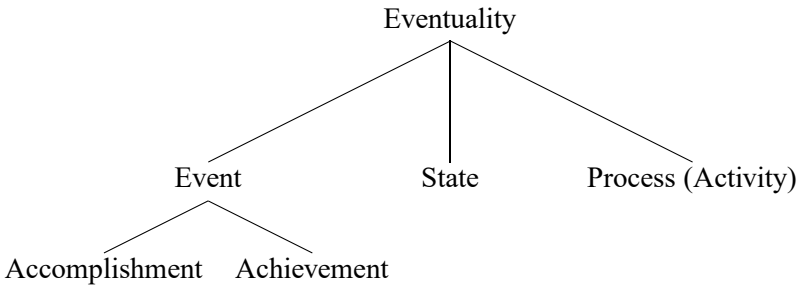

Figure 2.1: The Different Types of Aspects.

to the fact that there are other parts of speech than verbs which evoke propositions. In *TimeML* (Pustejovsky et al., 2003), for example, which is a specif cation language for event and temporal expressions in natural language text, linguistic expressions which evoke propositions can be verbs, nouns, adjectives, predicative clauses and prepositional phrases (PP). These different expressions have different semantic scopes. Consider Sentence (2.1) where the verb *arrive* is the linguistic expression evoking a proposition.

(2.1) [The Prime Minister arrived at the Party Conference]$_S$.

Its semantic scope is a sentence. One can, however, convert the verb to a noun *arrival* as in:

(2.2) [The arrival of the Prime Minister at the Party Conference]$_{NP}$ was expected for Wednesday afternoon.

The semantic scope of this expression is restricted to the noun phrase (NP) and not the entire sentence.
I will adopt the notion of the scope of events which corresponds to propositions. Note, however, that I must restrict the set of expressions evoking propositions to verbs and nouns since it is beyond the capability of state-of-the-art NLP tools to determine participating entities of propositions which are evoked by the other expressions.

2.2.2 Events and Aspect

In an aspectual typology of sentences the term *event* always occurs. This section will present the typology stated in (Bach, 1986) which is very similar to the other popular classif cation scheme of (Vendler, 1967). A discussion of an aspectual typology might shed light on what an *event* is. Figure 2.1 displays the different aspectual types (or *eventualities*) of sentences. The three main classes are *events*, *states* and *processes*. *Events* are def ned as a unique happening in the outside world whose temporal extension is f nite. *Accomplishments* are *events* that may or may not take an extended amount of time. Thus, it is meaningful to ask *how long* an

event of this type lasted. Furthermore, these events have mostly def nite culminations. A typical example is:

(2.3) Agatha made a sandwich.

Achievements, on the other hand, are instantaneous. That is why, it makes no sense to ask how long such a specif c event lasted. An example for this type of event is:

(2.4) Agatha won the race.

States differ from events in that they hold for varying amounts of time. It neither makes sense to ask *how long* a state lasted nor whether it culminated. A typical sentence which reports a state is:

(2.5) The dress is white.

The f nal type of sentence are *processes* (or sometimes referred to as *activities*). Like *events*, they are happenings. They differ from them in having no natural f nishing point. An example for this type is:

(2.6) Man ran.

Aspectual classif cation is a very complex task (which, in this thesis, should be rephrased as the task of distinguishing events from non-events) and has yet to be explored further. There already exist some computational models. The model presented in (Moens & Steedman, 1988) is a very sophisticated one which considers the interaction of the three main sources *lexical aspect*, *grammatical aspect* and *context*. From a theoretical point of view this model accurately accounts for various aspectual phenomena. It, however, relies heavily on world knowledge. Thus, an implementation for an open-domain application is almost impossible to realize. A more data-driven approach is described in (Siegel & McKeown, 2000). In this paper, aspectual classif cation is achieved by applying various machine learning methods. The mediocre performance of all methods applied suggests that a mere surface-based approach seems to be insuff cient for robust aspectual classif cation. Apart from the technical problems that one encounters when implementing aspectual classif cation, it is not even guaranteed that the notion of *aspectual events*[3] is appropriate for QA.

2.2.3 Events in Sentence Semantics

Since *events* are propositions it should be obvious that the underlying linguistic modelling is part of sentence semantics. Semantic modelling should be an integral part of the QA module to be implemented. The f nal design, however, will be essentially inf uenced by the capacity of the corresponding tools needed for this implementation.

[3]By this I mean those linguistic expressions which are classif ed as an event according to an aspectual typology like the one presented above.

As already mentioned, events can be seen as happenings in the outside world. The ultimate task in sentence semantics is to reconstruct the situation of the outside world, i.e. to characterize events on the basis of the information taken from linguistic expressions. (Basically, this just paraphrases the linguistic notion of *meaning*.)

Frame Semantics

Though there are, admittedly, many logics modelling events, I will only mention one type, namely *Frame Semantics (FS)* introduced in (Fillmore, 1968), since this form of representation seems to be the most appropriate for QA. (An explanation will follow below.) In the following, I will use the terminology of the *FrameNet* project (Fillmore, Johnson, & Petruck, 2003) which is a multilingual project to develop a frame-based lexical knowledge base. Semantic units in FrameNet are def ned according to FS. Unlike standard formalisms, such as *First Order Logic (FOL)*, the units to be modelled in FS are not lexically motivated but realized as specif c *frames*. According to (Fillmore, 1968) frames are def ned as

> units for the conceptual modelling of the world: structured schemata representing complex situations, events, and actions.

A frame is triggered by the so-called *frame evoking element (fee)* which is some kind of predicate. The participants, the so-called *frame elements (fes)*, are semantic arguments of the predicate, i.e. the *fee*. Sentence (2.7) can be represented in FS by something like Formula (2.8):

(2.7) Brutus stabbed Caesar in the back with a knife.

(2.8) $\exists f[TypeOf(f, CauseHarm) \wedge FEE(f, stabbing) \wedge Agent(f, B) \wedge Victim(f, C) \wedge BodyPart(f, b) \wedge Instrument(f, k)]$

It states that there is a frame f which is of type $CauseHarm$ and it is evoked by the *frame evoking element FEE* lexicalized by $stabbing$. There are four *fes*, namely Brutus B which has the role of the $Agent$, Caesar C which has the role of the $Victim$, the back b which has the role of the $BodyPart$ and the knife k which is the $Instrument$ within this frame.
Fes in FS are the uniform type of representation for complements and adjuncts[4]. There are two main benef ts in the concept of *fes*. Firstly, the entities are assigned an explicit semantic role[5]. (Fillmore, 1968) describes these roles as

> conceptual participants in a situation in a generic way, independent from their grammatical realization.

In FOL, on the other hand, Sentence (2.7) would be represented as Formula (2.9):

(2.9) $Stab(B, C, b, k)$

[4] These terms are explained in Appendix B.
[5] These roles were originally called *thematic roles* in (Fillmore, 1968).

Arguments are assigned to a predicate. Their meaning is only encoded by their position in a relation which is denoted by the literal they are part of. Thus, in FS, more information of the participants of a proposition are encoded which can be essential for further reasoning tasks.
Secondly, *fes* are represented in individual units, i.e. literals. By this representation one actually increases the capability of determining entailment relationships between different utterances. For example, one can show that Sentence (2.7) entails Sentence (2.10), since in FS (2.8) $\models$ (2.11).

(2.10) Brutus stabbed Caesar.

(2.11) $\exists f[TypeOf(f, CauseHarm) \wedge FEE(f, stabbing) \wedge Agent(f, B) \wedge Victim(f, C)]$

(2.12) $Stab(B, C)$

Unfortunately, one cannot establish this entailment relation in FOL, since for the underlying formulae of Sentences (2.7) and (2.10): (2.9) $\not\models$ (2.12).
The expressive power of FS that is provided by the abstraction from lexical and grammatical realizations can be illustrated with the following example. The situations in Sentences (2.13) and (2.15) are identical but only the linguistic representation and point of view are different. Fortunately, due to the labelling of frames and semantic roles the representations of the two sentences in FS are almost identical. Formulae (2.14) and (2.16) only differ in their respective *fees*, i.e. *gave* and *received*, but the remaining literals are identical.

(2.13) The teacher gave the student a book.

(2.14) $\exists f[TypeOf(f, Giving) \wedge FEE(f, give) \wedge Agent(f, t) \wedge Recipient(f, s) \wedge Theme(f, b)]$

(2.15) The student received a book from the teachter.

(2.16) $\exists f[TypeOf(f, Giving) \wedge FEE(f, receive) \wedge Agent(f, t) \wedge Recipient(f, s) \wedge Theme(f, b)]$

The frame labels and semantic roles denote situations in the outside world and, thus, are independent of the lexical units. For reasoning tasks, *fees* should be ignored. Thus, one could achieve a logical representation which states that both situations are actually the same.

Why Frame Semantics is the Optimal Form of Representation for QA

FS seems to be the most appropriate form of representation in QA. This is because it is a shallow form of representation but contains much more than just structural information obtained by syntactical analyses. Current state-of-the-art deep semantic parsers generating full FOL representations are inappropriate since their processing

is too ineff cient. Scope ambiguities and underspecif cations are problems which occur massively if a word by word translation of natural language into a logic language is attempted. But the complexity of current TREC questions (e.g. TREC 2005), which is a measure of how complex questions to be processed can be, do not require a representation of that sort.
From the perspective of shallow processing the semantic content of FS as provided by FrameNet is fairly detailed. Consider the difference of semantic information provided by a named-entity (NE) tagger. The set of different types of NEs in state-of-the-art taggers is very small, i.e. usually there are the types: *person*, *organization*, *location* and *time*. These types are not even related to the proposition. Thus, one fails to distinguish between an *agent* and a *patient* since both entities would be labelled as a *person*.
These insuff ciencies should all be rectif ed in a frame-based representation provided by FrameNet. Question-Answer Pair (2.17)-(2.18) should illustrate the usage of FS in QA:

(2.17) [How many students]$_{Victim}$ did [Kip Kinkel]$_{Killer}$ [**kill**]$_{Killing}$?

(2.18) [The **shooting**]$_{Killing}$ of [two students]$_{Victim}$ in a school cafeteria in Springf eld, Oregon caused a high media attention.

Note that I changed the notation of FS. Instead of a FOL-like representation, I bracketed the constituents of the sentences with *fe*- and *fee*-labels. The latter can be identif ed by their bold lexical units.
In Formula (2.17), the frame *Killing* is evoked by *kill*, *how many students* is the *fe Victim* and *Kip Kinkel* the *fe Killer*. Note that the question constituent *how many students* is treated as a normal *fe*. This treatment is vital since it is needed for matching candidate answers. The answer sentence has the following frame structure: *shooting* evokes the frame *Killing* and *two students* is the *fe Victim*. It should be obvious that one obtains the answer of the question by matching *how man students* and *two students* via their common *fe*-label in a common frame.
I deliberately chose a more complicated case since it illustrates the robustness of FS in this application. Firstly, the expressions evoking the propositions in the two sentences, i.e. *kill* and *shooting*, are both different lexical units and belong to different parts of speech, i.e. *kill* is a verb whereas *shooting* is a noun. Furthermore, the frame representing the question possesses one *fe* that the answer sentences does not contain. All these differences should not complicate the matching of the frame structures of question and answer sentence - at least not theoretically. The underlying frame structures are not identical but, due to the fact that they contain no contradictory information, they can be unif ed which is the pre-requisite for an appropriate matching of question and answer sentence.
Other ways of representing these sentences would, however, be problematic. The fact that one participating entity is missing in the answer sentence and the fact that the predicates are lexicalised differently would make a matching on the basis of FOL impossible.

Finally, I should comment on the relation between events and frames. As already mentioned earlier, frames cover virtually any kind of proposition, i.e. not only events. This is, however, only true if one interprets the term *event* in an aspectual context (see Section 2.2.2). Nowadays in (computational) linguistics, this term, or more precisely the term *event structure*, is also used to describe predicate-argument structures or similar semantic forms of representations, e.g. frame structures. For example, in TimeML (Pustejovsky et al., 2003), any predicate is labelled as an event[6]. This may be ascribed to one precursor of FS, the so-called *event semantics* (Davidson, 1967), which actually def ned a logic representation for *aspectual events*. Since then, the term event structure has also been used in other theories though they do not necessarily exclusively deal with aspectual events.
In QA, one could, therefore, say that event questions are those questions which can be answered on the grounds of matching event structures. For example, the f rst two of the following questions might be answered with the help of frame structures though only Question (2.19) deals with an *aspectual event*.

(2.19) Who killed John F. Kennedy?

(2.20) To which company does Youtube belong?

(2.21) Who is Al Gore?

The predicate *belong* in Question (2.20) rather describes a state than an event. Question (2.21) is problematic since this question does not evoke a frame, or more generally, the sentence does not contain a predicate, and therefore one cannot use event structures in order match this question with appropriate answer sentences. Section 4.2 will present how those event questions can automatically be recognized.

2.3 Event-based Modelling in Existing QA Systems

I now turn to existing QA systems and look if and how event-based modelling is designed. Thus, one can obtain a complementary view to the theoretical concepts. The type of event modelling that I will perform in QA should be faithful to the theoretic notions but practical issues will also have to be considered. A measure of what a good trade-off might be, could therefore be read off from the design of existing systems. Unfortunately, the term *event-based modelling* can hardly be found anywhere in topical publications. That is why, I have looked at systems which use syntactic and semantic processing, since they more or less model event structure (see also previous section).
(Sinha & Narayanan, 2005) address event-based QA and also answer extraction on the basis on FrameNet but this approach is designed for a closed-domain, namely *weapons of mass destruction (WMD)* scenarios. This allows to focus on reasoning

[6] or more precisely *event denoting expression (ede)*

on the basis of event ontologies which are domain specif c. The concrete algorithm to match questions and answer sentences is only described in a superf cial manner. So, there are hardly any insights of this paper that can be used for the current task.
(Saurí, Knippen, Verhagen, & Pustejovsky, 2005) present an event recognizer for QA. Though this is not an entire system this paper gives crucial insights into how an event in an open domain looks like. Some ideas correspond to the concepts presented in the preceding sections, such as *events* are lexical units evoking propositions. It is an algorithm looking for predicates rather than performing aspectual classif cation. *EVITA*, this is the name of the event recognizer, is a rule-based implementation which carefully analyses the context of possible events. Unfortunately, only examples of rules are displayed, so the publication does not offer suff cient information for a re-implementation. Neither is the prototype publicly available. Therefore, this publication can only be regarded as a support for the directions already formulated.
Some publications, such as (Buchholz, 2001), (Clarke et al., 2002), (Durme, Huang, Kupść, & Nyberg, 2003), and (Li & Croft, 2001) describe open-domain QA systems using grammatical relations which can be regarded as some elementary form of *event structure*[7]. With the notable exception of (Durme et al., 2003), these papers do not offer a uniform model to include those linguistic features. Instead heuristics, whose motivations are not always visible, are used for answer extraction. All of these implementations suffer from a low recall. Hard linguistic constraints seem to be too restrictive. (Shen, Kruijff, & Klakow, 2005) and (Shen & Klakow, 2006) address this problem. By modelling the matching of syntactic relations between questions and answers by means of data-driven models, syntactic criteria gain some robustness. However, the proposed method models events only implicitly since all possible syntactic relations are taken into consideration and thus no form of event structure is explicitly generated.
(Kaisser, 2006) presents a novel approach which makes use of FrameNet. This paper comes closest to the concept of *event structure* that is going to be followed in this thesis. Since, in this paper, the frame matching between questions and answer sentences is used as a hard constraint, the modelling remains very restrictive. However, the general usage of FrameNet in QA (as already presented in Section 2.2.3) seems to work. One can claim that even though this paper is only a preliminary report on the usage of this lexical resource (e.g. important components, such as word-sense disambiguation, are not yet integrated in that system).

2.4 A Practical Def nition of Event Structure in Question Answering

There now follows a def nition of events in open-domain QA. Most aspects have already been mentioned and explained in the previous sections of this chapter:

[7] This is due to the fact that the grammatical function of a particular entity is indicative of its semantic role.

Event structures are formalizations of propositions, such as frame structures in FS. Thus, events do not have to coincide with *aspectual* events (see Section 2.2.2). Every nominalization and full verb is regarded as an *event denoting expression (ede)*. In an answer extraction algorithm, event questions are those questions which successfully match answer sentences by means of their underlying event structures. For a successful matching on the basis of event structure an event question has to ask for an entity which can be identif ed as a participant of the event[8], such as the agent of the *kill* event in the following question:

(2.22) Who killed John F. Kennedy?

An appropriate model for the internal structure of events are frame structures from FS which model the external concept of an event (i.e. they abstract from the lexical units denoting the event structure in text).

[8] In case the question evokes more than one event, it suff ces that the question constituent represents an entity which participates in one of the events.

Chapter 3

Data and Tools

The last chapter explored events in liguistic theory and existing QA systems. On the basis of these insights consequences for the basic design of the implementation of an event-based QA module were drawn. Whether this approach can really be followed depends, however, on the data on which QA will be performed and the software tools that are available for the implementation. A thorough inspection of these issues is therefore imperative. The f nal method to guide the implementation should consider the results of this and the previous chapter.

3.1 The TREC 2005 Data Collection

The data on which my implementation is going to be developed will exclusively consist of the TREC 2005 question set (Voorhees & Harman, 2005) and the corresponding text corpus, i.e. the AQUAINT corpus (Voorhees & Tice, 2000). For this chapter only the question set is of interest since a question induces the QA process[1]. It determines what is to be looked for. Four aspects are of main interest:

- Does the frequency of event questions suff ce for an exclusive event-based answer extraction algorithm?
- What role does time play?
- What role do spatial coordinates play?
- How can other participants of events be characterized?

The following subsections deal with each of these items separately.

[1] Chapter 4.6 will look at the set of the relevant candidate answer sentences. This issue is not covered in this chapter since an evaluation on these data is only possible after acquiring labelled training data for my method which will all be described in the following chapter. In order to preserve a chronological order, I have, therefore, to postpone this assessment.

3.1.1 Event Structure in the TREC 2005 Question Set

Chapter 2.4 stated an operational def nition of event structure for my implementation. With this def nition of events, I annotated the question set of TREC 2005 in order to determine the relevance of event questions. I counted 200 event questions. In relative terms this amounts to 37.74%.
In order to recognize events in natural language text one needs to characterize the linguistic expressions that trigger events. I call such expressions *event denoting expressions (edes)*. Apart from full verbs there are other classes of words which are potential *edes*, such as nominalizations. They are the second most frequent linguistic realization of *edes*. In total, I counted 80 nominalizations in the 530 TREC 2005 questions (i.e 15.09% of the questions contain such a nominalization). This should justify the modelling of nominalized *edes*. All other *edes*, for example, adjectives, will not be considered as *edes* in this thesis[2].
This quantitative evaluation clearly supports the event-based modelling suggested in the previous chapter.

3.1.2 The Role of Time

This section looks at the role of time in the TREC 2005 question set. I will not exclusively restrict this analysis to the set of event questions as it might be interesting to see whether the distribution of the questions relating to time in the set of all questions differs from that in the set of event questions. It is a commonly accepted view that temporal information plays a crucial role for *aspectual* events. Whether this also holds for the events that are considered in this thesis will be investigated below.

Quantitative Analysis of the Different Types of Questions Involving Time

I begin with a typology of questions involving temporal information. The most obvious type of question asks for a specif c point in time, such as:

(3.1) When did the submarine sink?

In the following, I will refer to this type as *temporal question*. For current QA modelling, this type is highly important since approximately 12.45% of the questions in TREC 2005 are of this particular type. Each of them is also an event question. So, temporal questions are a reliable indicator of events.
A similar question type are *duration questions*, such as:

(3.2) How long was the debate scheduled to be?

They are fairly rare - only 0.38% of the questions are of this type. *Periodicity questions*, such as Question (3.3) are equally seldom:

[2] I encountered less than a hand-full of the other types in the TREC 2005 question set.

Question Type	Frequency	Percentage
Temporal Question	66	12.45
Duration Question	2	0.38
Periodicity Question	2	0.38
Temporal Relation Question	12	2.26
Question with Temporal Expression	18	3.39

Table 3.1: Statistics of Questions Involving Temporal Components.

(3.3) How often did Richard Nixon stand for president?

Both *periodicity questions* and *duration questions* also coincide exclusively with event questions.
Another type of question involve *temporal relations*, such as:

(3.4) How many people died when St. Helens erupted?

There are approximately 2.3% questions to be found in the set. This type differs from the above mentioned in that no explicit temporal expression is required in the question or the answer sentence. Unfortunately, the linguistic objects that are related temporally need not be *events*. Only around one fourth of the temporal relation questions are event questions. The rest are cases, such as:

(3.5) How old was Crosby when he died?

The matrix clause cannot be converted into an event structure due to the absence of a predicate.
The final question type describes questions which do not ask for a temporal expression but contain one, such as:

(3.6) What cruise line attempted to take over NCL in *December 1998*?

In these cases, the temporal expression should be regarded as a participant of an *event*. Ideally, the best answer sentence contains the identical temporal expression as the question. Unfortunately, only 3.39% of the questions contain such a temporal expression. If one only considers event questions this even decreases to 1.7%.
Table 3.1 summarizes the statistics for TREC questions and Table 3.2 for the subset of event questions. All in all, these results suggest that temporal modelling, in general, should be included into a QA system. I could also find evidence for the claim above mentioned that the inclusion of temporal modelling is, in particular, useful for event-based modelling QA, since many of the identified question types involving time co-occur exclusively with event questions.
Due to the limited time for the implementation I will only consider temporal questions and questions containing temporal expressions. These are the two most frequently occurring types which means that they should be given priority.

Question Type	Frequency	Percentage
Temporal Event Question	66	12.45
Duration Event Question	2	0.38
Periodicity Event Question	2	0.38
Temporal Relation Event Question	3	0.57
Event Question with Temporal Expression	9	1.7

Table 3.2: Statistics of Event Questions Involving Temporal Components.

Modelling Temporal Expressions

Now that temporal modelling in event-based QA could be justif ed, one needs to explore further what the appropriate design might look like. Two questions should be answered:

- How are temporal expressions represented?
- How are they located?

The f rst question is important for matching two temporal expressions. Hence, it is more relevant for questions with temporal expressions. The second question is equally relevant for both temporal questions and questions with temporal expressions.
The mere recognition of temporal expressions is fairly easy. State-of-the-art NE taggers perform this task fairly reliably. An appropriate representation of temporal expressions in QA, however, is very complex. (Passonneau, 1988) states that

> temporal information is distributed across several nonunivocal lexical and grammatical elements.

Temporal expressions might become problematic if they have to be semantically interpreted. The mere recognition does not specify a temporal expression further. A formal representation of these expressions, which allows a semantic comparison of these terms, is needed. Otherwise, expressions like *last Monday* and *7/8/2006* cannot be compared. In general, different levels of granularity of temporal expressions and the occurrence of anaphoric expressions require some form of normalization and anaphora resolution. Software tools like (*GUTime (Time Tagger)*, n.d.) are specif cally designed to solve these tasks. As far as TREC 2005 is concerned, however, the usage of this kind of processing is not really needed:
The problem of temporal anaphoras is not present. I inspected a random sample of 10 questions with temporal coordinates and checked the set of corresponding answer sentences due to (*TREC Answer Patterns*, 2005). I did not f nd any anaphoric temporal expression in this set.
The normalization of explicit temporal expressions is not that diff cult for the TREC questions as might be expected. The temporal expressions that are to be

compared with each other are almost exclusively year dates. Now, it might be possible that one encounters dates in question and answer sentence with different formats, such as *July 2006* and *7/8/2006*. A normalization of such expressions is straight forward since one only has to extract the year of these expressions. I even suspect that a common string matching algorithm, such as the *Levenshtein-distance* which is implemented in (*Sam's String Metrics*, n.d.), might suff ce for comparing temporal expressions since the great majority of question-answer pairs in TREC 2005 contains identical temporal expressions in question and answer sentence. (I conclude this from the same evaluation on a random sample of TREC questions and corresponding answer sentences where all the temporal expressions were identical.)

3.1.3 The Role of Space

Intuitively, spatial modelling for QA should be as important as temporal modelling. The similarities between these two types will become obvious when one looks at the different types of questions involving locations.

Quantitative Analysis of the Different Types of Questions Involving Locations

There are two types of questions involving locations (and both of these types will be modelled in my implementation), one being *locative questions*, i.e. questions which ask for a location, such as Question (3.7), and the other being questions which include a spatial coordinate which can be used as a reference point for matching candidate answer sentences, such as Question (3.8).

(3.7) Where was George Foreman born?

(3.8) When did the f rst McDonald's restaurant open $[$in the U.S.$]_{LOC}$?

The percentage of locative questions with $11,13\%$ is similar to that of temporal questions. However, there are far fewer event questions among those locative questions (approximately only 68%). The remaining locative questions (approximately 32%) are description questions, such as:

(3.9) Where is Port Arthur?

This result suggests that temporal questions are more indicative of event questions than locative questions. Related publications, such as, for example, (Crowe, 1995), conf rm this observation.
Approximately 8.67% of the TREC 2005 questions contain a location (being a named entity). 4.72% of the questions are event questions with a spatial coordinate. Table 3.3 summarizes the statistics presented above.

Question Type	Frequency	Percentage
Locative Question	59	11.31
Locative Event Question	40	7.35
Locative Description Question	19	3.58
Question with Spatial Coordinate	46	8.67
Event Question with Spatial Coordinate	25	4.72

Table 3.3: Statistics of Questions Involving Spatial Expressions.

Modelling Spatial Expressions

As far as locative questions are concerned, modelling should be fairly straight forward. The most important tool is a conventional named-entity (NE) tagger which supports the detection of locations. The situation is unlikely to occur that one has to choose between many locations nearby an event under investigation. This is due to the fact that locations do not occur that often. According to a state-of-the-art NE tagger[3], a location occurs every 20 sentences[4]. Of course, not every spatial coordinate is recognized since ordinary NE taggers only recognize locations being names of countries, cities or rivers. In order to increase the coverage one could use knowledge sources, such as WordNet, in order to recognize locations being common nouns, such as *house*, *school* or *hospital*. As far as questions involving spatial components in TREC 2005 are concerned, however, these types of locations are irrelevant, since locative questions ask for named entities and spatial coordinates that might both occur in question and answer sentence are also named entities.
Comparing spatial coordinates can be highly complicated. Spatial information behaves quite differently from temporal information in this respect. There are, admittedly, similarities as to the classif cations of anaphoric and non-anaphoric realizations, but there are two properties which are fundamentally different and these properties demand some different modelling of the two types of information.
Whereas temporal information can be described by a f nite grammar, spatial information cannot as easily be described in that *generative* way. An appropriate modelling of spatial information heavily relies on a large database. It can only be built manually, since it requires the world knowledge of an expert.
Due to the lack of a common structure in spatial coordinates, it is fairly diff cult to interpret two different locations. Temporal information can often be normalized to a format with suff cient granularity in order to determine the relation, i.e. similarity, between two dates, for example, *Saturday, 11/23/1963, 5:25pm* and *November 1963*[5]. But to state a relation between *Exeter* and *United Kingdom* it requires again

[3] I use the tagger from (Curran & Clark, 2003b).

[4] I computed this number by counting the number of spatial expressions in a set of documents annotated by a NE tagger and normalizing this value by the number of total sentences.

[5] Note, however, that sometimes, this does not work if one date is too unspecif c as in *Monday* and *September 1999*.

external knowledge, i.e. an ontology of locations which def nes *Exeter* as a city in the country *United Kingdom*. A fairly reasonable approach for QA could be the metrics proposed by (Makkonen, Ahonen-Myka, & Salmenkivi, 2003). Locations are hierarchically ordered by levels of geographic specif city, such as *continent*, *country*, *district/county* and *city*. The similarity of two locations can then be represented by their distance in the hierarchy.
Since this task is beyond the scope of this thesis and the TREC data do not suff - ciently provide spatial coordinates for event questions, though the amount of event questions which contain a spatial coordinate is more than twice the size of the event questions containing temporal coordinates, I will not compare spatial information in semantic terms (for example by means of a geographic ontology). Instead locations are compared by the orthographic similarity of the terms that denote them. Fortunately, as with temporal coordinates, spatial coordinates are also usually identical in questions and answer sentence. (At least, I did not encounter other cases in TREC 2005.)

3.1.4 Other Propositional Entities of Events

To regard space and time as the only propositional entities of events is inappropriate for answer extraction in event-based QA. The TREC 2005 questions also refer to the living beings and things (both physical and abstract) involved in events. Since events are closely connected to propositions (at least in this thesis I take this point of view), all arguments of propositions (in syntactic terms this corresponds to complements[6] and adjuncts of a verb or nominalization) should be regarded as participants[7]. From this perspective time and place are entities which are not so tightly related to events than other entities because they are mostly adjuncts. Only the fact that the proportion of these questions among the event questions in TREC 2005 is so great (more than half of the event questions either involve space or time) make these two participating entities so important.
How propositional entities of events, in general, will be recognized, interpreted and matched in my QA module - so far I have only discussed how temporal and spatial entities are to be treated - will be described in the forthcoming chapter in detail.

3.2 Existing Tools

After looking at the data to be processed I should also assess the availability and performance of NLP tools which could be used for the implementation of my answer extraction module. How these tools relate to each other will become obvious when the overall architecture of my module will be presented in the next chapter.

[6]Note that in this thesis I also subsume *subjects* by this term.
[7]For a more detailed explanation of these syntactic terms see Appendix B.

3.2.1 Stemmer

In any text retrieval task some form of stemming must be applied. The reason for this is that one needs to group all inf ectional forms of a lexical unit. The simplest stemming methods convert these forms into an abstract form (e.g. *emigrated* and *emigrates* are transformed into *emigrat$*). For my implementation a reduction to such abstract word forms is insuff cient as these stems cannot be looked up in lexicons, which I also use in my module. Therefore, I need some more complex processing which returns the lemma of the inf ectional form instead of an abstract stem (e.g. *emigrate* instead of *emigrat$*). I have decided to use Abney's stemmer (Abney, 1997) since it offers precisely this functionality.

3.2.2 Part-of-Speech Tagging

Part-of-speech (POS) tagging is needed for various reasons. Most NLP tools such as syntactic parsers or named-entity taggers demand POS-tagged text as input. I will use the *C & C* tagger (Curran & Clark, 2003a) which is a statistical tagger.

3.2.3 Named-Entity Recognition

The greatest problem with state-of-the-art named-entity (NE) taggers is that they are not capable to recognize f ne-grained classes. Recognition is restricted to personal names, locations, and temporal expressions. As already indicated in Sections 3.1.2 and 3.1.3, locations and temporal expressions are just recognized and not interpreted. My method has to work with this limitation. I will use the NE tagger included in the *C & C* software package (Curran & Clark, 2003b).

3.2.4 Pronoun Resolution

An important factor in QA is the way in which pronouns are treated. The omission of pronoun resolution would prevent entities in answer sentences, if referred to by a pronoun, which also occurred in a question (usually in the form of a named entity) from being matched. The only tool that was available to me is a module of *Alyssa* (Shen, Leidner, Merkel, & Klakow, 2006), which is the QA system developed at LSV for TREC 2006. It uses components provided by GATE (Cunningham, Maynard, Tablan, Ursu, & Bontcheva, 2001).

3.2.5 Syntactic Lexicons

If one needs access to subcategorization frames of the lexical units triggering events, one needs a lexicon which enlists all possible frames of a particular verb or nominalization. One could argue that subcategorization frames could also be directly read from parses but in that case one would not be able to distinguish between complements from adjuncts[8]. One possible usage of the information of

[8] The terms *complement* and *adjunct* are explained in Appendix B.

subcategorization lexicons is to check whether the argument status of an entity in a question and its potential counterpart in an answer sentence are the same. If their status is identical, then this supports the view that the match of the two entities is positive. (I will later fully explain why I need these frames. For the moment, one should keep in mind that these frames can be seen as a starting point of event-structures.)
As far as verbs are concerned, I made use of *COMLEX Version 3.0* (Macleod, Grishman, & Meyers, 1998). The lexicon contains approximately 6000 verb entries[9]. At the time of implementing my model no other lexicon was available to me[10]. Nevertheless, the lexicon contains entries for all verbs found in the TREC 2005 question set.
Since nominalizations also play an important part in event-based modelling (see Section 3.1.1), a lexicon for these types of nouns is also needed. This lexicon, however, has to provide more information than a lexicon for verbs. These are:

(3.10) specif cation of the original verb of the nominalization

(3.11) list of subcategorization frames

(3.12) assignment of grammatical functions to complements

(3.10) is essential for QA if one needs to map nominalizations onto verbs. (Since the occurrence of nominalizations is signif cantly lower than the occurrence of verbs, it is often the case that nominalizations will only appear in either question or answer sentence. The other sentence contains the corresponding verb.) These mappings are required for measuring semantic similarities via WordNet (see also the upcoming Section 3.2.7). (Durme et al., 2003) explicitly state the limitation of current tools to compute (semantic) similarities exclusively within the same part of speech. As already mentioned, simple stemming methods to *abstract* lemmas do not help in this case. The reduction of the two words *termination* and *f nished* to *terminat$* and *f nish*, for example, cannot be used as input for tools computing semantic similarities (since *terminat$* is an abstract word form). So, the knowledge-driven mapping of the two word forms is the most appropriate solution for that problem. I hope that by using a lexicon which offers a mapping between nouns and verbs, I can increase the scope of measuring semantic similarities.
(3.11) is required for the same reasons as it is the case when dealing with verbs. (3.12) is needed since the assignment of grammatical functions to complements of nominalizations is not that straightforward as it is the case with verbs. Normally, one can identify the NP immediately preceding the (active) verb as the subject and

[9]It also includes other parts of speech than verbs but they will not be considered for further processing.

[10]For future work I would recommend using the new subcategorization lexicon called VALEX (Korhonen, Krymolowski, & Briscoe, 2006) since it has a larger coverage and more detailed information concerning subcategorization frames than COMLEX. The resource has been released in late 2006.

the NP immediately succeeding the verb as the object. Nominalizations do not follow these rules. The assignment of grammatical functions can only be determined with the help of explicit listing of the mappings. The following sentences illustrate this:

(3.13) The eruption $[$of the volcano$]_{SUBJ}$ was regarded as a bad omen.

(3.14) The assassination $[$of J.F.K$]_{OBJ}$ happened on 22nd November 1963.

(3.15) His removal $[$from government$]_{LOC}$ was not expected.

(3.16) The comment $[$from John$]_{SUBJ}$ was totally inappropriate.

One could distinguish between a $PP_{[of]}$ being a subject in Sentence (3.13) and an object in Sentence (3.14) with the help of valency information without further lexical information, i.e. *eruption* is intransitive and *assassination* is transitive, but, one could not distinguish between the locative function (this is no grammatical function) of the PP_{from} in Sentence (3.15) and its function as a subject in Sentence (3.16).
The only lexicon that is publicly available and which fulfils these criteria mentioned above is NOMLEX (Macleod, Grishman, Meyers, Barret, & Reeves, 1998). The original NOMLEX contains approximately 1000 entries which were manually annotated[11]. The extension which I use, called *NOMLEX-Plus* (Meyers et al., 2004), has been semi-automatically extended. In total, this lexicon contains 4900 nominalizations. This extension also includes nouns that take arguments like nominalizations, but are not morphologically related to any verb. This would, for example, allow us to map *to live* in Question (3.17) onto *home* in Answer Sentence (3.18).

(3.17) Where did Ronald Reagan live?

(3.18) Ronald Reagan, ex-president of the USA, has died at his home in Bel Air, Los, Angeles.

This is a very useful feature for QA since the coverage of potential mappings is further increased.
As far as grammatical functions are concerned, I only considered the functions subject and object since NOMLEX rarely states indirect objects and its inclusion would have complicated processing.

3.2.6 Syntactic Parsing

Syntactic parsing is needed as a basis for detecting the phrases of participating entities of events, recognizing subcategorization frames and deriving grammatical

[11] This lexicon also includes entries of partitive, relational and attributive nouns but they will be ignored for the present task.

relations. The only parsers that are appropriate for the current task are statistical parsers because they are sufficiently efficient for QA.
I tested two parsers that were available: the parser by Michael Collins (Collins, 1997) and the one by Eugene Charniak (Charniak, 2000). Their output is quite similar, Collins' parser sometimes produces flatter structures than Charniak's. Finally, I decided in favour of Collins' parser, however, since the output format of this parser can be better used for further processing. The output needs to be converted to a format which allows easy access to the different constituents of a parse trees. TigerXML (Mengel & Lezius, 2000) was found to be suitable. I also use (*TIGER API 1.8 - A Java Interface to the TIGER Corpus*, n.d.) which is a JAVA API for navigating TigerXML files which is publicly available.
I did not consider dependency-based parsers, such as MINIPAR (D. Lin, 1998), though it is more efficient parser, since dependency structures are not compatible with the remaining tools I use, such as COMLEX, NOMLEX or TigerXML. In order to use MINIPAR it would have required an automatic conversion of dependency structures to phrase structures which was also found too time-consuming for this thesis.

3.2.7 Semantic Lexicons

For the module to be built the only semantic lexicon which provides useful information is WordNet (Miller, Beckwith, Fellbaum, Gross, & Miller, 1990). This lexicon is concerned with lexical relationships (such as *hypernomy*, *hyponymy* or *meronymy*) between different semantic concepts also known as *synsets*. Each lexical entry is assigned to at least one synset. The structure of synsets and their different relations is similar to that of an ontology which allows some form of limited reasoning.
Another useful property of WordNet are its Lexicographer Files. These files are very general but informative classifications of subsets of synsets. A Lexicographer File only comprises a subset of synsets of the same part of speech. Table 3.4 displays the list of Lexicographer Files for nouns and verbs. Note that the Lexicographer Files are orthogonal to the synset graph encoding the different lexical relationships. As far as nouns are concerned these classes can be used as a simplified alternative to semantic roles from FS. The Lexicographer Files of verbs, on the other hand, might be seen as a simplified frame from FS. (Section 3.3 will explain this analogy in detail.)
For this thesis, I use WordNet via two different tools. The first is (*JWNL - Java WordNet Library*, n.d.) which is a JAVA API for navigating through WordNet. Common functions allow looking up the synsets of lexical items and traversing synset nodes in the *WordNet synset graph*. Apart from that it also provides more complex operations, such as computing the strength of a relationship holding between two synsets. I discovered, however, that these complex operations are highly instable and should not be used. I use this tool primarily for obtaining Lexicog-

Part of Speech	Lexicographer Files
noun	acts, animal, artifact, attribute, body, cognition, communication, event, feeling, food, group, location, motive, object, person, phenomenon, plant, possession, process, quantity, relation, shape, state, substance, time
verb	body, change, cognition, communication, competition, consumption, contact, creation, emotion, motion, perception, possession, social, stative, weather

Table 3.4: Lexicographer Files of Nouns and Verbs in WordNet.

rapher Files of specif c lexical items. Since for QA determining the semantic strength of two synsets seems fairly useful I also use a Perl package called WordNet::Similarity (Pedersen, Patwardhan, & Michelizzi, 2004) which calculates the distance of two synsets in terms of *hyperonymy* relations. The advantage of this tool is that it supports state-of-the-art metrics (Budanitsky & Hirst, 2006) in order to calculate semantic distances in WordNet. All these metrics return a numeric weight which can be used for automatic comparisons of the synsets of two lexical units.

3.2.8 Semantic Parsing

The only semantic parser that is freely avaible at the moment is *Shalmaneser* (Erk & Padó, 2006) which labels plain text with complete frame structures to be found in FrameNet. The processing comprises recognizing *frame evoking elements (fees)*, determining their frames and labelling their *frame elements (fes)*. The tool comes with a *ready-to-use* mode, i.e. both syntactic and semantic parse models have already been trained. In this mode, syntactic parsing is done via Collins' parser (Collins, 1997). Because of the limited time I could only test the tool within this mode.

Unfortunately, due to the low performance of this tool on the TREC 2005 questions I could use not it for the f nal implementation. Various reasons are responsible for that. Appendix A describes in detail which problems were encountered and suggests possible explanations.

3.3 An Alternative but Similar Model to Frame Structures

The previous section stated that semantic processing using FrameNet cannot be used due to the insuff ciently robust performance of *Shalmaneser*. Consequently, an alternative model has to be devised. This model should be an approximation of

frame structures (since frame structures are still assumed to be the most appropriate form for this task) by making use of other existing NLP tools which perform better.
In order to allocate different but semantically related lexical units to one event, i.e. if one intends to approximate frames, one can use lexical resources, such as WordNet (see also Section 3.2.7). Thus, different *edes*, such as *to shoot* and *to kill*, can be related to the same concept.
Additionally, WordNet and NE taggers can be easily used for assigning semantic classes to the participating entities of events. The insurmountable problem, however, is that these semantic classes, such as *person*, *object* or *cognition*, are def ned independently of events. Thus, there are event scenarios in which these labels fail to discriminate between the different participating entities. For example, imagine a murder event like *X shot Y*. General semantic labels would label both X and Y as *person* and would not help us in a subsequent question like *Who shot Z*[12]*?* to distinguish between *X* and *Y*. Semantic roles provide the expressive power to discriminate between *X* which is regarded as the *agent* and *Y* which is the *patient*. *X* is the answer, since the question asks for the *agent* and not the *patient*. One might wonder, however, in how far the usage of grammatical functions, such as *subject* and *object*, can be harnessed in combination with general semantic classes in order to approximate semantic roles. In simple cases, such as *X shot Y*, grammatical functions can be used to distinguish between *X* being a subject and *Y* being an object. They can be derived from subcategorization frames. The assignment of these functions can become complex, however, if one considers passivization, controlling or raising[13]. Since the detection of grammatical functions can be erroneous, one could augment the entities with semantic classes mentioned above. For, in some cases, such as Sentence (3.19), one can distinguish the subject and object just by their semantic tag.

(3.19) [Peter]$_{person}$ wrote [a letter]$_{communication}$.

In my method, I will make usage of both semantic tags and grammatical functions in order to have as much information as possible in order to distinguish different entities participating of some event.

[12]It is not trivial to unify Z and its counterpart in a relevant answer sentence, such as Y in the current example, since the entity might not be realized as the same NP, e.g. *John F. Kennedy* and *the President of the United States*.

[13]These terms are explained in Appendix B.

Chapter 4

Method

This chapter presents a model for event-based QA. The overall design is displayed in Figure 4.1. Both questions and their corresponding candidate answer sentences are transformed from plain text to event structures. This will be explained in Section 4.1. Event structures are more abstract representations of the sentences which include various linguistic information. The event structures generated from questions need some post-processing, as will be described in Section 4.2. These filters rule out insufficiently processed questions and non-event questions. Answer sentences do not need this kind of processing. However, some effort must be spent in finding them in the AQUAINT corpus. How this is done will be described in Section 4.3. Event-based representations of a question and candidate answer sentence contain all information that are needed for matching them as will be explained in Section 4.4. The goal of this matching is twofold. At first, a candidate answer sentence must be checked for relevance concerning the question. Then, if the sentence fulfils sufficient criteria, an answer snippet will be extracted from it. Since the matching of questions and candidate answer sentences is done by a data-driven model combining various information from different linguistic and non-linguistic levels, its unknown parameters must be estimated. How this is done and how the necessary labelled training data are acquired will be described in Section 4.5. Finally, Section 4.6 will discuss a descriptive statistics that I have obtained from the labelled training data. Hopefully, this might give some insight into the characteristics of the answer sentences of event questions. This section should be regarded as a digression supplementing the contents of Chapter 3.1.

4.1 From Plain Text to Event Structures

How the transformation of plain text to a representation of event structures is achieved is illustrated in Figure 4.2. The first processing step is pronoun resolution because the tool I use (see also Chapter 3.2.4) requires plain text as input format. Following this, the text is processed by a POS tagger (see Chapter 3.2.2). Its output is both processed by a syntactic parser (see Chapter 3.2.6), a NE tagger

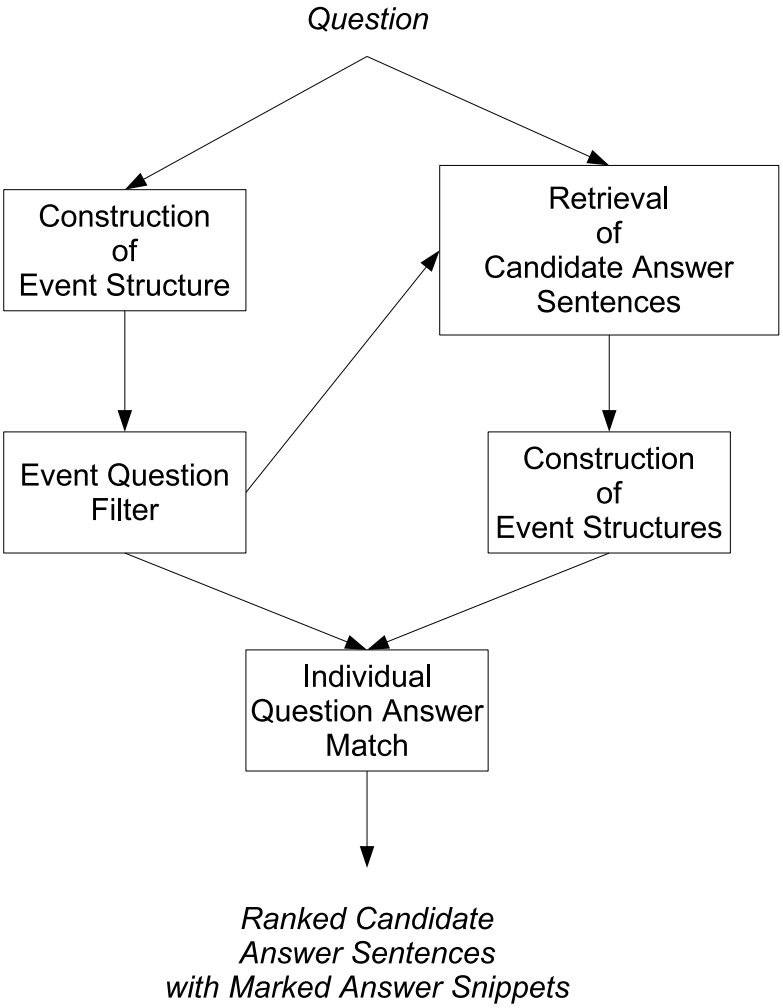

Figure 4.1: Overall Design of the System.

(see Chapter 3.2.3) and a look-up method using WordNet. Note that only full verbs and (common) nouns are looked up. An expression is not tagged with the label of its synset[1], as it is normally the case in other methods, but with the name of the Lexicographer File which includes this synset (see Chapter 3.2.7). This is done since these names are sufficiently general for semantic comparison between two different lexical units. As far as nominal expressions are concerned, the tagging of the two semantic components is complementary. A NE tagger tags proper nouns whereas WordNet labels common nouns. For reasons of uniformity the labels of the NE tagger are mapped onto corresponding Lexicographer Files.
The output of the parser are phrase-structure trees encoded in TigerXML (Mengel & Lezius, 2000). These trees are far too variable to be considered a basis for matching questions and answer sentences. That is why they are converted to some form of predicate-argument structure which also explicitly states the three main grammatical relations, *subject*, *object* and *indirect object*[2]. This is an abstract representation which ignores some variations in syntactic surface structures (i.e. phrase structure trees), such as active and passive alternation (see Examples (4.1)-(4.4)) or the different ordering of syntactic constituents in questions and declarative sentences (see Examples (4.5)-(4.8)).

(4.1) How many people were killed by Kip Kinkel?

(4.2) kill([Kip Kinkel]$_{SUBJ}$,[how many people]$_{OBJ}$)

(4.3) Kip Kinkel killed 4 people.

(4.4) kill([Kip Kinkel]$_{SUBJ}$,[4 people]$_{OBJ}$)

(4.5) Whose policy did the U.N. criticize?

(4.6) criticize([the U.N.]$_{SUBJ}$,[which policy]$_{OBJ}$)

(4.7) The U.N. criticized Iran's nuclear policy.

(4.8) criticize([the U.N.]$_{SUBJ}$,[Iran's nuclear policy]$_{OBJ}$)

This transformation can become fairly difficult when it comes to imperatives since the imperative itself along its corresponding predicate-argument structure has to be ignored (because the imperative will not turn up in the answer sentence). This is illustrated in Examples (4.9) and (4.10):

(4.9) Name the most famous movie produced by Steven Spielberg.

[1] By *the* synset I mean the first synset WordNet offers as a lexical unit. This synset normally encodes the most frequent meaning of a word. Any word-sense disambiguation (which would select the most appropriate synset for a term given its context) is neglected since processing would have been beyond the scope of this thesis.

[2] Note that due to technical restrictions on processing nominalizations only *indirect objects* of verbs can be recognized.

(4.10) produce([Steven Spielberg]$_{SUBJ}$,[the most famous movie]$_{OBJ}$)

Another difficulty in Sentence (4.9) is the elliptic relative clause which has to be normalized as well[3].
For traversing the phrase structure trees I use (*TIGER API 1.8 - A Java Interface to the TIGER Corpus*, n.d.). In order to produce precise subcategorization frames I use COMLEX, a subcategorization lexicon for verbs, and NOMLEX, a subcategorization lexicon for nominalizations (see Chapter 3.2.5).
In general, the method for constructing a predicate-argument structure from a parse tree follows conceptually the method proposed in (C.-S. Lin & Smith, 2006), though I had also to deal with questions and nominalizations, which makes my algorithm considerably more complex than the fairly simple algorithm in (C.-S. Lin & Smith, 2006).

After syntactic and semantic information have been acquired they are combined. This representation is a simplified representation of frame structures from FS (which has already been briefly introduced in Chapter 3.3). Given the structure in Example (4.4) the predicate and its arguments obtain semantic labels. The corresponding structure looks like:

(4.11) kill^{change}([Kip Kinkel]$^{person}_{complement:SUBJ}$,[4 people]$^{person}_{complement:OBJ}$)

Finally each predicate-argument structure is augmented by *satellites*. They are nominal expressions which could not be identified as syntactic arguments (either complements or adjuncts) but are in the vicinity of a predicate. A satellite can relate to any *ede* occurring in the same sentence. One reason why these entities are included is that due to the limitations of the parser some entities cannot be established as a syntactic argument to a predicate. A typical example where the inclusion of satellites would be vital is Sentence (4.13) which contains the answer to Question (4.12):

(4.12) Who won the 1998 Nobel Prize in Literature?

(4.13) After [Naguib Mahfouz]$^{person}_{satellite}$, [who]$^{undef}_{complement:SUBJ}$ [won]$^{competition}_{pred}$ [the Nobel Prize in Literature]$^{possession}_{complement:OBJ}$ [in 1998]$^{time}_{adjunct}$ ...

(4.13) also summarizes the information present in an event structure[4]. These (graph) structures are encoded in a specifically designed XML-format (which is conceptu-

[3] This normalization could only be performed on questions since a naive algorithm is not sufficiently robust for the vast syntactical variability in the AQUAINT corpus. The TREC 2005 question set, on the other hand, is far more syntactically restricted which is why a simple algorithm works here fairly well.

[4] To be precise, this illustration is still a simplification. Most sentences have more complex structures when multiple events are evoked. This is due to the fact that some expressions can be multifunctional. For example, a nominalization can be both a predicate, i.e. *ede*, and a complement of another *ede* being a verb, or an entity may be a satellite in one event structure and a complement in another.
Apart from that, each entity is also characterized by the spatial distance to its *ede*. This is necessary since some metrics in my answer extraction algorithm need this information.

ally very similar to TigerXML). As far as questions are concerned, the question constituent (normally, the phrase in which the interrogative pronoun occurs) is specially marked. There are basically two types of questions constituents, one in which there is only an interrogative pronoun, as in Question (4.14), and the other in which there is an NP with a real head noun, as in Question (4.15):

(4.14) [Who] killed J. F. Kennedy?

(4.15) [How many crewman] were lost in the disaster?

Since in Question (4.14) the corresponding NP is semantically empty, no semantic information can be deduced from the pronoun; WordNet and the NE tagger will not output any information. That is why the question type can be additionally used as a semantic label. For this thesis, I have used the question types presented in (Li & Roth, 2005). The labels were assigned by using the output of the question classifier of the Alyssa system (Shen et al., 2006). For matching operations with candidate answer constituents, it is of course necessary to match these labels with the ordinary semantic labels. A mapping for all common question types onto appropriate Lexicographer Files of WordNet has therefore been implemented. For more information, see also Appendix E.
In cases where the question constituent comprises more than just the interrogative pronoun, such as in Question (4.15), semantic information from the head noun (in this case *crewman*) can also be considered.

4.2 How Event Questions are Filtered

When questions have been converted to corresponding event structures they have to be filtered. There are two reasons for that. On the one hand, the processing of questions is fairly error-prone, and if certain premises are not fulfilled, the question will not be considered further. For example, if there is no parse for the question[5] or vital syntactic information, such as the detection of the main verb or the question constituent, could not be detected, the question is not processed further. On the other hand, not every question which could be successfully transformed to an event structure is an event question. This may sound paradoxical at first, but the event structure presented in the previous subsection is a mere transformation from a parse tree to a more abstract structure, hence, it is a transformation of one representation format to another but no inherent classifier. The difficulty of building such a classifier is that a high precision cannot be pursued since it would reduce the recall too much. Given that only the TREC 2005 question set could be worked with, a reasonable recall must be maintained. This means that the linguistic constraints that are incorporated in my method must not be too restrictive. The question sets from previous years could not be used, since the proportion of event questions is

[5] Collins' parser constructs a pseudo parse tree comprising one non-terminal combining all terminal nodes in case the sentence cannot be parsed.

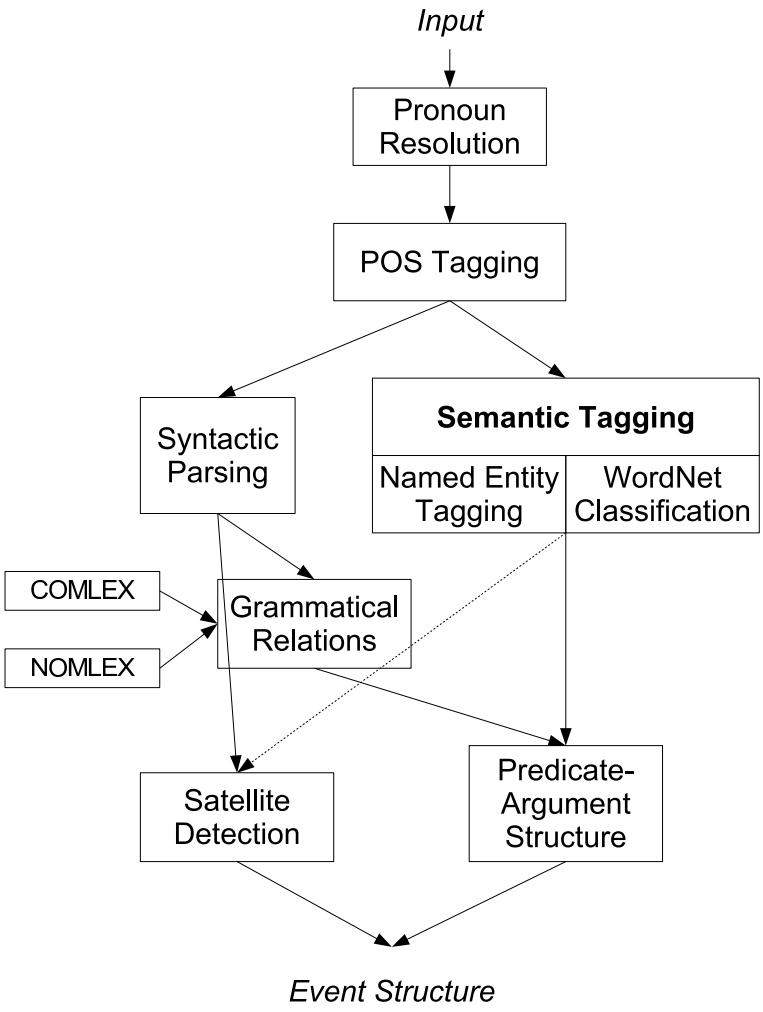

Figure 4.2: From Plain Text to Event Structure.

extremely low. Neither could the dataset of TREC 2006 be used since it is not available at the time of the implementation of my module.
The f lter I have designed uses three different kinds of information being:

- syntactic information
- question type
- lexical information

There is one major syntactic pre-requisite for event questions but it has a fairly strong discriminatory power. It requires the main verb to be a full verb or, in case of copula constructions, the main predicate to be a nominalization. Thus, one can identify Questions (4.16) and (4.17) as event questions and Question (4.18) as a non-event question:

(4.16) Who killed John F. Kennedy?

(4.17) Who was the killer of John F. Kennedy?

(4.18) Who was John F. Kennedy?

Unfortunately, not all full verbs indicate event questions. For example, Question (4.19) would not be excluded by the previous rule but it is clearly not an event question. It should not be treated as such since the kind of event-based modelling that I propose in this thesis would not identify the answer snippet in a typical answer sentence, such as Sentence (4.20)[6]:

(4.19) What does the abbreviation WHO stand for?

(4.20) The World Health Organization (WHO) was established on 7 April 1948.

Given the question type set I use in my implementation, Question (4.19) would be classif ed as an *abbreviation question*. All of these questions should be excluded from further processing. There are also other question types which indicate non-event questions. Appendix C lists all question types and whether they include or exclude event questions.
There are, however, questions which, from a syntactic point of view, could be event questions and the corresponding question type does not exclude event questions, and yet these questions should not be treated as such. Typical examples are a subtype of locative questions. Question (4.21) is a locative non-event question whereas Question (4.22) is an locative event question:

(4.21) Where is Port Arthur located?

(4.22) From where did the Hindenburg start on her f nal journey?

[6]This is due to the fact that this answer sentence only evokes an *establish* event which cannot be found in Question (4.19) and, thus, this question-answer pair cannot be matched on the basis of common events.

In order to detect further non-event questions, such as Question (4.21), I wrote some surface patterns. These rules basically comprise a set of some lexical items which should not occur in certain syntactic conf gurations (e.g. exclude all locative questions with *locate* as the main verb in passive voice).
All in all, the f lter I used classif es 234 of the questions as event-questions[7]. Approximately, 193 questions are real event-questions. This is a satisfactory result which produces suff cient data for further processing.

4.3 How Candidate Answer Sentences are Computed

There are two different ways how candidate answer sentences are retrieved:

(4.23) Using (*TREC Answer Patterns*, 2005) with pointers to documents

(4.24) Using QTile (Leidner et al., 2004) in combination with the document retrieval output of the Alyssa system (Shen et al., 2006).

These two types of candidate answer sentences are needed for two different evaluation scenarios which will be discussed in Chapter 5.1 and Chapter 5.2, respectively. (4.23) are manually labelled candidate sentences. If I test my module on these data, I obtain a fairly unbiased evaluation of its performance. Misclassif cations solely derive from shortcomings of my method.
In order to assess how my model works in real life, it cannot rely on such artif cial data but must obtain its input from a retrieval module of a QA system (as (4.24) suggests). For the retrieval of candidate answer sentences from a list of ranked documents which is the output of Alyssa's document retrieval component I use *QTile* which is a component of the *QED-system* (Leidner et al., 2004)[8]. As a query I do not use the original question but some corresponding event-based representation. For example, Question (4.25) is reformulated as Example (4.26):

(4.25) What was the attendance at Super Bowl XXXIV?

(4.26) {attendance, at Super Bowl XXXIV, attend}

As processing answer sentences, in general, is very time-consuming, I decided to process only the top 15 sentences that QTile outputs. Given this amount of sentences an optimal conf guration of the remaining parameters of this tool was determined by iterative optimization[9]: Of the top 60 documents returned by Alyssa,

[7] Note that I treat *list* questions in the same way as *factoid* questions. An individual treatment of these two types would have been too time-consuming. The omission of the smaller question class, i.e. list questions, would have decreased the amount of event questions considerably since the proportion of event questions among list questions is comparable to that among factoid questions.

[8] I could not use Alyssa's sentence retrieval component because it was not available when I started my implementation.

[9] This optimization was performed on 100 questions of the total of 234 (presumed) event questions (see also Section 4.2) of the TREC 2005 question set. Note that these data were not part of the test set used in the f nal evaluation (see also Section 5.2).

QTile only considers the top 20, and uses the 2 most highly ranked sentences within each of these documents for the f nal ranking from which only 15 sentences (out of 40 possible) are returned for matching questions and candidate answer sentences. This optimal conf guration achieved a *mean reciprocal rank* (Voorhees, 2000) of 0.257.

4.4 Matching Questions and Candidate Answer Sentences

This section presents the most important part of my implementation, namely the method how a question and potential answer sentences are matched. Before I will explain the algorithm, however, I will introduce some terminology which is vital for understanding my method.

4.4.1 Some Terminology and Def nitions

The terms explained in the following will be dealt with in a *bottom-up* fashion, i.e. I will start with the terms describing atomic units and continue with the terms which denote more complex units.
My method is organized by different kinds of *mappings*. A *mapping* aligns some linguistic expression of a question with one linguistic expression of a candidate answer sentence. The mapping also requires that the type of the linguistic expression of a question and a candidate answer sentence are the same or at least compatible with each other.
The three atomic mapping types are illustrated in Figure 4.3. They are: *edeMaps*, *qArgMaps* and *argMaps*. *EdeMaps* are mappings of event-denoting expressions (*edes*). These are linguistic expressions which evoke events. A formal type schema of this mapping type is:

(4.27) $edeMap$: `ede` $\rightarrow$ `ede`

Since, in my implementation, event evocation can only be bootstrapped via some form of predicate-argument structure, *edes* are conf ned to full verbs or nominalizations. Basically, these are the common expressions representing predicates. In Figure 4.3, there are two positive mappings being *attempted* $\rightarrow$ *tried* and *take over* $\rightarrow$ *take over* [10].
QArgMaps are the mappings from question constituents to answer constituents. As far as the type of these constituents are concerned, they are just linguistic units describing entities. In the context of event structure, entities can be either *complements*, *adjuncts* or *satellites* (the terms are explained in Appendix B). A formal type schema is:

(4.28) $qArgMap$: `entity` $\rightarrow$ `entity`

[10] The negative mappings are *attempted* $\rightarrow$ *take over* and *take over* $\rightarrow$ *tried*. I will not consider the negative mappings in the forthcoming examples.

In our current example, i.e. Figure 4.3, the positive mapping would be *what cruise lines → Carnival Cruise Lines*. Note that the answer constituent is not automatically the answer snippet which the program should output. It should only include this snippet. Take for instance the following question-answer pair:

(4.29) How many crewmen were lost in the disaster?

(4.30) 118 crewmen died in the accident.

The positive *qArgMap* is *how many crewmen → 118 crewmen* but the answer is only *118* and not *118 crewmen*.
ArgMaps are all mappings of entities surrounding an *ede* with exception of the question constituent and the answer constituent which have their own mapping type. The type schema is:

(4.31) $argMap$: `entity` → `entity`

In our current example, the positive *argMaps* are *NCL → NCL* and *in December 1999 → NIL*. The latter mapping is called positive since the only correct mapping for the temporal expression is a *zero-argument* since the answer sentence does not contain any appropriate counterpart. This example also exemplif es that, in practice, these mappings often remain partial, i.e. the set of expressions of a question can only rarely be mapped completely onto expressions of an answer sentence. Due to the fact that TREC questions tend to be very short, they do not contain many arguments so that, usually, there are only one or two positive *argMaps* within one question-answer pair.
EsMaps are mappings of event structures. Unlike the previous types of mappings, this is a complex type, which comprises all atomic mappings above mentioned. The type schema is:

(4.32) $esMap$:
`<edeMap,qArgMap,`$list$`(argMap) >` →
`<edeMap,qArgMap,`$list$`(argMap) >`

Figure 4.4 illustrates one instance of such a positive mapping type. (There is another positive instance for the *attempt/try* event). Both structures can be described as two attribute-value matrices:

$$\begin{bmatrix} \text{esMapI} & \begin{bmatrix} \text{edeMap} & \text{tried} \rightarrow \text{attempted} \\ \text{qArgMap}_{complement} & \text{what cruise line} \rightarrow \text{Carnival Cruise Lines} \\ \text{argMapI}_{satellite} & \text{NCL} \rightarrow \text{NCL} \\ \text{argMapII}_{satellite} & \text{in December 1999} \rightarrow \text{NIL} \end{bmatrix} \end{bmatrix}$$

(4.33)

$$\left[\begin{array}{ll} \text{esMapII} & \left[\begin{array}{ll} \text{edeMap} & \text{take over} \rightarrow \text{take over} \\ \text{qArgMap}_{satellite} & \text{what cruise line} \rightarrow \text{Carnival Cruise Lines} \\ \text{argMapI}_{complement} & \text{NCL} \rightarrow \text{NCL} \\ \text{argMapII}_{adjunct} & \text{in December 1999} \rightarrow \text{NIL} \end{array}\right] \end{array}\right]$$

(4.34)

Why do entities in *qArgMap* and *argMaps* appear in both structures (though with a different status[11])? Firstly, event structures are no strict *predicate-argument structures*. Otherwise, the constituents *NCL* and *in December 1999* would, for example, not appear in the first structure because they are no direct arguments[12]. Secondly, one should recall the task for which these structures are to be used. Events of questions are matched with events of answer sentences. They can only be characterized by their individual contexts. Since positive matchings are mostly partial, it is vital to consider as much context information as possible. This means that one should not restrict oneself to mere syntactic arguments (that could be identified) but consider other neighbouring linguistic entities, as well. (I defined these expressions as *satellites*.) This is, in particular, vital since the syntactic processing of the my implementation is very limited. For example, in the second event structure *qArgMap*, i.e. the mapping between *what cruise line* and *Carnival Cruise Lines*, can only be classified as a mapping of two *satellites*, since the *controlling*[13] relationship of each of these entities to their respective *edes*, i.e. *what cruise line* to *take over* and *Carnival Cruise Lines* to *take over*, is not recognized.
Question (4.35) displays a case in which the function of the satellite is essential in order to establish some connection between the question constituent *which cruise line* and the *ede take over* at all. This is necessary, since we cannot match Question (4.35) and Answer Sentence (4.36) via the *announce* event, since it is only present in the question (and this *ede* directly relates to the question constituent, i.e. it is a complement).

(4.35) Which cruise line announced to take over NCL in December 1999.

(4.36) Carnival Cruise took over NCL.

Of course, there is another problem when one tries to match *what cruise line* and *Carnival Cruise Line* since the former is a satellite of *take over* and the latter is a *complement* of *took over*. Thus, the complete match of this event structure would look like (4.37):

[11] For example, *NCL* is a *complement* in Example (4.34) but only a *satellite* in Example (4.33).

[12] *try* selects the entire VP *to take over NCL in December 1999* instead of merely the two embedded NPs.

[13] This term is explained in Appendix B.

$$\left[\begin{array}{ll} \text{esMapIII} & \left[\begin{array}{ll} \text{edeMap} & \text{take over} \rightarrow \text{took over} \\ \text{qArgMap}_{?} & \text{what cruise line} \rightarrow \text{Carnival Cruise Lines} \\ \text{argMapI}_{complement} & \text{NCL} \rightarrow \text{NCL} \\ \text{argMapII}_{adjunct} & \text{in December 1999} \rightarrow \text{NIL} \end{array}\right] \end{array}\right]$$

(4.37)

Note that the type of *qArgMap* is marked with a question mark since the two entities do not match in syntactic terms. Such minor type clashes are allowed in my model. As I will explain in the following section, a positive match between expressions of a question and an answer sentence does not require complete type identity but a fair amount of similarity, i.e. matching operates on the basis of *soft* rather than on *hard* constraints.[14]

At this stage, I should also point out the difference between *ede* and event structures. An *ede* is the linguistic unit which triggers an event but it does not represent the entire structure with all participating entities which is represented by the event structure. For readers familiar with FrameNet, the similarity between a *frame evoking element* and an *ede*, on the one hand, and *frame structure* and event structure, on the other hand, might help to distinguish these two different notions.

Finally, there is *qaMap* which maps complete questions and answer sentences. Basically, this means that this mapping includes all previously mentioned mapping types. The corresponding type schema is:

(4.40) $qaMap: list(\texttt{esMap}) \rightarrow list(\texttt{esMap})$

Figure 4.5 illustrates this term.

4.4.2 The Mathematical Model

The mathematical model described in this section is to serve the two functions previously mentioned, which are: ranking the candidate answer sentences (and thereby assessing the relevance of each sentence with regard to the question) and extracting for each of the relevant sentences the most likely answer snippet. The model is mainly concerned with determining the quality of different mappings which are possible, given a question and a candidate answer sentence. This operation is def ned by the function val whose type schema is:

$$val : \texttt{map} \rightarrow [\texttt{0}; \texttt{1}] \tag{4.1}$$

[14] It is of course a bit dangerous to design a model in which a question-answer pair, such as (4.35)-(4.36), can be matched since also incorrect matchings, such as Question-Answer Pair (4.38)-(4.39), could be established when the context is that *Stella Lines* only attempted it but *Carnival Cruise Lines* succeeded in doing so.

(4.38) Which cruise line took over NCL in December 1999.

(4.39) Stella Lines tried to take over NCL.

Fortunately, I encountered no TREC questions where such misinterpretations can happen. Apparently, such scenarios are too complex for TREC.

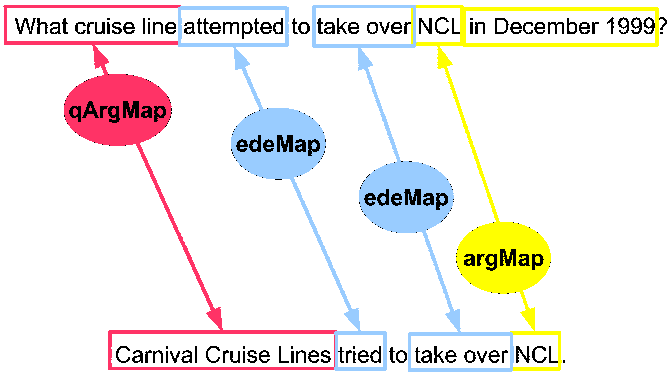

Figure 4.3: Atomic Mapping Types.

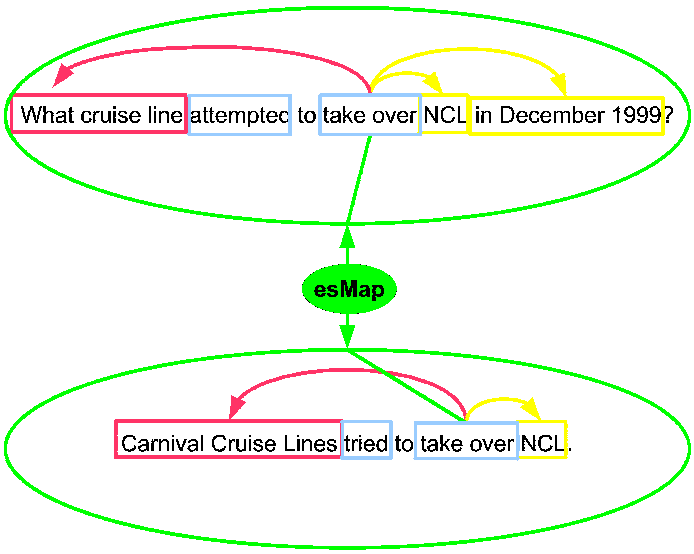

Figure 4.4: Event Structure Mapping.

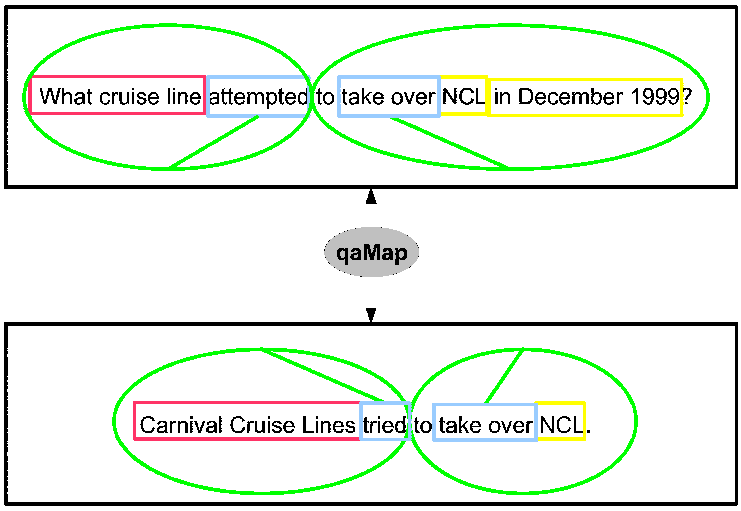

Figure 4.5: Question Answering Mapping.

The general formula of this function is a simple logistic regression formula:

$$val(map) := \sigma\left(\vec{w}^T \vec{f} + b\right) \tag{4.2}$$

`map` is a placeholder for all different mapping types (i.e. `edeMap`, `qArgMap`, `argMap`, `esMap` and `qaMap`). w_i in $\vec{w}$ are the weights for features f_i in $\vec{f}$ and b is a bias. Each feature f_i is a specif cally designed similarity measure which maps onto a value in $[0; 1]$ (where 1 indicates optimal similarity). The weights w_i are typically estimated by some supervised learning method. The sigmoid function σ (see also Equation 4.3) guarantees that the range of the output value of val is as claimed in 4.1.

$$\sigma(x) := \frac{1}{(1 + e^{-x})} \tag{4.3}$$

For more information about logistic regression including how to estimate the weights, please consult (Witten & Frank, 2005).

The best mapping $\hat{map}$ is def ned as:

$$\hat{map} := \arg\max_{map} (val(map)) \tag{4.4}$$

An optimal mapping would, therefore, be assigned the value 1 and the most inappropriate mapping would, conversely, be assigned 0.
The different $val()$ functions will be explained in a *top-down* fashion. The candidate answer sentences as a whole are assessed according to $val(qaMap)$ which is

def ned as:

$$val(qaMap) := \sigma\left(\alpha \cdot \max_i\left(val\left(\hat{esMap_i}\right)\right) + \beta \cdot irMatch + b_{qaMap}\right) \tag{4.5}$$

where $irMatch$ is a mapping which determines a question-answer pair by the IR-metrics. It is def ned as:

$$irMatch(q,a) := SSR(q,a)^{\alpha''} \cdot MTR(q,a)^{\beta''} \tag{4.6}$$

where q are the terms occurring in the questions, a is the list of terms occurring in the candidate answer sentence, $\alpha'' = 0.125$ and $\beta'' = 1$. α'' and β'' have been optimized on the TREC-9 data collection. Equation 4.7 def nes the *span size ratio (SSR)* and Equation 4.8 the *matching term ratio (MTR)*:

$$SSR(q,a) := \frac{|q \cap a|}{(1 + \max(mms(q,a)) - \min(mms(q,a)))} \tag{4.7}$$

$$MTR(q,a) := \frac{|q \cap a|}{|q|} \tag{4.8}$$

where mms is the *minimal matching span*. Given a matching span ms, let b_a (the beginning of the excerpt) be the minimal value in ms, i.e., $b_a = \min(ms)$, and e_a (the end of the excerpt) be the maximal value in ms, i.e. $e_a = \max(ms)$, a matching span ms is a *minimal matching span* (mms) if there is no other matching span ms' with $b'_a = \min(ms')$, $e'_a = \max(ms')$ such that $b_a \neq b'_a$ or $e_a \neq e'_a$ and $b_a \leq b'_a \leq e'_a \leq e_a$.

Given a question q and a candidate answer sentence a, where the function term_at_pos$_a(p)$ returns the term occurring at position p in a, a *matching span* (ms) is a set of positions that contains at least one position of each matching term, i.e. $\bigcup_{p \in ms}$ term_at_pos$_a(p) = q \cap a$. (For a detailed discussion of these metrics including their optimization, please consult (Monz, 2004). These metrics are typical for term-based approaches for answer extraction as discussed in Chapter 1.1. In this thesis, they are just used to back-off other more important structural metrics.)

The set of expressions denoted by $\hat{esMaps_i}$ with $i = 1 \ldots n$ are the optimal mappings for each of the n event structures appearing in a question. Mostly, however, a question contains only one event structure. Then, it is trivial to determine $\max_i\left(val\left(\hat{esMap_i}\right)\right)$.
The val function for $esMap$ is def ned as:

$$val(esMap) := \sigma\left(\alpha' \cdot val\left(\hat{edeMap}\right) + \beta' \cdot val\left(\hat{qArgMap}\right) + \gamma' \frac{1}{m}\sum_{i=1}^{m} val\left(\hat{argMap_i}\right) + b_{esMap}\right) \tag{4.9}$$

Note that the set denoted by $\hat{argMaps_i}$ with $i = 1 \ldots m$ are the optimal mappings for each of the m arguments appearing in a question.

The three atomic mapping types *edeMap*, *argMap* and *qArgMap* are def ned in the following equations:

$$val(edeMap) := \sigma\left(\sum_{i=1}^{l} w^i_{edeMap} \cdot f^i_{edeMap} + b_{edeMap}\right) \tag{4.10}$$

$$val(argMap) := \sigma\left(\sum_{i=1}^{l} w^i_{argMap} \cdot f^i_{argMap} + b_{argMap}\right) \tag{4.11}$$

$$val(qArgMap) := \sigma\left(\sum_{i=1}^{l} w^i_{qArgMap} \cdot f^i_{qArgMap} + b_{qArgMap}\right) \tag{4.12}$$

A detailed explanation of the features f_{edeMap} is given in Appendix D.1. Appendix D.2 lists the features of f_{argMap} and $f_{qArgMap}$. Note that though the feature sets for *argMap* and *qArgMap* are almost identical[15], my model allows corresponding weights of the individual features in the two mapping types to be different. This should increase the expressiveness of the model.
The set-up for learning the weights w_i (i.e. α and β in Equation 4.5, α', β' and γ' in Equation 4.9, $\vec{w}_{edeMap}$ in Equation 4.10, $\vec{w}_{qArgMap}$ in Equation 4.12, and $\vec{w}_{argMap}$ in Equation 4.11) will be discussed in detail in the Section 4.5.

Turning the Model into an Algorithm

This section brief y describes how the mathematical model can be used as an answer extraction algorithm. At f rst, one calculates the optimal mappings which are possible given an event question and a candidate answer sentence. The calculation can only happen in a bottom-up fashion. One begins with $ede\hat{M}ap$, $qAr\hat{g}Map$ and $arg\hat{M}ap$ since these mappings are needed for computing $es\hat{M}ap_i$ which, themselves, are needed for obtaining $qa\hat{M}ap$. After the calculation of the best mappings, candidate answer sentences are ranked according to $val(qaMap)$. The highest rank, therefore, contains $qa\hat{M}ap$.[16] The best answer snippet can be computed from the best question argument mapping, $qAr\hat{g}Map$ in $qa\hat{M}ap$. In case of more than one event structure in the question, one has to choose between all $qAr\hat{g}Map_{es\hat{M}ap_i}$. For reasons of simplicity I always take the mapping with the highest value. This method is particularly appropriate since, in case of multiple events in questions, rarely all events occur in relevant answer sentences.
If the question type to be processed is no *numeric* type (the set of question types are listed in Appendix E), the answer constituent of $qAr\hat{g}Map$ is also the answer snippet, otherwise one needs to f lter the numeric expression of the constituent. This is easily done with a look-up list of the major numeric expressions.

[15] Both of these mapping types have only one unique feature.

[16] In case one does not want to rank the candidate answer sentences but wants to classify them into relevant and irrelevant sentences, then one can use val as a *discriminant function* and classify all candidate answer sentences with $val(qaMap) > 0.5$ as relevant.

4.5 Data Optimization

The previous section described an algorithm to rank a list of candidate answer sentences according to relevance (concerning the question) and extract plausible answer snippets accordingly. This algorithm was def ned on the basis of a mathematical model. This section describes how the unknown parameters of the model, i.e. the weights of the different features within this model, can be obtained. I will f rst discuss the acquisition of labelled training data which are required to perform training and how I annotated them. Then, I will brief y discuss how the unknown parameters are learned. Finally, I will state the results of the learning method.

4.5.1 How the Manual Data are Acquired and Annotated

The basis of the manual annotation are the relevant answer sentences to the 193 event questions which could be identif ed and properly processed (as already stated in Section 4.2, my automatic question f lter returned 234 presumed event questions but 41 were false positives). In order to obtain the relevant answer sentences, I extracted the relevant documents for each of these questions using (*TREC Answer Patterns*, 2005) def ned by Ken Litkowski. Unfortunately, the patterns are incomplete which is why only 189 of the 193 questions could be considered further. I divided these questions into two partitions. The f rst partition comprising 100 questions was used for estimating the parameter weights and the remaining 89 questions were laid aside for testing my implementation[17]. The relevant sentences in each of the extracted answer documents had to be identif ed manually. Since there is often more than one relevant document for a question, I could obtain 349 question-answer pairs though I only had 100 questions. Unfortunately, a further 39 pairs had to be removed. In these pairs there have been major faults in processing the candidate answer sentences so that a correct answer could not have been extracted. These cases include incorrect pronoun resolution, missing parses[18], or answer snippets being elliptic constituents[19]. In total, there were 310 question-answer pairs for training. Figure 4.6 summarizes what amount of data has been used in which processing step.

I decided to estimate the unknown parameters in f ve separate learning scenarios, one for each mapping type in Section 4.4.2[20]. One decisive reason why I did not

[17] This scenario will be described in Chapter 5.1.

[18] Collins' parser occasionally fails to deliver a real parse for a sentence. Instead, it returns a pseudo-parse tree comprising only one non-terminal node directly dominating all terminal nodes.

[19] These are phrases with missing head nouns such as *113* instead of *113 crewmen*. During processing the text such words will not be considered as *entities*, since they are no proper NPs.

[20] I should point out that the number of training instances for the different mapping types is not identical to the number of question-answer pairs, i.e. 310. For example, there can be more event structures and arguments in one question-answer pair. In some cases, on the other hand, relevant answer sentences do not contain similar events to the ones found in the questions. In these cases no training instance can be obtained from these particular pairs. Section 4.6.1 will discuss this issue of

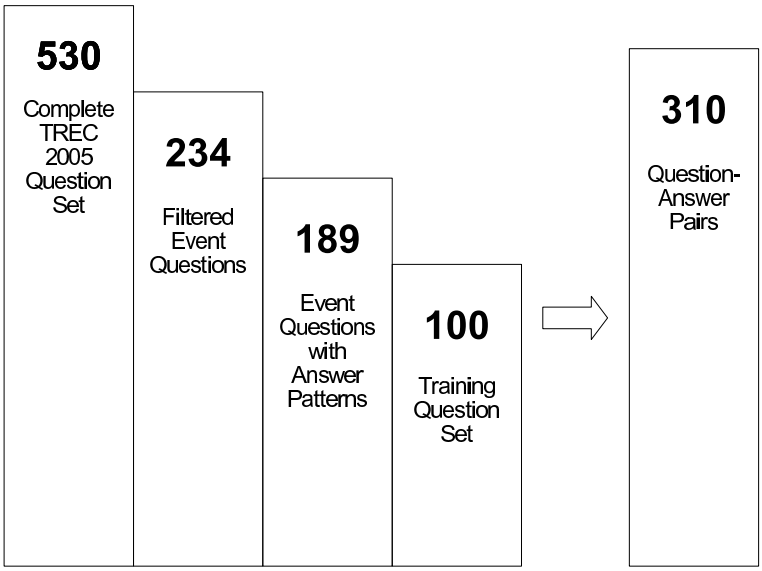

Figure 4.6: Usage of Data during Processing.

carry out a less time-consuming method, for example, by devising a single learning method, is that with the separate scenarios I have obtained important data for a descriptive statistics concerning the behaviour of answer sentences of event questions which will be discussed in Section 4.6.

Annotating Data for Atomic Mappings

This section describes how the data for parameter estimation of the formulae of atomic mappings, i.e. *edeMaps*, *qArgMaps* and *argMaps*, have been annotated. The procedure is identical in all three cases. In order to carry out the annotation economically, I only annotated the positive mappings. For example, when annotating a question-answer pair for the unknown parameters of *edeMap* for each *ede* in the question I explicitly labelled an appropriate mapping candidate in the candidate answer sentence, if present. The remaining mappings, which are negative but also theoretically possible, can be easily determined automatically. This procedure is particularly eff cient since there is often a strong imbalance between positive and negative mappings, i.e. there are far more negative mappings than there are positive mappings.

Annotating Data for Complex Mappings

The annotation of the data for the mapping formulae of the complex mapping types, i.e. *esMaps* and *qaMaps*, is different from the method applied for atomic map-

elliptic event structures in more detail.

pings. This is due to the fact that positive and negative mappings cannot both be taken from the set of the relevant answer sentences. Since these mapping types reflect the similarity of two entire sentences, namely question and answer sentence, it is not possible to obtain negative mappings from a question-answer pair where the candidate answer sentence is a relevant answer sentence. Consequently, I additionally extracted irrelevant candidate answer sentences in order to generate negative mapping instances. This is not trivial since the irrelevant candidate answer sentences should not be too dissimilar from the relevant candidate answer sentences, otherwise the learned parameters may not be sufficiently discriminating[21]. That is why I selected the set of irrelevant answer sentences for a question by taking sentences from documents which also included relevant answer sentences. This guarantees (in most cases) that the irrelevant sentences belong to the same topic as the relevant sentence and therefore should bear some resemblance to them.

4.5.2 Learning the Weights

I now turn to the estimation of unknown parameters of my model. Once positive and negative mapping instances have been annotated, each mapping instance must be augmented by a feature vector. Such a vector represents all characteristic properties of a specific mapping instance. For example, the feature vector for *qArgMap* in Equation 4.11 is $\vec{f}_{qArgMap}$ [22]. These vectors are converted to ARFF-format. An extract from such a file is displayed in Appendix F. Now, one can learn the parameter weights for the logistic regression formulae by using the WEKA toolkit. For more detailed information concerning the ARFF-format and the WEKA toolkit, please consult (Witten & Frank, 2005).

Though WEKA supports a logistic-regression learner[23], it is not advisable to use it in its standard mode. The problem is that the class distribution of the training data is highly imbalanced. There are always far more negative than positive training instances. For example, regarding the training data for *qArgMap*, the relation between positive and negative instances is approximately $1:59$. Such extreme imbalances have a significant impact on the performance on learning as (Drummond & Holte, 2005) and (Weiss & Provost, 2003) point out. The resulting classifier tends to classify instances to the majority class (which, in our case, are the negative mappings). The learning problem that I address here, would thus be seriously deteriorated, since I am, in particular, interested in recognizing positive mappings. In order to avoid this problem, one can either apply *re-sampling*, (i.e. *downsampling* or *upsampling*, as described in (McCarthy, Zabar, & Weiss, 2005)), or *cost-sensitive learning*. I prefer the latter since it does not modify the distribution of

[21] This insight can be exemplified with a simple classification problem. Imagine you want to build a classifier which is to distinguish the digit 1 from other digits in handwritten data. If your training material does not comprise instances of similar characters, such as the digit 7, but mostly fairly different digits, such as 4 or 8, the classifier may become less useful since it is trained to distinguish fairly dissimilar things, such as 4 and 1, but not different things which look similar like 7 and 1.

[22] The features are described in Appendix D.2.

[23] Per default, it uses *ridge estimators* as described in (Cessie & Houwelingen, 1992).

the training data as it is the case in re-sampling. Instead, training itself is altered in such a way that not the solution with the minimal *error* (as it is usually the case) but the solution with the minimal *cost* is computed. Costs can be applied to the different types of misclassif cations[24] by a *cost-matrix*. When dealing with a highly imbalanced class distribution, one can increase the importance of instances belonging to the minority class. This is achieved by giving misclassif cations of the actual members of the minority class, i.e. if the minority class is the set of positive instances, then this corresponds to the false negatives, a fairly high cost. One should also see that the cost of the misclassif cations of the majority class, i.e. in the current problem, these are the false positives, are given a fairly low cost. In general, the costs for false negatives and true positives should ref ect the size of the positive and negative class.

According to (McCarthy et al., 2005) the performance of cost-sensitive learning is comparable to re-sampling, sometimes it is even better. Fortunately, it is also implemented in WEKA and can be wrapped around logistic regression. (Witten & Frank, 2005) offer more detailed information about this type of learning including how it can be performed in WEKA.

Tables 4.1 and 4.2 illustrate the effect of cost-sensitive learning. The f rst table displays the confusion matrix of a classif er for the *qArgMap* which has been trained without cost-sensitive learning. Since the negative class, i.e. the class with false mappings, is considerably larger than the positive class, learning focuses on classifying instances of the former class correctly. As a consequence, only 3 of 203 positive instances are classif ed correctly. The second table displays the classif cation on the same data but this time cost-sensitive learning has been applied. There is a signif cant rise in the number of positive instances classif ed correctly (i.e. 3 to 177). This is exactly what should happen. Unfortunately, the improvement on the classif cation of the minority class goes at the expense of the performance on the classif cation of the majority class. The number of negative instances being classif ed incorrectly rises from 4 to 2939. This number may appear fairly high, but one should consider that incorrectly classif ed negative instances weigh far less than an incorrectly classif ed positive instances. Taking the distributional relation of these two classes into account, i.e. $1 : 59$, one could say that the 2939 misclassif ed negative instances weigh as much as approximately 50 misclassif ed positive instances which is a much more reasonable number.

The optimization of the parameters of *qaMap* has not been learned by logistic regression. Instead, I have solved this problem via iterative optimization. In order to do so, I had to modify the logistic regression problem in Equation 4.5, repeated

[24] In case of binary classif cation problems, such as the current problem, there are only two types of misclassif cations, being *false positives* and *false negatives*.

		Predicted Class	
		yes	**no**
Actual Class	**yes**	3	200
	no	4	11932

Table 4.1: Confusion Matrix of *qArgMap* without Applying Cost-Sensitive Learning.

		Predicted Class	
		yes	**no**
Actual Class	**yes**	177	26
	no	2939	8997

Table 4.2: Confusion Matrix of *qArgMap* having Applied Cost-Sensitive Learning.

in 4.13, to a simple linear interpolation 4.14:

$$val(qaMap) := \sigma\left(\alpha \cdot \max_i\left(val\left(es\hat{M}ap_i\right)\right) + \beta \cdot irMatch + b_{qaMap}\right) \tag{4.13}$$

$$val(qaMap) := \alpha \cdot \max_i\left(val\left(es\hat{M}ap_i\right)\right) + (1-\alpha) \cdot irMatch \tag{4.14}$$

Note that α is def ned in $[0;1]$.[25] I decided to use this formula here since it is the only of the f ve $val()$ equations in Section 4.4.2 which can be reformulated in such a way that there is only one unknown parameter, namely α, (note that β in Equation 4.13 is expressed as $1-\alpha$ in Equation 4.14). Only in such simple cases an iterative optimization is advisable. The benef t is that since the entire parameter space is explored, though, admittedly, with a very large stepsize[26], one also obtains a view on the course of the target function which is to be optimized.

4.5.3 Assessing the Features

There now follows a discussion of the robustness of the features used in my method. In order to assess the features individually I have trained each classif er on each corresponding feature separately. Thus, I can compare how the different features within one classif er differ in discriminatory power. The overall performance of these classif ers has been measured by both precision and recall (Rijsbergen, 1979). I could not use F-score (Rijsbergen, 1979), which combines these two measures, because the imbalance of the two classes (i.e. the class with positive instances,

[25] There is no such restriction imposed upon the feature weights in logistic regression.

[26] I varied α from $0.0-1.0$ with stepsize of 0.1.

which is usually very small, and the class with negative instances, which is usually very large) distorts this measure. The false positives (disproportionately) dominate the F-score. Consequently, the number of false negatives is mostly neglected. By looking separately at the precision and recall the high imbalance of classes is only present in the precision. The recall is not affected since this measure only regards the actual positive instances and thus, unlike the distorted F-score, this measure offers some information as to the size of false negatives. The precision are very small values due to the false positives which are part of the denominator. This means that relative size between the different features should be assessed rather than their absolute values.
Apart from the performance of the classif ers relying only on one single feature, I also included the performance of the classif er combining all features[27]. When assessing the overall performance of this resulting classif er one should recall that the best classif er is to maximize both precision and recall equally. The consequence of this is that if one looks exclusively at precision or recall, the fact that there are some features which have a higher score than the combined classif er does not mean that the combined classif er performs badly. Only if there is some feature which exceeds the combined classif er in both precision and recall, then this would be the logical consequence.
In addition to precision and recall, I also computed the mutual information of each feature pair within one classif er. Thus, one can get an overview of how distinct the features are from each other. Unfortunately, there is no upper bound for this measure, so it does not make sense to assess the mutual information of some feature pairs in absolute terms.

Features of EdeMap

Figure 4.7 illustrates precision and recall of the features of *edeMap*. There are three very strong features with both a high precision and recall, namely, `lemma`, `semI` and `semII`. (`pos` and `genverb` only have a high recall.) This result implies a high proportion of event mappings with identical lemmas as *edes*. It also shows that by using WordNet, i.e. by using feature `semI` or `semII`, one can signif cantly increase the recall (and simultaneously maintain a reasonable precision) since also synonymy and, to a small extent, other semantic relations are taken into consideration. The fact that both semantic features virtually perform equally well is a surprise since feature `semI` is a considerably simpler binary feature than the continuous feature `semII`. `mainarg` is the feature with the worst performance due to its low recall. I mainly ascribe it to the fact that it is a very weak feature. (The matching *edes* do not have to be the main predicates of the sentence.) The combined classif er performs well both in terms of precision and recall.

[27] Unfortunately, there was no time to do some feature selection in order to f nd the best combination of features. This does not have to be the full set of features. Sometimes, two features model the same information and thus, only one of them is necessary. It might also be the case that some features express irreconcilable views so that their co-occurrence in the f nal classif er is disadvantageous.

	pos	semI	semII	mainarg	frame	genverb
lemma	0.014	0.119	0.119	0.004	0.006	0.001
pos		0.007	0.011	0.010	0.019	0.032
semI			0.053	0.001	0.003	0.002
semII				0.002	0.007	0.002
mainarg					0.092	0.000
frame						0.041

Table 4.3: Mutual Information of *edeMap* Features.

Table 4.3 lists the mutual information of the feature pairs. The strongest features (in terms of precision and recall) are also fairly similar. The mutual information of {`lemma`,`semI`} and {`lemma`,`semII`} are the highest to be found. This might suggest that not all of these three features are absolutely necessary. However, the conclusion that one should remove one of the semantic features is not necessarily advisable, since the corresponding mutual information is not that high as the one of the two other pairs just stated. The fact that the feature pairs {`pos`,`genverb`}, {`mainarg`,`frame`} and {`frame`,`genverb`} are also similar to certain extent is no surprise, either, since these are all syntactic features. The remaining feature pairs contain relatively independent information.

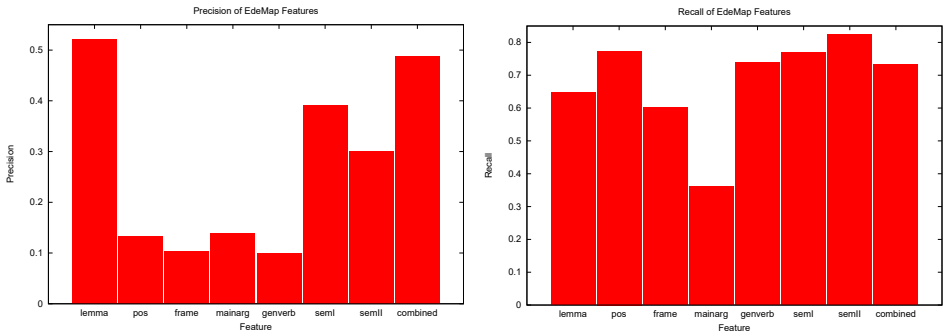

Figure 4.7: Precision and Recall of *edeMap* Features.

Features for QArgMap

Precision and recall of the features used in *qArgMap* are displayed in Figure 4.8. In general, there are two features which perform well in both evaluations. They are `dist` and `semIII`. Section 4.6.3 will look exclusively at the role of spatial distance and the statistic results should explain the performance of the corresponding feature `dist`. The fact that `semIII` is a very robust feature is a very fortunate result since it was specif cally designed for *qArgMap*[28]. The high precision of

[28]Note that all other features are also used in *argMap*.

argstat certainly catches the eye. Unfortunately, this feature has a poor recall, so it is only partially beneficial to the overall classification. A similar situation holds with the performance of semII and phrase but the performance in the two evaluation measures are opposite. Syntactic features, such as phrase and gram, have a mediocre recall and additionally an extremely low precision. So, there is no single syntactic feature which performs well in general .
The performance of the combined classifier has a low precision. Still, it is significantly higher than that of the average of the individual features and it also retains a very high recall at the same time. The low precision should be kept in mind for the future evaluation of the entire implementation. Since *qArgMap* is one central part of answer extraction, it will have a significant impact on the overall classification. In general, syntactic features, in particular gram, are thought to be fairly important for answer extraction. I assume that three things are responsible for this counterintuitive result. Firstly, there is a considerable number of answer constituents which are not directly syntactically related to *edes* (see also the results in the forthcoming Section 4.6.2). Secondly, there are syntactic relations which cannot be modelled with the output of Collins' parser (see also Section 4.6.2). Thirdly, I noticed that the syntactic analyses by Collins' parser were frequently erroneous.
The mutual information of the feature pairs (see Table 4.4) offers some interesting results. Again similarities of syntactic features, such as pos and phrase, are no surprise. But the high mutual information of the surface-based feature semII and the syntactic features phrase, pos and gram is unexpected. However, without a more detailed analysis, a proper interpretation of these similarities is not possible.
The fact that semIII has always a low mutual information is some positive result, since it is also the best feature as to precision and recall. This means one cannot even claim that this feature is lacking orthogonality to its sister features. A similar result offers dist, the second best feature. However, there are some similarities to the features argstat and gram. This means that one could waive these expensive syntactic features and still preserve some amount of information in feature dist. For example, together with ori, another surface-based feature which also shares some considerable information with syntactic features pos, phrase and gram, dist may often suffice to distinguish between a subject and an object of a clause.

Features for ArgMap

Figure 4.9 displays precision and recall of the features of *argMap*. As far as recall is concerned, there is only one weak feature, namely argstat. In *qArgMap*, this feature was already classified as high precision and low recall. The performance in *argMap* supports this view. Feature pos can be seen as the complete opposite of argstat having the highest recall of all features and a very low precision. As in *edeMap*, lemma is still one the best performing features. phrstr, which is conceptually very similar, also performs well. With the exception of argstat, all remaining syntactic features have a low precision. The surface-based features

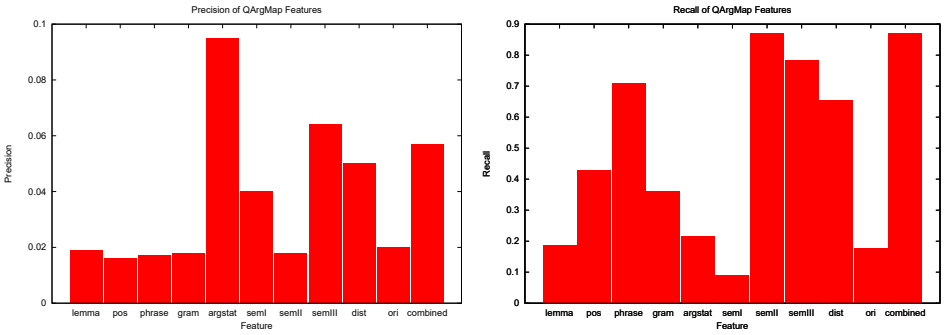

Figure 4.8: Precision and Recall of *qArgMap* Features.

	pos	semI	semII	semIII	phrase	gram	argstat	dist	ori
lemma	0.065	0.036	0.061	0.004	0.055	0.046	0.001	0.008	0.023
pos		0.050	0.290	0.001	0.301	0.223	0.002	0.003	0.138
semI			0.042	0.009	0.037	0.027	0.001	0.004	0.019
semII				0.004	0.236	0.157	0.002	0.027	0.001
semIII					0.001	0.001	0.000	0.001	0.001
phrase						0.373	0.004	0.040	0.197
gram							0.000	0.052	0.122
argstat								0.176	0.001
dist									0.020

Table 4.4: Mutual Information of *qArgMap* Features.

	pos	semI	semII	phrase	phrstr	gram	argstat	dist	ori
lemma	0.056	0.065	0.130	0.210	0.119	0.004	0.007	0.001	0.002
pos		0.012	0.131	0.054	0.062	0.051	0.080	0.018	0.000
semI			0.100	0.002	0.041	0.001	0.000	0.000	0.001
semII				0.082	0.060	0.050	0.039	0.009	0.003
phrase					0.018	0.106	0.031	0.020	0.002
phrstr						0.002	0.016	0.001	0.001
gram							0.139	0.048	0.000
argstat								0.045	0.000
dist									0.000

Table 4.5: Mutual Information of *argMap* Features.

`dist` and `ori` behave similarly to the majority of syntactic features. Again, this is no surprise, since similarities between these two groups were already discovered in *qArgMap*. Unfortunately, none of the semantic features plays such an outstanding role as it was the case in *edeMap* or *qArgMap*. The combined classif er has both a fairly high recall and precision, so it should give some good judgements.
As far as the mutual information of the feature pairs, as displayed in Table 4.5, are concerned, no really new observations can be made. Semantic features share information among each other, and so do syntactic features (to a certain extent). As in *edeMap*, the surface-based feature `lemma` is similar to many other features including all semantic features but this time also syntactic features, such as `pos` and `phrase`. As in *qArgMap*, the mutual information of the {`semII`,`pos`} is fairly high. Moreover, there is again some similarity between `dist` and the two syntactic features `gramfunc` and `argstat`, but not as strong as in *qArgMap*.

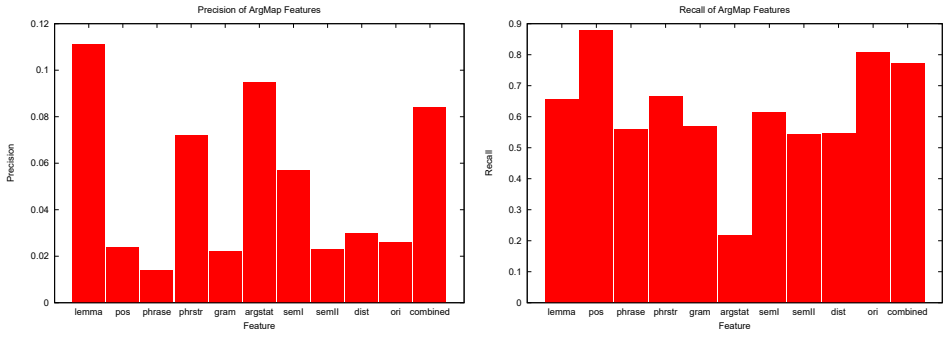

Figure 4.9: Precision and Recall of *argMap* Features.

Features for EsMap

The performance of *esMap*, as displayed in Figure 4.10, suggests that the two tasks of my model, namely determining the relevance of an answer sentence concerning a question and extracting an answer snippet of a relevant answer sentence are

	qargmap	argmap
edemap	0.008	0.064
qargmap		0.028

Table 4.6: Mutual Information of *esMap* Features.

two fairly *orthogonal* tasks. *EsMap* is a mapping type which is intended to contribute to the f rst task. So are the two features `edemap` and `argmap`. The feature `qargmap`, however, is to contribute to the second task which might explain why its performance is fairly low in *esMap*. From all of the four classif cation scenarios discussed so far, i.e. *edeMap*, *qArgMap*, *argMap* and *esMap*, only *esMap* has a combined classif er which exceeds the best classif ers trained on a single feature in both precision and recall. The mutual information of the three feature pairs (see also Table 4.6) shows only some mild similarity with {`edemap`,`argmap`}.

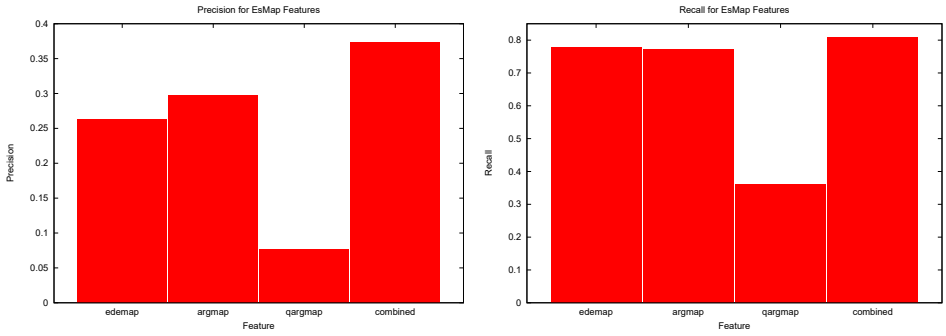

Figure 4.10: Precision and Recall of *esMap* Features.

Interpolation of Two Relevance Metrics for QaMap

In Section 4.5.2, I already stated that for *qaMap* I use a simple linear interpolation for combining *esMap* and *irMatch* instead of using logistic regression. Figure 4.11 shows the corresponding plot of the performance on the complete parameter space. Apart from the F-score[29], which is the measure that is optimized, I also displayed recall and precision. Recall from Equation 4.14, repeated in Equation 4.15, that $\alpha = 0.0$ means that only *irMatch* is considered and, conversely, that $\alpha = 1.0$ means that only *esMap* are considered.

$$val(qaMap) := \alpha \cdot \max_i \left(val\left(\hat{esMap_i} \right) \right) + (1 - \alpha) \cdot irMatch \qquad (4.15)$$

The plot clearly shows that *irMatch* has a high recall whereas *esMap* has a high precision, just as expected. Chapter 1.1 discussed the characteristic difference

[29]Note that as far as *qaMap* is concerned there is no extreme imbalance of classes so that the F-score would be distorted.

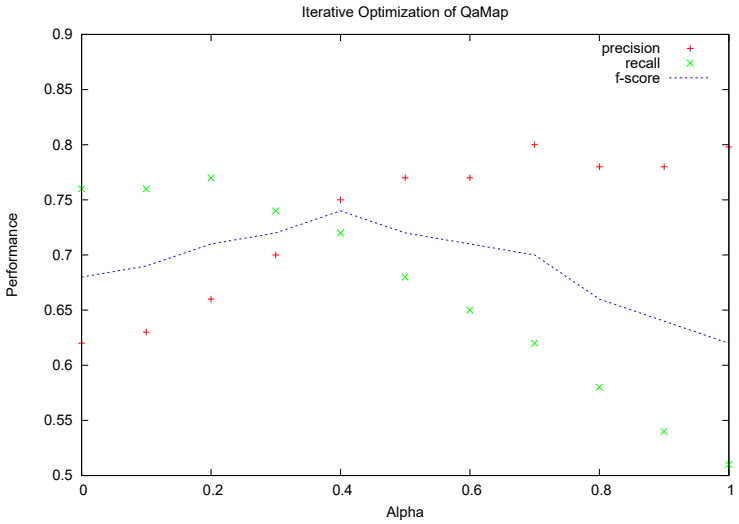

Figure 4.11: Iterative Optimization of *qaMap*.

between *term-based* comparison (in my model this corresponds to *irMatch*) and *event-based* comparison (in my model this corresponds to *esMap*). Considering these properties of *irMatch* and *esMap* the fact that the optimal conf guration is $\alpha = 0.4$ is fairly plausible since the optimized measure is a trade-off between recall and precision. This combination of the two approaches performs better than each individual approach. The best F-score of an individual approach, i.e. the F-score of the *term-based* approach, increases from 0.68 to 0.74 by including the information offered by the *event-based* approach. This is clearly a signif cant improvement of the overall performance.

Summary of Feature Extraction

In general, semantic and surface-based features have turned out to be the most robust features as to both precision and recall. Syntactic features do not perform that well. With only one exception, they have a fairly low precision. Only one feature, i.e. `argstat`, has a reasonable precision but has also a very low recall. Due to this performance analysis not all features included in each classif cation scenario should be really necessary in order to obtain a classif er with a performance similar to that of the combined classif er. The combined classif ers using all features within a classif cation scenario f nd a good trade off between precision and recall. In no classif cation scenario there is any feature which excels the combined classif er in both precision and recall. So, the combined classif er achieves satisfactory results despite the omission of some feature selection.

4.6 Digression: Evaluating the Annotated Data

Before describing the evaluation of my answer extraction algorithm in the forthcoming chapter, I will now turn to the annotated data for a moment and look at three different aspects being the different types of alignments between events (i.e. *edes*) in question-answer pairs, the different types of (syntactic) relations between answer constituent and main *ede* (in relevant candidate answer sentences) and the spatial distance between them. This evaluation is a digression and should be considered a complement to the preliminary data studies in Chapter 3.1. Each of these statistics has been collected on all 310 relevant question-answer pairs of the training data.

4.6.1 Evaluation of Event Alignments in Question-Answer Pairs

The purpose of this evaluation is to illustrate the complexity of aligning *edes* in event questions and the corresponding answer sentences. It shows what proportion of alignments should be handled by the current implementation, what other types there are and which of these types should be dealt with in forthcoming implementations. Table 4.7 displays the distribution of the different types of *edeMap*[30]. The different types are ordered according to their complexity.
The first type is an alignment where the *edes* are the same lemma. I consider an abstract lemma which subsumes word forms of different parts of speech, i.e. a mapping from *winner* to *to win* would be assigned this alignment type. With a relative frequency of more than 40%, this is the most frequently occurring type among the question-answer pairs from which I have obtained this statistics.
The next alignment type covers *simple synonyms* which can be acquired by using common semantic lexicons, e.g. WordNet. Thus, expressions, such as *to kill* and *to shoot*, can be matched. I encountered these alignments in more than 10% of question-answer pairs.
Complex synonyms are all those cases in which two words are synonymous but their relationship cannot be established by the method used for the previous type. Instances of this type comprise synonyms across different parts of speech, such as *victory* and *to win*. These relations can only be established with additional resources, such as NOMLEX.
My implementation will presumably only handle these first three types, since only these types were taken into consideration while designing my model. This amounts to approximately 52% of the entire sets of question-answer pairs. One should keep in mind this number when it comes to evaluating the performance of my method.
I included the next type *identical frames* though I never encountered it in the dataset I evaluated, since current research much focuses on the usage of FrameNet in various applications, including QA. The conclusion from this result should, however,

[30] I only consider the main *edes*, i.e. the main predicates of the sentences, when dealing with question-answer pairs. So, whenever the expression *the ede* appears, I mean the main *ede* of a sentence.

not necessarily be to neglect this knowledge base. The main reason for the absence of any instance of that type is that the frames in the current version of FrameNet are very specif c so that terms, such as *to sell* and *to buy*, which are very closely related are not part of the same frame. Since frames, however, are organized in a frame hierarchy, one should be able to establish relations via common abstract ancestor frames. In our current example, one could establish a semantic similarity between *to sell* which evokes the frame *Commerce_sell* and *to buy* which evokes the frame *Commerce_buy* by considering their (abstract) mother frame *Commerce*. One requires a reasoning algorithm for frame relations in order to make effectively use of FrameNet for establishing event alignments. I imagine that a tool similar to WordNet::Similarity (mentioned in Chapter 3.2.7) which computes a similarity score by calculating the path of two specif c frames within FrameNet might be successfully used for the current task. Unfortunately, such software does not yet exist. Such a tool might also cover a substantial part of the following alignment type which I called *intermediate textual entailment*. I assigned all those alignments to this type which are more complex than its preceding type but should be covered by shallow methods normally applied in *Recognizing Textual Entailment* (Dagan, Glickman, & Magnini, 2005). An example for this is the alignment of the two pairs *graduate (from a academic institution)* and *attend (an academic institution)*. Given its size, i.e. 16.45%, this is a type which should be dealt with in forthcoming implementations of event-based QA. However, one should point out that these are fairly diff cult cases which often require word sense disambiguation. Another diff culty is posed by *multi-word expressions*, such as *to lose one's life*, which are a subset of this type.

Next comes the most complex type. These are event alignments which require very deep semantic processing, such as a transformation to *First Order Logic* and the subsequent usage of theorem provers. Actually, I do not really think that such alignments should be dealt with in the near future, since this class is fairly small and there are other types which cover more instances and should be easier to tackle. One of the most diff cult examples I found was the following question-answer pair:

(4.41) Name some of Sosa's competitors.

(4.42) And yet, in true patriotic fashion, the Dominicans also wonder why McGwire has received so much more media coverage than Sosa, even though they both have been in hot pursuit of the home run record.

In order to f nd the correct answer from this answer sentence, one must be able to infer that *McGwire* is a competitor of *Sosa*, since they both pursue the home run record.

The f nal type is an extra type since it is fundamentally different from all previous types. It describes cases in which the *ede* is not realized. An example is the following question-answer pair:

(4.43) Which famous book did Rachel Carson write?

Type of Event Alignment	Frequency	Percentage
Identical Lemmas	125	40.32
Simple Synonymy	35	11.29
Complex Synonymy	3	0.97
Identical Frames (from FrameNet)	0	0
Intermediate Textual Entailment	51	16.45
Hard Textual Entailment	13	4.19
Elliptic Event Structures (with Null *edes*)	83	26.8

Table 4.7: Distribution of the Different Types of Event Alignments between Event Questions and Candidate Answer Sentences.

(4.44) The most famous book by Rachel Carson, *Silent Spring*, caused the banning of DDT.

The *write* event is not present in the answer sentence but implicitly evoked by one of its participants, i.e. the object *book*. The event-based method proposed in this thesis cannot cope with this type of ellipsis since the *ede* is the anchor for an event[31]
.
Though such deletions occur in 25% of the alignments, the problem is not that serious since in many cases the questions evoke multiple events. Question (4.47) contains, for example, the two *edes conference* and *take place*:

(4.47) Where did the $[\text{conference}]_{ede}$ $[\text{take place}]_{ede}$?

In a relevant answer sentence, the second *ede* is unlikely to appear but the first *ede*, which can only be recognized as an *ede* since I have included the recognition of nominalizations in my method, is very likely to re-appear.
All in all, the event-based method proposed is far from complete, as the evaluation of the alignments shows. There is still considerable work to be done in order to tackle textual entailment and elliptic event structures.

[31] Some readers may now object that according to the definition given in Chapter 2.4 Question (4.43) is no event question since the answer sentences for this question cannot be matched on the basis of event structure. However, this is not really true. This would be the consequence if all answer sentences for a question would not contain a similar event structure. Questions, such as (4.45), are clearly no event questions, since there is no predicate-argument structure evoked by this question.

(4.45) Where is Port Arthur?

So, one can say, in advance, that event structure will not be useful for finding an answer for this questions. In Question (4.43), however, the situation is different since the question underlies an event structure which might be reflected by answer sentences, such as (4.46), which should be unproblematic for my proposed answer extraction algorithm, since nominalizations, such as *author*, can be mapped onto verbs, such as *write*.

(4.46) Rachel Carson, $[\text{author}]_{ede}$ of *Silent Spring*, died in 1964.

4.6.2 The Different Syntactic Relations between Answer Constituent and Main Ede

This section presents the distribution of the different syntactic relations between answer constituent and main *ede*. The purpose of this evaluation is to assess what type of syntactic processing is really needed for event-based QA. (Please note that some of the syntactic terms mentioned below are explained in Appendix B.) I distinguish four different types. The first type describes the absence of any immediate syntactic relation between answer constituent and *ede*. An entity qualifies for a direct syntactic relation if it is either complement or adjunct of the *ede*. Often, there is some indirect relation but it cannot be elicited by means of the kind of processing done in my implementation. Two predominant types of indirect relations are the answer constituent being a modifier instead of the head of a complex NP[32] (as in Question-Answer Pair (4.48)-(4.49)) or the antecedent of a relative pronoun, i.e. the *ede* is directly related to a relative pronoun but the answer constituent is the antecedent and therefore cannot be reached without resolving the relative pronoun, which is not supported in the current implementation (see also Question-Answer Pair (4.50)-(4.51)).

(4.48) What country offered aid for the victims of Hurricane Mitch?

(4.49) [[Russia]$_{answer\ constituent}$'s cash-starved government]$_{SUBJ}$ is [offering]$_{ede}$ financial aid to Central American countries devastated by Hurricane Mitch, a news report said on Wednesday.

(4.50) Who won the Nobel Prize in Literature in 1988?

(4.51) [Naguib Mahfouz]$_{answer\ constituent}$, [who]$_{SUBJ}$ [won]$_{ede}$ the Nobel Prize for Literature in 1988, was born in the Gamaliya quarter of Cairo.

Almost 30% of the answer constituents are not directly related to the main *ede*. This means that there are clear limits for syntactic processing in event-based QA. One should also point out that these 30% are even more problematic for semantic processing, such as labelling of semantic roles, for example, via FrameNet since state-of-the-art tools which automatically assign these roles can only operate reliably within a subcategorization frame (note that such a frame only expresses direct syntactic relations). Since the assignment of semantic roles is more complicated than the syntactic processing, such as determining subcategorization frames, one should expect less than the remaining 70% to be labelled with semantic roles.
The distribution of the first type already justifies why I have made use of shallow metrics, such as textual proximity between answer constituent and *ede*, and very shallow semantic tagging, such as NE tagging. In my implementation, all answer constituents belonging to the first type of Figure 4.8 can only be recognized on the basis of these metrics.

[32] Note that grammatical functions are only assigned to the head of an NP.

Type of Syntactic Relation	Percentage
No (Direct) Syntactic Argument	28.22
Syntactic Argument without Grammatical Function	42.33
Syntactic Argument with Grammatical Function	26.38
Syntactic Argument via Controlling or Raising	3.06

Table 4.8: Syntactic Relations between Answer Constituent and Main Ede.

The remaining 70% of the answer constituents are directly related to the main *ede* but one should divide these cases in further subtypes. More than half of them and 42% of the total amount of candidate answers are directly syntactically related but do not really possess a (strong) grammatical function, i.e. subject, object or indirect object. These are mainly answers to temporal or locative questions. Such answer constituents are mainly realized as adjuncts. Such recognition should, therefore, not require too expensive syntactic processing.
The third type is the most important for the kind of syntactic processing that is performed in my implementation since it describes the kind of syntactic arguments which are either subjects, objects or indirect objects of the main *ede*. The size of 26% justifies some extra processing for this class. This is, in particular, true since other metrics, such as NE tagging or orientation to the *ede*, are lacking robustness in these cases. (Subject and object may swap their positions due to active-passive alternation and the semantic tags of the two types can be identical in many situations). 23% instances of this type involve a nominalized main *ede*. This justifies the usage of recognizing subcategorization frames for nominalized predicates (by using NOMLEX).
The last type could also have been subsumed by the first type since this syntactic relation is very difficult to recognize and, therefore, it is not supported by my implementation. This means that these instances are treated as those instances of the first type. I listed cases of *controlling* and *raising* in a separate type since, unlike the cases associated in the first type, they are supported by some state-of-the-art parsers, such as MINIPAR[33]. With only 3%, however, I consider the occurrence of this type too rare to be modelled.
One can conclude from this statistics that the usage of both syntactic processing like the one proposed in my implementation but also the inclusion of some shallow metrics, such as spatial similarity or orientation to the *ede*, is appropriate.

4.6.3 The Role of Spatial Distance

As already mentioned above, my implementation does not solely consider syntactic and semantic aspects but also incorporates surface-based metrics. One important metric is the spatial distance (textual proximity) between answer constituent and

[33] With this tool all syntactic arguments within such a construction should be recognized properly.

ede. The further away a potential answer constituent is from an *ede* in a document, the more unlikely this constituent is the answer. I examined the annotated relevant answer sentences in order to f nd some evidence for this behaviour in the current dataset. For this task, I drew a histogram (see also Figure 4.12). It clearly substantiates the claim that answer constituents have to be in the immediate vicinity of the *ede*. Most answer constituents are two words apart from the *ede*. The most frequent distance is not 1 since I calculated the distances between *ede* and head of the answer constituent. If one considers that most answer constituents are either NPs and PPs at the right of the *ede*, one notices that determiners and prepositions separate the semantic head of the answer constituent and its *ede*.
Notice that there are cases in which the *ede* is also the answer constituent. One example is the following question-answer pair:

(4.52) How many visitors does Longwood Gardens get per year?

(4.53) This 1,050 - acre horticultural showplace attracts more than 800,000 visitors a year, which means visitors should prepare for overcrowding on holidays and special weekends.

Normally, these are questions asking for some numeric expressions and the (main) predicate is a nominalization (denoting a person) which means that it can function both as an event and as a participating entity of this event.
Of course, the insight that the relation between answer constituent and *ede* is fairly local, challenges the usage of syntactic parsing. One of its benef ts is that it can establish long-range dependencies, as in:

(4.54) [Dutta]$_{answer\ constituent}$, who holds a master's degree in communications, [won]$_{ede}$ out over 78 other contestants.

Given the statistics in Figure 4.12, such cases are very rare as far as answer constituents are concerned. Though I make use of some syntactic processing in my implementation which should establish such dependencies, this is still some good news because the performance of correctly recognizing such long-range dependencies is fairly low. (Hence, the low performance should not affect the overall result since long-range dependencies are not that prominent.) I encountered various cases in which syntactic parsing failed. A typical example is illustrated in Sentence (4.55), where a part of the apposition, i.e. *P.R.*, is wrongly interpreted as the entire subject NP:

(4.55) [Rafael Celestino Benitez]$_{answer\ constituent}$, a native of Juncos, P.R., [graduated]$_{ede}$ from the U.S. Naval Academy at Annapolis in 1939.

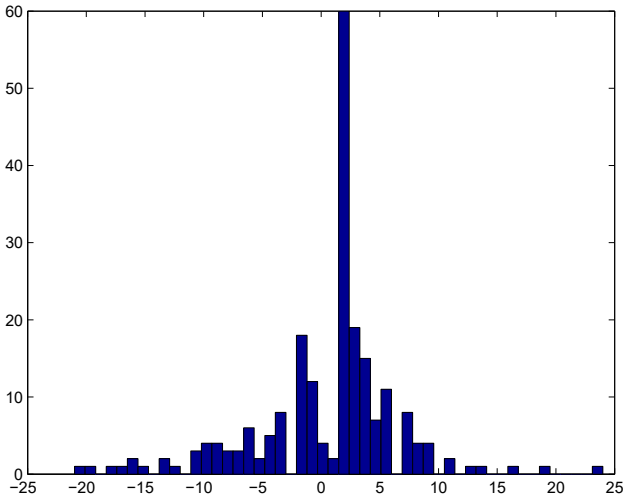

Figure 4.12: Histogram of the Spatial Distances between Main Ede and Answer Constituent in Candidate Answer Sentences.
(Negative distances denote answer constituents left from the main EDE whereas positive distances denote answer constituents right from the main EDE.)

Chapter 5

Evaluation

This chapter describes the two types of evaluations carried out in order to assess the performance of my method. The first evaluation is on artificial data and the second uses the output of components of a real existing QA system. Both evaluation methods are necessary since the different circumstances that accompany the two scenarios cover two different aspects. The first method gives an estimate of the potential of the proposed method, i.e. it tells what the system can cope with under ideal circumstances, whereas the second tells how good it performs in real life, taking erroneous processing of external auxiliary components into consideration. Since the method I propose for answer extraction relies on the output of two other components, namely question classification[1] and sentence retrieval, there can be a significant gap between the two scenarios depending on the performance of these two external components.

5.1 Scenario I: Testing on Artificial Data

This section describes the performance on artificial data, namely a set of question-answer pairs which has been manually extracted along the data which were used for data optimization (see also Chapter 4.5.1)[2]. Both the capability to recognize relevant candidate answer sentences concerning a question (see Table 5.1) and the extraction of answer snippets was assessed in this scenario. Unlike a real QA scenario, the ranking of candidate answer sentences was neglected[3]. In total, there were 376 relevant and 775 irrelevant question-answer pairs[4]. Evaluation on answer extraction was performed in four different runs. Run 1 is exactly the algorithm ex-

[1] By this I mean the general question type classification and not the classification of questions into event and non-event questions.

[2] Recall that from 189 questions I only used 100 questions for training. The remaining 89 questions were used for this test scenario.

[3] This aspect has been assessed in scenario II.

[4] The fact that the size of the relevant and irrelevant pairs is not equal should reflect that, usually, the number of irrelevant answer sentences is significantly higher than the number of relevant answer sentences.

plained in the previous chapter. Run 2 employed a post-filter comprising a list of stopwords which cannot be answers (such as pronouns, auxiliaries and conjunctions) no matter what the specific question looks like. This filter was employed since, occasionally, such answers were returned, mostly due to parsing errors. In Run 3, this filter is extended by removing all those answers which contained exactly terms from the question. Thus, wrong answers to Question (5.1), such as Answer (5.2), can be excluded:

(5.1) Who was Sosa's competitor for the home run title in 1998?

(5.2) *Sosa

Of course, a perfect data-driven model would directly incorporate such heuristics that I encoded as an external post-filter as features. Unfortunately, I did not realize the necessity of such heuristics until I had almost completely implemented my module.
Run 4 is identical to Run 3 as far as post-filtering is concerned. However, I made a slight change to the model, namely to Equation 4.12 in Chapter 4.4.2 repeated below:

$$val(qArgMap) := \sigma\left(\sum_{i=1}^{l} w^i_{qArgMap} \cdot f^i_{qArgMap} + b_{qArgMap}\right) \tag{5.1}$$

This equation states how the quality of a mapping between question constituent and potential answer constituent is measured. The drawback of this equation is that it is a global formula which does not consider the different properties of different event question subtypes. In order to rectify this, I defined the modified Equation 5.2 which is dependent on a particular event question type, i.e. $qType$:

$$val(qArgMap|qType) := \sigma\left(\sum_{i=1}^{l} w^i_{qArgMap}(qType) \cdot f^i_{qArgMap} + b_{qArgMap}(qType)\right) \tag{5.2}$$

However, since the data for training are very sparse I could not estimate weights for every specific type[5]. That is why I restricted $qType$ to the four most general event question types being *locative questions*, *temporal questions*, *numerical questions*[6] and *propositional questions*[7]. This modification rests on the intuition that the weights of the different features of *qArgMap* are likely to differ with respect to an

[5]These event question types are listed in Appendix C.

[6]These are mostly questions asking for quantities.

[7]These are basically all those event questions which do not belong to either of the three previous event question types.

Recall	Precision	F-Score
0.825	0.713	0.765

Table 5.1: Performance of Relevance Detection in Scenario I.

event question type. For example, in case of a temporal question, semantic information, in particular, the output of a NE tagger mostly suff ces in order to extract the correct answer snippet from a relevant candidate answer sentence. In case of propositional questions, such as Question (5.1), other features, in particular syntactic features may be most informative.

Apart from the four different runs I also assessed the correctness of a returned answer snippet in two different ways. A strict criterion demands that the answer snippet must exactly match the regular expression of (*TREC Answer Patterns*, 2005) and a more lenient criterion only demands that the regular expression must match a substring of the returned answer. I also evaluated due to latter criterion since I wanted to have an idea how many almost correct answers my method produced.

Table 5.1 displays the performance of relevance detection[8] whereas Table 5.2 displays the performance of answer extraction.[9] The F-score of relevance detection at about 0.765 is satisfactory but this task is far easier than the task of answer extraction. The results of the answer extraction are fairly promising with the best F-score of 0.405 but one should keep in mind that this evaluation has been carried out on artif cial data. The usage of a post-f lter improves the performance although the mere inclusion of stopwords, i.e. Run 2, has little effect on the overall result. The improvement by merely 0.015 from Run 3 to Run 4 gained by using question type dependent answer extraction is a bit disappointing since this modif cation should increase the expressivity of the model. I suspect that the sparse data are responsible for this minor improvement. The recall in Run 1 to Run 3 remains the same. This tendency is fairly natural since a f lter is restrictive. Therefore, fewer false positives are returned (this corresponds to a higher precision). Since false positives are neglected in recall, f ltering should not affect this measure. The amount of false negatives should remain the same.

There is a considerable gap between the lenient evaluation and the strict evaluation. The strict evaluation loses a third of the F-score that matched in the lenient evaluation in Run 4. This implies that among the false answer snippets returned there is a considerable proportion of inexact answer snippets. This is not unusual for answer extraction. It also suggests that there is still some room for improvement which might be achieved by not too expensive modif cations of the original design. After all, almost correct answers are easier to cope with than totally incorrect answers.

[8] By this the relevance of an answer sentence with regard to the question is meant.

[9] Note that for a detailed evaluation more experiments than just the four runs would have to be expected since there are more ways how to combine these methods. However, due to the limited time that was available, I had to conf ne my evaluation to these four runs.

Criterion	Measure	Run 1	Run 2	Run 3	Run 4
lenient	Recall	0.740	0.740	0.740	0.754
	Precision	0.433	0.435	0.473	0.509
	F-Score	0.546	0.548	0.577	**0.608**
strict	Recall	0.616	0.616	0.616	0.641
	Precision	0.244	0.250	0.285	0.296
	F-Score	0.349	0.356	0.390	**0.405**

Table 5.2: Performance of Answer Extraction in Scenario I.

5.2 Scenario II: Testing on Real-Life Data

The more important evaluation is presented in this section. It discusses the performance of my method applied on the output of retrieved answer sentences using the TREC system Alyssa (Shen et al., 2006) and QTile (Leidner et al., 2004). How these sentences have been obtained is explained in Chapter 4.3. The evaluation was carried out on the same questions on which scenario I was evaluated. One should keep in mind that these questions are exclusively genuine event questions.
For each of these questions QTile returns a list of up to 15 candidate answer sentences. In some cases, this list can be shorter, when the query, which was formulated from the question, only matched fewer sentences. This ranked list is re-ranked by my method, i.e. by $val(qaMap)$ (see also Equation 4.5 in Section 4.4.2). Answer extraction will only be evaluated by checking the answer snippet extracted from the most highly re-ranked (relevant) candidate sentence. Table 5.3 displays the performance of detecting relevant answer sentences (concerning questions) whereas Table 5.4 displays the performance of answer extraction. In general, there is massive drop in performance if compared with the results of scenario I. The F-score of relevance detection drops by 48% (from 0.765 to 0.400) and similarly does the best F-score of answer extraction (from 0.405 to 0.190). Of course, the comparison of those values is controversial since they are tested in different scenarios but this loss in quality is still striking. Unfortunately, augmenting the basic answer extraction algorithm by additional f lters or question type dependent extraction did not improve the overall result. The two f lters applied in Run 2 and Run 3 have exactly the same F-score of Run 1. Precision in Run 3 is a slightly better but this goes at the expense of a deteriorating recall, so nothing is gained. According to the previous section, the recall should not be affected by the f lters. Apparently, the f lter is too restrictive, i.e. it f lters not only false positives but also some true positives.
The lacking changes in performance in the f rst three runs might be ascribed to the very limited available test data. Whereas in scenario I, answer extraction has been done on each relevant candidate answer sentence for a question, in scenario

II only the most highly ranked relevant candidate sentence has been considered[10]. Thus, scenario I comprised approximately 15 times more test instances for evaluation than scenario II. Since there are fewer testing instances in scenario II, minor changes to the answer extraction algorithm, as they are implemented in Run 2 and Run 3, cannot really be measured since eligible instances are missing.

The decrease of the strict F-score in Run 4 by more than 50% is a very surprising result. The question type dependent answer extraction was designed to produce a more expressive model, so one would not expect such a drastic drop in performance; after all the performance could be improved on the data used in scenario I. I suspect that this result may be ascribed to some form of overf tting. The test data in scenario I are far more similar to the training data than the test data used in scenario II, so that the lacking generalization of the model had only a signif cant impact on the test data in scenario II.

As in the previous evaluation, there is a considerable amount of inexact answers. One fourth of the correct answers in the lenient evaluation are no exact answers (that is a bit less than in scenario I).

The performance of the external components used in this evaluation is as follows: according to (Shen et al., 2006), the accuracy of the question type classif er used in Alyssa is 80.8%. The retrieval component in combination with QTile has a severe loss in actual answer sentences of about 46.67%. This means that the sentence retrieval can only return a set of potential answer sentences with at least one actual answer sentence in just more than 53% of the questions posed. The loss of answer sentences in Alyssa[11] is signif cantly smaller but this is due to the fact that far more answer sentences are used for answer extraction. At TREC 2006, Alyssa used 60 sentences for answer extraction whereas I use one fourth of this amount, i.e. 15 sentences. (As already explained, I cannot raise that number since processing, especially syntactic processing, would take too long.) If one only considered the top 15 sentences of Alyssa's sentence retrieval, the loss of data would be slightly better than QTile, i.e. the loss could be dropped by approximately 7%.

In order to fully assess my method one would have to compare the performance of a state-of-the-art QA system, such as Alyssa, on event questions (in the ideal case these would be exactly those questions which I used in my evaluation). Unfortunately, those data were not available to me. The overall F-score of Alyssa of 0.191 as stated in (Shen et al., 2006) should not be used since they do not ref ect the performance on event questions. Neither would it be appropriate to measure my event-based answer extraction module on all those questions on which Alyssa has been evaluated, since the method proposed in this thesis is no complete QA system. It concerns exclusively event questions. Therefore, the performance on other questions is not that important. After all, the implementation described in

[10] Note that I chose these two evaluations in order to show that my method can be used both for classif cation and ranking.

[11] Alyssa incorporates a newly designed sentence retrieval module which I could not use because it was not available at the time of implementing my answer extraction module. Therefore, I had to use QTile.

Recall	Precision	F-Score
0.944	0.254	0.400

Table 5.3: Performance of Relevance Detection in Scenario II.

Criterion	Measure	Run 1	Run 2	Run 3	Run 4
lenient	Recall	0.550	0.550	0.478	0.333
	Precision	0.164	0.164	0.172	0.117
	F-Score	0.253	0.253	0.253	0.173
strict	Recall	0.471	0.471	0.400	0.177
	Precision	0.119	0.119	0.125	0.050
	F-Score	0.190	0.190	0.190	0.078

Table 5.4: Performance of Answer Extraction in Scenario II.

this thesis should be regarded as a plug-in for existent QA systems to improve their performance on event questions and not as their competitor.

Chapter 6

Discussion

This section discusses the results of the evaluation. I will only focus on answer extraction and will not broach the issue of relevance detection of answer sentences since this is only an intermediate step. Having looked at a large amount of examples, I did not find any cases in which relevance detection of answer sentences worked in some unexpected way. After all, the performance of this subtask, as described in Chapter 5, was far better than that of answer extraction so the latter should also be the centre of the following discussion.
First, I will have a look at some examples[1] in order to illustrate the results of this implementation. Then, I will comment on drawbacks that I could make out which are responsible for misclassifications. Finally, I will discuss the integration of my method in state-of-the-art systems, such as Alyssa, by pointing to the benefits and problems that this integration might entail.

6.1 Some Examples

In general, temporal and locative questions are easier to tackle than other propositional questions. Therefore, I am, in particular, interested in how my implementation dealt with the latter types. The success of my model can be mainly ascribed to the fact that many different kinds of information have been combined to a uniform model. In Question-Answer Pair (6.1)-(6.2), for example, the *ede* changes its part of speech. (Matching the *edes* is no problem since I use NOMLEX.).

(6.1) Who manufactures Viagra?

(6.2) It quotes a State Drug Administration official as saying that the results will partly decide whether Pfizer, the American manufacturer of Viagra, is allowed to sell the little blue pill in China.

A complete reliance on syntactic information would be insufficient in this case. With NOMLEX one can match the object, i.e. *Viagra*, but would run into trouble

[1] Note that most examples were taken from scenario I since this evaluation involved the answer extraction of more answer sentences than scenario II.

when it comes to matching the subject of the question. Unfortunately, there is no immediate syntactic relationship between *manufacturer* and the answer *Pfizer*. However, my model also considers semantic and spatial information. *Pfizer* is an *organization* should be compatible with question type *person*[2]. It is very near the *ede* and it has also the correct orientation. These factors are all considered and that is why my module returns the correct answer, i.e. *Pfizer*. Thus, my method is fairly robust against syntactic variation as far as not only the relationship between answer constituent and *ede* but also other semantic information are strong. Even abbreviated relative clauses could be dealt with as exemplified by the following question-answer pair:

(6.3) How many students were wounded?

(6.4) Prosecutors have recommended 7 1/2 years for each attempted murder count for the 25 students Kip wounded and a detective he attacked with a knife.

Of course, one could argue that a simple term-based approach might also produce this answer. My method can, however, additionally consult syntactic information in case the terms are not sufficiently similar or the distance between *ede* and answer constituent in the candidate answer sentence is not local. Syntactic information can be vital in the following question-answer pair:

(6.5) Who won the crown?

(6.6) Dutta, who holds a master's degree in communications, won out over 78 other contestants.

A term-based approach is always very locally confined and could not bridge the intervening relative clause.
It is not that difficult to construct an algorithm which can find Question-Answer Pair (6.1)-(6.2) and another algorithm which can find Question-Answer Pair (6.5)-(6.6), but a model which covers both pairs by combining both term-based and structural information, is more of a challenge. Fortunately, my method achieves this to a certain extent. However, it has also its limitations. The greater the distance between *ede* and answer constituent the less likely it is to be found[3]. Question-Answer Pair (6.5)-(6.6) is a case where this works but Question-Answer Pair (6.7)-(6.8) is a case where my program failed.

(6.7) List students who were shot by Kip Kinkel.

(6.8) Richard Peek Jr. 19, who was wounded in one arm in the bloodshed at Thurston High School, was shot in the head while hunting deer with his 17-year-old brother, Robert, said Lane County sheriff's Sgt. Byron Trapp.

[2]Note that this semantic class contains organizations.
[3]Often, this is caused by erroneous parsing of long-range dependencies.

When the distance is great my module is also susceptible to interfering entities. For example, in Question-Answer Pair (6.5)-(6.9) my program returns *Mpule Kwelagobe* instead of *Lara Dutta*.

(6.9) Lara Dutta succeeded Mpule Kwelagobe of Botswana to become the 49th winner of the pageant and the first Miss Universe of the new millennium.

This is due to the fact that the features of my model favour *Mpule Kwelagobe*. It is nearer to the *winner* event and contains the correct semantic type *person*. More sophisticated grammatical relation modelling would be needed in order to extract the right answer in this example.
The most complex question-answer pair that has been analysed correctly is:

(6.10) Who did Foreman defeat for his first heavyweight championship?

(6.11) In George Foreman's first boxing incarnation, he was a lean, snarling ogre who mauled Joe Frazier to win the heavyweight championship 27 years ago.

It is a surprise that the correct answer could be found though my model can only recognize the underlying syntactic relationships to a certain extent. This example also suggests that synonyms can be recognized.
One should not expect to encounter many of such question-answer pairs as the one stated above. One cannot even guarantee that very simple cases are correctly processed. For example, my module did not manage to extract *Lookheed Martin* from the following question-answer pair:

(6.12) Who manufactures F-16?

(6.13) The F-16 aircraft are manufactured by Lockheed Martin but the engines and some components are made by Pratt & Whitney and General Electric.

This example shows how instable this processing is. Some incorrect POS tag or some incorrect parse tree may be responsible for returning wrong answers.
As already predicted in Chapter 4.6.1, no question-answer pairs requiring some form of *Intermediate Textual Entailment*, such as *to win (a beauty pageant)* → *to be crowned (in a beauty pageant)* or *to lose (one's life)* → *to die*, could be matched. This means that any textual entailment beyond synonymy, such as *to witness* → *to see*, *win* → *victory* or *to write* → *author*, cannot be performed with my module yet. (Sekine, 2006) confirms that textual entailment on the basis on WordNet alone, as it is also done in my module, is insufficient. I assume that in order to increase the coverage one would have to employ more sophisticated processing, e.g. paraphrase acquisition in the fashion of (D. Lin & Pantel, 2001) or (Hasegawa, Sekine, & Grishman, 2005). These methods would have to take context information into account.
In general, the origins for misclassifications are fairly variable. Apart from some structural drawbacks, which will be discussed in detail in the forthcoming section,

the mistakes mostly derive from some erroneous processing of the NLP tools that I have used in my program. As far as parsing is concerned, either the scope or label of a phrase is incorrect or the constituents are incorrectly attached in the parse tree. Erroneous parse trees have an impact on many important features, such as `frame`, `gram` or `argstat` (see Appendix D for an explanation of these features). Another problem is the limited coverage of NE tagging. As already stated in Chapter 4.6.2, many answer constituents are not directly syntactically related to the *ede*. Some of the previous examples, such as Sentences (6.2), (6.4), or (6.11), substantiate that claim. In these cases, one major source of information is the semantic content. Since in many questions, a specif c entity has to be recognized, WordNet cannot contribute any information and one has to rely on NE tagging. The major problem is, however, that state-of-the-art NE taggers only identify few types of entities (see also Chapter 3.2.3). This set of entities is often insuff cient when, for example, the name of a f lm or a medical product is being looked for.

6.2 Conceptual Drawbacks

One could argue that the model presented in this thesis lacks expressive power. The features which are weighted in logistic regression are combined in a linear fashion[4]. The result of this is that the decision boundaries generated by my model can only be linear. This is an idealized assumption which may not correspond to reality. But models learning more complex decision boundaries like decision trees or support vector machines are less robust against overf tting. Since the training data I acquired are very sparse the usage of those complex classif ers would require an increase in the size of the labelled training data. If this is not possible one might also check the current feature set and perform some form of *feature selection* in order to increase the performance of the current classif er. An increase in performance might be possible if the current feature set contains some very noisy features or the size of the feature set is already too large (given the size of the training data) so that some form of overf tting occurs.
Though the features I selected are useful indicators, they do not always suff ce for robust answer extraction. This, in particular, applies to the features used in *qArgMap*. (For more information about these features please go to Appendix D.2.) In a considerable number of questions, only very few features return an output. This phenomenon affects questions in which there is hardly any information in the question constituent, such as:

(6.14) [Where] was George Foreman born?

The only information that can be deduced from this question is that the answer must be a location. The situation is different in Question (6.15) where one can use the phrase label *PP*, the preposition *in* and the semantic content *sea* of the question constituent for answer extraction.

[4] One can also def ne logistic regression as a linear regression embedded into a sigmoid function.

(6.15) [In what sea] did the submarine sink?

This implies, that in order to extract an answer from a question the answer extraction algorithm should not exclusively rely on the information inherent in the question to be processed. Additionally, one should have information (i.e. features) associated with each (event) question type to augment the information present in the question. For Question (6.14) this would mean that a phrase like $PP_{[in]}$ might be an answer constituent.
Some global constraints should be included for answer extraction as well. The evaluation in Chapter 5 has shown that there are such constraints imposed upon a well-formed answer snippet. It would add to the uniformity of an answer extraction algorithm, however, to include these constraints as features in the overall model rather than writing a post-f lter (as I have implemented it).
Another structural problem is that not all features are optimally *orthogonalized*. An obvious example is `semIII` where the output of WordNet Lexicographer Files and the NE tagger is merged. (The feature `semI` is similarly affected.) A better solution would use two separate features for these types of information. From a superf cial perspective these two sources work complementary. But there are cases in which they conf ict with each other. This particularly affects `semIII` because this is an important feature used in *qArgMap* and thus participating in extracting an answer snippet from a candidate answer sentence. The semantic classif cation of WordNet can only be established for common nouns, i.e. sets of entities, which would be useful in the following question-answer pair:

(6.16) How many *people* died in the accident?

(6.17) All 118 *crewmen* lost their lives.

NE tagging, on the other hand, tags, as the name says, individual entities. This would be useful in the following question-answer pair:

(6.18) *Who* won the contest?

(6.19) *Dutta* won out over 78 other contestants.

However, in Question-Answer Pair (6.20)-(6.21), without knowing whether one looks for an entity or a group of entities, there might be two candidates in the answer sentence, namely, *book* and *Silent Spring*:

(6.20) What *book* did Rachel Carson write?

(6.21) Rachel Carson wrote her famous *book* called *Silent Spring* in 1962.

Unfortunately, due to the spatial proximity and the syntactic relatedness the false candidate is preferred in my model. Examples like (6.20)-(6.21) are fairly frequent which means that this lacking distinction between individual entities and groups of entities has caused some signif cant amount of misclassif cations in answer extraction. A better algorithm would have to derive from a question whether a common

noun or a proper noun is the answer constituent and focus on the corresponding semantic feature during answer extraction.
Another drawback of my method is the way in which questions with multiple *edes* are tackled. According to Equation 4.14 in Chapter 4.5.2 repeated in Equation 6.1, only the strongest $\hat{esMap}$, i.e. the strongest mapping of all optimal event structures mappings[5], is considered for the f nal evaluation of *qaMap*[6].

$$val(qaMap) := \alpha \cdot \max_i \left(val \left(\hat{esMap_i} \right) \right) + (1 - \alpha) \cdot irMatch \tag{6.1}$$

This is necessary since, often, not all *edes* (and therefore event structures) re-appear in the answer. A typical example is illustrated in the following question-answer pair:

(6.22) What [caused]$_{ede}$ the [death]$_{ede}$ of Sani Abacha?

(6.23) Nigerian military ruler General Sani Abacha, aged 54, [died]$_{ede}$ of a heart attack on Monday.

On the other hand, there are cases in which a more restrictive approach would be needed. Imagine, for example, Question (6.24) where two *edes* are evoked:

(6.24) How many [students]$_{ede}$ were [wounded]$_{ede}$?

According to Equation 6.1, a candidate answer sentence only containing the *student* event and not the *wound* event could suff ciently match in order to classify the candidate answer sentence as relevant. A simple alternative to Equation 6.1 which would solve this problem is Equation 6.2 but this solution causes a signif cant drop in recall when it comes to cases, such as Question (6.22).

$$val(qaMap) := \alpha \cdot \frac{1}{n} \sum_i^n val \left(\hat{esMap_i} \right) + (1 - \alpha) \cdot irMatch \tag{6.2}$$

So both options are not ideal. A more sophisticated solution would not average the event structures but weight them due to the strength of their corresponding *edes* appearing in the question to denote (real) events. Such an approach would give *wounded* a high and *student* a low weight, so that each relevant answer sentence for Question (6.24) would have to contain a *wound* event.

Finally, the success of a syntactically motivated event structure should also be brief y discussed. As already mentioned in the feature discussion in Chapter 4.5.3, some expensive syntactic features, such as `gram` or `argstat` (see also Appendix D.2), are not very strong features. Two reasons might be responsible. Firstly, the tools which are responsible for these features are not suff ciently reliable. Secondly, these features are not that important for QA as previously assumed.

[5]Note that I assume that a question may contain more than one event.
[6]i.e. the mapping between the entire question and the entire candidate answer sentence

Both reasons are true to some extent. In general, one should not put too much emphasis on syntactic features in QA. Grammatical relations are only useful if their recognition is really reliable. Otherwise, spatial metrics, such as `dist` and `ori`, are a good alternative. These are not linguistic features but they can express a fair amount of those syntactic features. Additionally, they are far more efficient. Using a match of subcategorization frames in questions and answer sentences, as expressed in feature `argstat`, is not always beneficial. Due to the considerable syntactic variability between event questions and their corresponding answer sentences, subcategorization frames are rarely constant. This insight is also crucial for the labelling of semantic roles, because, in general, subcategorization frames are the starting point of further semantic processing.

6.3 Integration in State-of-the-art Systems

The evaluation of my method using components of state-of-the-art systems, as described in Chapter 5.2, is a surprise given the rather positive results on the artificial data in Chapter 5.1. Of course, a certain drop in performance is expected. In a QA system, answer extraction has to rely on the quality of other QA components. It was already suggested that the output of the retrieval component is mainly responsible for this. The set of returned candidate answer sentences often fails to include a relevant answer for a specific question.
Moreover, it might also be that the retrieval component of Alyssa is not very compatible with the event-based answer extraction algorithm presented in this thesis. This assumption is not too far-fetched if one considers the lacking uniformity of these two components. Recall that one of the benefits of my answer extraction is that not only lemma-identical words can be matched but also synonymous words (even across different parts of speech). Thus, one can match events, such as *home* and *to live* or *to win* and *victory*. This expressive power of my method can only be harnessed effectively, if such relations between words occur in question-answer pairs. Whether this is the case, depends on how potential answer sentences are extracted from the retrieval component of the QA system. If the query for this component happens in a simple lemma-restricted fashion, as it is the case with *Alyssa*, one cannot guarantee that relevant candidate answer sentences with synonymous expressions are found. For example, if the query for retrieval only contains the terms of the question, such as *to live* or *to win*, one cannot expect that candidate answer sentences are retrieved where these events are reflected by synonymous words, such as *home* and *victory*. This can only happen by using a query-expansion where synonymous terms are included.
Though the performance of my module when used with Alyssa's document retrieval and QTile might be fairly low, it could still be useful for the overall Alyssa system. Since my answer extraction algorithm is conceptually very different from the algorithms currently integrated in Alyssa (there is a dependency-based algorithm and a rule-based approach), my model may provide complementary infor-

mation. In future work, one should compare my method with the dependency-based approach[7] since these two components are both linguistically motivated and particularly useful for event questions. By using a mapping from nominalizations onto verbs the method proposed in this thesis should enable more predicates in question-answer pairs to be matched than the dependency-based approach. My model explicitly covers subcategorization frames in various features which, again, is not the case in the dependency-based algorithm. As far as semantics is concerned, both systems use WordNet. However, the method proposed in this thesis uses the complex algorithms provided by WordNet::Similarity (instead of directly checking WordNet-relations between two words) and, additionally, Lexicographer Files (which is not done in the other approach, either).

Due to these individual characteristics, I may say that my proposed model has its own *view* on a potential question-answer pair. This independent *view* is a useful property which qualifies this model for inclusion in Alyssa's fusion process where the *judgements* of the different answer extraction methods are considered and a final answer is computed. How much impact the inclusion of my method finally would have on the overall performance of Alyssa is, however, left for future work to decide.

Before my module could be included in a QA system, however, some technical problems would have to be solved. I already stated that Collins' parser performed poorly. I suspect that this performance cannot only be exclusively ascribed to the parser. QA much relies on processing questions. However, the corresponding training material, which is vital for statistic parsers, is very sparse. (Müller, 2004) discusses this issue and points out that only half percent of the Penn Treebank are full questions. An improvement of the performance of a parsing would, therefore, require treebanks comprising more questions than it is usually the case rather than changing the parser.

Apart from that, technical problems were encountered with various tools, in particular, TigerAPI and WordNet::Similarity. Currently, I only have software modules which work exclusively on the output of Collins' parser. The usage of another parser would, therefore, have a great impact on the overall architecture. The removal of TigerAPI is even more problematic since this is the only navigation tool for TigerXML. I would neither suggest to use a different format since TigerXML is ideal for my purposes. Similar reasons can be brought forward when it comes to WordNet::Similarity. It is the only tool available of its kind and its usage is vital for the overall performance, in particular, for *edeMap* and *qArgMap* (see also Chapter 4.5.3). To make it worse, TigerAPI and WordNet::Similarity are not very stable. For example, the command-line interface of WordNet::Similarity gets stuck after a certain amount of queries has been posed, which can only be avoided by re-loading the tool at regular intervals which is very time-consuming.

[7] i.e. the algorithms should be compared on the identical set of questions

Chapter 7

Summary, Contributions, Conclusions and Future Work

7.1 Summary

In this thesis, I have developed and implemented an event-based model for answer extraction in open-domain QA. The model ref ects both linguistic properties of events and insights gained by a descriptive analysis of the TREC 2005 question set. Practical and technical restrictions on the implementation meant that no semantic processing apart from using WordNet and NE tagging was possible.
The aim of the model was to use event structures in a QA scenario optimally so that those aspects are considered which cannot be covered by non-event-based methods. In order to make the overall model robust against syntactic variability some surface-based metrics were taken into consideration. All metrics were combined to a uniform model which used these different sources of information in a data-driven way.

7.2 Contributions

The novel contributions presented in this thesis comprise:

- **statistical analysis of relevant answers**
  The statistical analysis of relevant answer sentences of event questions in the TREC 2005 data (see Chapter 4.6.1) cannot only be used as a quantif cation of an upper bound of the performance of my implementation, but also guide future implementations of open-domain event-based QA, since these data offer detailed information as to the importance of syntactic, semantic and surface-based processing;

- **feature analysis**
  A data-driven feature analysis showed how individual features perform in event-based QA and how much unique information they encode (see Chapter 4.5.3);

- **new features**
  Among the features for the proposed answer extraction algorithm, semantic classes of *Lexiographer Files* of WordNet for semantic tagging have been used for the first time. Moreover, subcategorization information has been included in the feature set by using the two lexicons NOMLEX and COMLEX. NOMLEX could also be harnessed to model semantic similarities between terms across different kinds of parts of speech which are not necessarily lexically related e.g. *to live* and *home*;

- **cost-sensitive learning**
  In order to be able to build a robust classifier on heavily imbalanced data, some specifically designed form of cost-sensitive learning has been applied to answer extraction. To the best of my knowledge this is the first application of cost-sensitive learning to QA; and

- **sentence relevance detection**
  For the relevance detection of sentences I showed a successful way how to combine term-based and event-based matching (*qaMap*).

7.3 Conclusions

The evaluation of this implementation on artificial data proved that this model works to a certain extent. The notion of *ede* which is independent of part of speech thus allowing nominalizations to be mapped onto verbs and vice versa, is useful.
The data-driven approach for combining different features for answer extraction has shown that semantic features, such as a mapping from question types to WordNet Lexicographer files (`semIII`), and surface-based features, such as the distance from answer consituent to its *ede* (`dist`), are among the cheapest and most effective features used. Syntactic features performed poorly; only in one case (`argstat`) a reasonable precision could be achieved. This result challenges the usefulness of syntactically motivated event structures in open-domain QA. A final judgement, however, cannot made at this stage since better results might be achieved by using more robust parsing.
As far as relevance detection of answer sentences with regard to questions, i.e. *qaMap*, is concerned, I could show that a high recall term-based approach and a high precision event-based method can be combined in order to achieve a better performance than just the individual methods.

7.4 Future Work

The question in how far this implementation can be used in a real QA system could not be def nitely answered since no proper comparative evaluation could be carried out. For this purpose one would have to have access to the performance of a state-of-the-art QA system on the same event questions I used in my evaluation.
In order to use my module in such a QA system, a more robust retrieval component which incorporates some knowledge-based form of query-expansion might be a useful complement.
Furthermore, some technical problems of my implementation have to be solved. This concerns the speed of processing, stability and the performance of some NLP tools, in particular, the parser I used.
As far as the appropriateness of the model is concerned, a larger labelled training-set would be desirable in order to test more complex (non-linear) learning methods. More data would also allow the usage of a more robust form of question-type dependent answer extraction algorithm. Alternatively, some form of feature selection might also be benef cial.
In order to broaden the matching of events, i.e. *edes*, which are not lexically related or synonymous, some more advanced form of paraphrase detection is needed (i.e. either some sophisticated usage of FrameNet[1] or some unsupervised data-driven approach).
For a more reliable extraction of answer snippets from relevant candidate answer sentences better NE tagging which identif es more f ne-grained types would also be desirable. In contrast to semantic roles, they are less bound to syntactic information, which seem to be too variable to deal with sensibly.

[1] Note that I consider FrameNet for matching predicates and not as a means to recognize semantic roles.

Appendix A

Performance Issues When Running Shalmaneser on TREC 2005 Questions

Chapter 3.2.8 stated that the present version of *Shalmaneser* could not be used for the final implementation of my answer extraction module. There now follows a detailed description of what problems occurred and their potential causes. I tested the tool by tagging all TREC 2005 questions.

The results of this experiment were disappointing. Hardly any questions were tagged completely so that I did not even consider a proper quantitative analysis worthwhile. The tool performed badly on some crucial levels. Not only has the recall been low (many *fees* and *fes* have been overlooked) but also the labelling was seldom convincing. Often *fes* and *fee* were assigned to the same constituent which is rarely right.

By transforming the TREC 2005 questions manually into declarative sentences and then run the tool again I intended to find out whether the tool only performed so unsatisfactorily because of lacking training material for questions[1]. (It is a known fact that the corresponding training material for both syntactic and semantic parsing is sparse.) Contrary to my expectation, the performance of the final output was only slightly better than the results of the original run though the syntactic structures had changed to more familiar parse trees of ordinary declarative sentences.

I strongly assume that the performance was so low because of other idiosyncratic properties of these questions. The amount of *edes* being nominalizations, such as Questions (A.1)-(A.4), might be one crucial reason for the performance:

(A.1) Who was the killer?

(A.2) Who were on-ground witnesses to the accident?

[1] I transformed them manually in order to obtain plausible sentence structures. This would not necessarily be guaranteed if I transformed them automatically.

(A.3) What was the outcome of the U.S. trial against the pilot?

(A.4) Who was Sosa's competitor for the home run title in 1998?

According to (Erk & Padó, 2006) *Shalmaneser* cannot cope with these types of structures i.e. a correct assignment of frame structures to propositions evoked by these predicates is not possible.
I also noticed that nominalizations and verbs could cause conf icts for syntactic parsing, such as in Questions (A.5) and (A.6), i.e. nominalizations are considered verbs (and sometimes even vice versa):

(A.5) How much money did UPS pay out in insurance claims in 1984?

(A.6) When did he make his famous ride?

Apart from that many *imperative questions*, such as Sentences (A.7)-(A.9), had rarely a correct parse:

(A.7) Name products manufactured by Merck.

(A.8) Identify the nationalities of passengers on Flight 990.

(A.9) List other horses who won the Kentucky Derby and Preakness but not the Belmont.

Perhaps enhanced versions of the tool which have a larger coverage, in particular with regard to nominalizations, can yield better results. Research in the area of widening the coverage, as presented in (Burchardt, Erk, & Frank, 2005), seem to be promising methods in order to improve this tool. I assume, however, that in order to semantically tag nouns as frames further semantic training material has to be provided.

Appendix B

Syntax Glossary

This appendix explains some crucial syntactic terms. Note that for many of them there are no commonly accepted def nitions. The def nitions stated below mainly follow (Radford, 1997). Throughout this thesis, these syntactic terms are used according to the def nitions given in this appendix.

B.1 Subcategorization

Subcategorization is the division of lexical categories (e.g. nouns, verbs etc.) into *subcategories* motivated by both syntactic and semantic criteria in order to account for different dependency relations within a sentence. In this thesis, the term subcategorization is restricted to the obligatory syntactic frame of predicates (i.e. either full verbs or nominalizations). This is sometimes referred to as *strict subcategorization*. In Sentence (B.1), for example, the predicate *sent* subcategorizes the subject *Mary*, the direct object *the letter* and the indirect object *to Peter*. The set of all subcategorized arguments of a predicate is also referred to as *subcategorization frame*.

(B.1) [Mary]$_{NP}$ [*sent*]$_{V}$ [the letter]$_{NP}$ [to Peter]$_{PP}$.

In this thesis, a further technical restriction conf nes the set of subcategorized arguments to NPs and PPs.

B.2 Complement

If a predicate subcategorizes a syntactic argument that is obligatory, then this argument is referred to as a *complement*. Contrary to other def nitions, I include subject NPs to this set as well. In Sentence (B.2), the verb *hit* selects two complements being the subject *Peter* and the object *John*.

(B.2) [Peter]$_{NP}$ [hit]$_{V}$ [John]$_{NP}$.

Note that in case one of these complements is missing, such as in Sentence (B.3), the sentence is incomplete and therefore ungrammatical.

(B.3) *[Peter]$_{NP}$ [hit]$_{V}$.

B.3 Adjunct

In addition to complements, adjuncts are those arguments a predicate selects which are optional. In Sentence (B.4), the PP *in 1972* is an adjunct.

(B.4) [Richard Nixon]$_{NP}$ [visited]$_{V}$ [China]$_{NP}$ [in 1972]$_{PP}$.

If one removes this phrase, as in Sentence (B.5), the sentence is still well-formed.

(B.5) [Richard Nixon]$_{NP}$ [visited]$_{V}$ [China]$_{NP}$.

B.4 Satellite

Satellites are NPs and PPs which are in the vicinity of a predicate (i.e. member of the same sentence) but not (directly) syntactically related. In Sentence (B.6), the NP *the United Kingdom* is not directly related to the full verb *condemned* since it is embedded into another NP *the goverment of the United Kingdom* being the subject of that verb.

(B.6) [The government of [the United Kingdom]$_{NP}$]$_{NP}$] has [condemned]$_{V}$ [the terrorist attack]$_{NP}$.

Note that this term has been coined within the context of this thesis.

B.5 Controlling Construction

Controlling constructions are those constructions in which a verb which takes a sentential complement determines some syntactic argument of the embedded predicate within the sentential complement. In Sentence (B.7), the verb *promised* subcategorizes (among other arguments) the sentential complement *to leave the house as soon as possible*. Within this sentential complement the verb *leave* lacks an *overt* subject, i.e. it is not realized (I denote this empty constituent with e). The controlling verb *promised* controls this subject. This means that the semantic and syntactic content of the subject of *promised*, i.e. *Peter*, is projected onto the subject of the embedded verb *leave*, i.e. the empty constituent e.

(B.7) Peter$_i$ promised Mary e_i to leave the house as soon as possible.

B.6 Raising Construction

Raising Constructions are those constructions in which arguments of a predicate are moved out of their clausal boundaries. In Sentence (B.8), the subject of *lost* has been moved out of the inf nitival clause in order to become the syntactic subject of the embedding clause. The empty constituent e signalizes the slot from where the word has been raised.

(B.8) John$_i$ appeared e_i to have lost the competition.

Thus, *John* is both the (syntactic) subject of *appeared* and *lost*.
From a technical perspective raising and controlling constructions can be dealt with in the same manner. The difference of these constructions lies in the semantic interpretation. In contrast to controlling constructions, the raised argument has no semantic relevance in the clause into which it has been raised, i.e. in Sentence (B.8) there is no direct relation between *appear* and its (syntactic) subject *John* from a semantic point of view (it is only a semantic argument of *lost*).

Appendix C

Classif cation of Question Types

This appendix displays which question types presented in (Li & Croft, 2001) can co-occur with event questions. This classif cation is used during event question classif cation (see Chapter 4.2) for ruling out those questions which bear a question type which never co-occurs with event questions.

Question Type	**Description**	**Potential Event Question Type ?**
	Abbreviation	
abb	abbreviation	no
exp	expression abbreviated	no
	Entity	
animal	animals	yes
body	organs of body	yes
color	colors	no *(too rare)*
creat	creative material (inventions, books, etc.)	yes
currency	currency names	yes
dis.med	diseases and medicine	yes
event	events	yes
food	food	yes
instru	musical instruments	yes
lang	languages	yes
letter	letters of an/the alphabet	no *(too rare)*

continued on next page

Question Type	Description	Potential Event Question Type ?
other	all other entities that cannot be classif ed as entities of the other classes within this section	yes
plant	plants	yes
product	(mostly) man-made products	yes
religion	religions	no *(too rare)*
sport	sports	yes
substance	elements and substances	yes
symbol	symbols and signs	no *(too rare)*
techmeth	techniques and methods	no
termeq	equivalent terms	no
veh	vehicles	yes
word	words with a special property	no
Description		
def	def nition of something	no
desc	description of something	no
manner	manner of an action	yes *(but too diff cult to model)*
reason	reasons	yes *(but too diff cult to model)*
Human		
gr	a group of organization of persons	yes
ind	an individual	yes
title	academic rank, title of nobility or professions	yes
desc	description of a person	no
Location		
city	cities	yes

continued on next page

Question Type	Description	Potential Event Question Type ?
country	countries	yes
mountain	mountains	yes
other	other locations which cannot be allocated to the other types of locations	yes
state	states	yes
Numeric		
code	postcodes, phone numbers etc.	no *(too rare)*
count	number of something	yes
date	dates	yes
dist	linear measures	no
money	prices	yes
ord	ordinal numbers	yes
other	other numbers which cannot be allocated to the other types of numeric expressions	yes
period	the duration of some action or event	yes
perc	fraction	yes
speed	speed	yes
temp	temperature	yes
weight	weight	yes

Appendix D

The Different Features for Matching Operations

This appendix describes the individual features used for matching different types of atomic entities mentioned in Chapter 4.4.1. Features are subdivided into the ones used for matching *event denoting expressions (edes)* and arguments (this includes the features for question arguments).

D.1 Features for Mapping Event Denoting Expressions

Feature (f^i_{edeMap})	**Description**	**Metric**
`lemma`	measures the similarity of the lemma	Levenshtein Measure (*Sam's String Metrics*, n.d.)
`pos`	measures similarity of POS tags	Similarity is measured on the basis of the size of the common prefix, i.e. $\frac{\#common\ prefix\ chars}{\#chars\ in\ POS\ of\ question}$.
`semI`	Can the matching *ede* be found within the same Lexicographer File of WordNet? *(Note that for this feature the output of NE tagging has been mapped onto the corresponding Lexicographer File.)*	binary feature

continued on next page

Feature (f^i_{edeMap})	**Description**	**Metric**
`semII`	distance of the synsets in WordNet of the *edes* via the hyponomy-relation	Wu & Palmer Metrics (Wu & Palmer, 1994)
`frame`	How similar are the sub-categorization frames of the two *edes*?	$\frac{\#common\ phrase{-}labels}{\#phrase{-}labels\ in\ subcat\ of\ question}$
`mainarg`	Are both *edes* the main predicates within the sentence (and therefore centre of the main (predicate-)argument structure)?	binary feature

D.2 Features for Mapping Arguments

Feature ($f^i_{qArgMap}$, f^i_{argMap})	**Description**	**Metric**
`lemma`	measures the similarity of the lemma	Levenshtein Measure as implemented in (*Sam's String Metrics*, n.d.)
`phrstr` *(only for argMaps)*	measures the similarity of the entire phrases (a phrase is represented by a string of its terminal nodes)	Levenshtein Measure as implemented in (*Sam's String Metrics*, n.d.)
`pos`	measures similarity of POS tags	Similarity is measured on the basis of the size of the common prefix, i.e. $\frac{\#common\ prefix\ chars}{\#chars\ in\ POS\ of\ question}$.
`phrase`	measures similarity of phrase labels	Similarity is measured on the basis of the size of the common prefix, i.e. $\frac{\#common\ prefix\ chars}{\#chars\ in\ phrase-label\ of\ question}$.
`gram`	measures the similarity of the grammatical functions	binary feature
`argstat`	Do the two matching arguments have the same status, i.e. are they both arguments, adjuncts or just NP-satellites found in the vicinity of the *ede*?	binary feature
`semI`	Can the matching arguments be found within the same Lexicographer File of WordNet? *(Note that for this feature the output of NE tagging has been mapped onto the corresponding Lexicographer File.)*	binary feature

continued on next page

Feature ($f^i_{qArgMap}$, f^i_{argMap})	**Description**	**Metric**
`semII`	distance of the synsets in WordNet of the arguments via the hyponomy-relation	Wu & Palmer Metrics (Wu & Palmer, 1994)
`semIII` *(only for qArgMaps)*	Is the Lexicographer File tag of the answer constituent reconcilable with the question type of the question constituent? Possible mappings are listed in Appendix E.	binary feature
`dist`	How similar is the spatial distances to the respective predicate?	$\frac{min(dist(arg^Q),dist(arg^A))}{max(dist(arg^Q),dist(arg^A))}$ where $dist$ is the distance-function (distance from the argument to its *ede*), arg^Q is the argument in the question and arg^A is the argument in the candidate answer sentence
`ori`	Is the orientation to the *ede* identical?	binary feature

Appendix E

Mapping from Question Classes to Lexicographer Files in WordNet

The appendix presents the possible mappings from question classes (Li & Croft, 2001) onto WordNet Lexicographer Files (Miller et al., 1990). These mappings are required for matching semantically empty question constituents to candidate answer constituents. As far as semantically empty question constituents are concerned the only semantic information can be drawn from the underlying question type with which the question has been assigned.

Class	Description	WordNet	Comments
		Abbreviation	
abb	abbreviation	*no mapping required*	These questions are exclusively non-event questions.
exp	expression abbreviated	*no mapping required*	These questions are exclusively non-event questions.
		Entity	
animal	animals	animal	
body	organs of body	body	
color	colors	*no mapping required*	too rare
creat	creative material (inventions, books, etc.)	act and communication	

continued on next page

Class	Description	WordNet	Comments
currency	currency names	quantity	
dis.med	diseases and medicine	artifact, state and substance	
event	events	act, communication, event, phenomenon, state and time	*time* are only epochs
food	food	animal, food, plant and substance	
instru	musical instruments	artifact	
lang	languages	communication	
letter	letters of an/the alphabet	*no mapping required*	too rare
other	all other entities that cannot be classif ed as entities of the other classes within this section	*no mapping possible*	This class is semantically very inhomogeneous.
plant	plants	plant	
product	(mostly) man-made products	artifact and substance	Low coverage expected since many entities of this kind are brands and can therefore not be mapped onto any WordNet Synset.
religion	religions	*no mapping required*	too rare
sport	sports	act and artifcat	
substance	elements and substances	artifact and substance	
symbol	symbols and signs	*no mapping required*	too rare
techmeth	techniques and methods	*no mapping possible*	Terms labelled with this class are too domain specif c and cannot be recognized by open-domain knowledge bases, such as WordNet.

continued on next page

Class	Description	WordNet	Comments
termeq	equivalent terms	*no mapping required*	These questions are exclusively non-event questions and need not be dealt with.
veh	vehicles	artifact	Many vehicles cannot be recognized because they are not addressed by the type of vehicle they belong to but a specif c name, such as *HMS Victory*.
word	words with a special property	*no mapping required*	These questions are exclusively non-event questions.
		Description	
def	def nition of something	*no mapping required*	These questions are exclusively non-event questions.
desc	description of something	*no mapping required*	These questions are exclusively non-event questions.
manner	manner of an action	*no mapping possible*	Answers are often a sequence of events; relations between these events and the question can only be reliably established via discourse analysis.
reason	reasons	*no mapping possible*	Answers are often events, relations between these events and the question can only be reliably established via discourse analysis.

continued on next page

Class	Description	WordNet	Comments
		Human	
gr	a group of persons (e.g. an organization)	group	
ind	individuals	person	
title	academic rank, title of nobility or professions	person	
desc	description of a person	*mapping not required*	These questions are exclusively non-event questions.
		Location	
city	cities	location	
country	countries	location	
mountain	mountains	location	
other	other locations which cannot be allocated to one of the other types of locations	location	
state	states	location	
		Numeric	
code	postcodes, phone numbers etc.	*no mapping required*	too rare
count	quantif ed noun	*any Lexicographer File*	
date	dates	time	
dist	linear measures	*no mapping required*	These questions are exclusively non-event questions.
money	prices	quantity	
ord	ordinal numbers	quantity	too rare
other	other numbers which cannot be allocated to the other types of numeric expressions	quantity	

continued on next page

Class	Description	WordNet	Comments
period	the duration of something	time	
perc	fractions	quantity	too rare
speed	speed	quantity	too rare
temp	temperature	quantity	
weight	weight	quantity	

Appendix F

An Extract from an ARFF File

This appendix illustrates an extract of a training f le in ARFF-format. ARFF is the preferred format for the WEKA toolkit which was used in this thesis to estimate parameter weights.

```
@relation qArgMap

@attribute lemma-match real
@attribute pos-match real
@attribute sem-match-I real
@attribute sem-match-II real
@attribute sem-match-III real
@attribute phrase-label-match real
@attribute gram-func-match real
@attribute arg-status-match real
@attribute distance-to-pred-match real
@attribute orientation-match real
@attribute judgement true,false
@data

0.0 0.0 0.0 0.0 0.0 1.0 1.0 1.0 0.6666667 0.0 true
0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.6666667 1.0 false
0.0 0.6666667 1.0 0.8181818 1.0 1.0 0.0 0.0 0.5 1.0 false
0.28571427 0.6666667 0.0 0.3809524 0.0 1.0 0.0 0.0 0.75 1.0 false
0.0 0.6666667 0.0 0.47058824 0.0 1.0 1.0 1.0 0.75 0.0 false
0.28571427 0.6666667 1.0 0.0 1.0 1.0 0.0 0.0 0.375 0.0 false
0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 true
0.0 0.0 0.0 0.0 1.0 1.0 1.0 1.0 0.33333334 0.0 false
0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.1875 0.0 false
0.19999999 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.1764706 0.0 false
0.19999999 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.16666667 0.0 false
0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.12 1.0 false
```

```
0.25 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.13636364 1.0 false
0.111111104 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.15 1.0 false
0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.75 0.0 false
0.19999999 0.0 0.0 0.0 1.0 1.0 0.0 1.0 0.6 0.0 true
0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.13636364 1.0 false
0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.15789473 1.0 false
...
```

References

Abney, S. (1997). *The SCOL Manual Version 0.1b.* (`http://www.sfs.uni-tuebingen.de/~abney/`)

Bach, E. (1986). The Algebra of Events. *Linguistics and Philosophy*, *9*, 5–16.

Buchholz, S. (2001). Using Grammatical Relations, Answer Frequencies and the World Wide Web for Question Answering. In *Proceedings of the 10th Text REtrieval Conference (TREC 2001).* Gaithersburg, Maryland, USA: National Institute of Standards and Technology (NIST).

Budanitsky, A., & Hirst, G. (2006). Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, *32*(1), 13–47.

Burchardt, A., Erk, K., & Frank, A. (2005). A Wordnet Detour to Framenet. In *Sprachtechnologie, mobile Kommunikation und linguistische Resourcen* (p. 16). Frankfurt am Main: Lang, Peter.

Cessie, S. le, & Houwelingen, J. van. (1992). Ridge Estimators in Logistic Regression. *Applied Statistics*, *41*(1), 191–201.

Charniak, E. (2000). A Maximum-Entropy inspired Parser. In *Proceedings of the First Meeting of the North American Chapter of Association for Computational Linguistics (NAACL 2000)* (pp. 132–139). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Clarke, C., Cormack, G., Kemkes, G., Laszlo, M., Lynam, T., Terra, E., et al. (2002). Statistical Selection of Exact Answers (Multitext Experiments for TREC 2002). In *Proceedings of the 11th Text REtrieval Conference (TREC 2002).* Gaithersburg, Maryland, USA: National Institute of Standards and Technology (NIST).

Collins, M. (1997). Three Generative, Lexicalised Models for Statistical Parsing. In P. R. Cohen & W. Wahlster (Eds.), *Proceedings of the 35th Annual Meeting on Association for Computational Linguistics* (pp. 16–23). Somerset, New Jersey: Association for Computational Linguistics.

Crowe, J. (1995). Constraint-based Event Recognition for Information Extraction. In *Meeting of the Association for Computational Linguistics* (pp. 296–298). Morristown, NJ, USA: Association for Computational Linguistics.

Cunningham, H., Maynard, D., Tablan, V., Ursu, C., & Bontcheva, K. (2001). *Developing Language Processing Components with GATE (Version 3).* (`http://gate.ac.uk/sale/tao/tao.pdf`)

Curran, J., & Clark, S. (2003a). Investigating GIS and Smoothing for Maxi-

mum Entropy Taggers. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 1 (EACL 2003)* (pp. 91–98). Morristown, NJ, USA: Association for Computational Linguistics.

Curran, J., & Clark, S. (2003b). Language Independent NER Using a Maximum Entropy Tagger. In *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL-03)* (pp. 164–167).

Dagan, I., Glickman, O., & Magnini, B. (2005). The PASCAL Recognising Textual Entailment Challenge. In *The PASCAL Challenges Workshop on Recognising Textual Entailment* (p. 177-199).

Davidson, D. (1967). The Logical Form of Action Sentences. In N. Rescher (Ed.), *The Logic of Decision and Action* (pp. 81–95). Pittsburgh: University of Pittsburgh Press.

Drummond, C., & Holte, R. (2005). Severe Class Imbalance: Why Better Algorithms Aren't the Answer. In *Proceedings of the 16th European Conference of Machine Learning.* Porto, Portugal: NRC.

Durme, B. van, Huang, Y., Kupść, A., & Nyberg, E. (2003). Towards Light Semantic Processing for Question Answering. In *HLT/NAACL Workshop on Text Meaning.* Morristown, NJ, USA: Association for Computational Linguistics.

Erk, K., & Padó, S. (2006). SHALMANESER - A Toolchain for Shallow Semantic Parsing. In *Proceedings of Language Resources and Evaluation (LREC 2006).*

Fillmore, C. (1968). The Case for Case. In E. Bach & R. Harms (Eds.), *Universals in Linguistic Theory* (pp. 1–90). New York: Holt, Rhinehart and Winston.

Fillmore, C., Johnson, C., & Petruck, M. (2003). Background to FrameNet. *International Journal of Lexicogaphy*, *16*, 235–250.

GUTime (Time Tagger). (n.d.). (`http://www.timeml.org/site/tarsqi/modules/gutime/index.html`)

Hasegawa, T., Sekine, S., & Grishman, R. (2005). Unsupervised Paraphrase Acquisition via Relation Discovery. In *11th Annual Meeting of the Japanese Association for Natural Language Processing.* Takamatsu, Japan.

Hirschman, L., & Gaizauskas, R. (2001). Natural Language Question Answering: The View from Here. *Natural Language Engineering*, *7*(4).

Hornby, A. (1995). *Oxford Advanced Learner's Dictionary of Current English* (Fifth ed.). New York: Oxford University Press.

JWNL - Java WordNet Library. (n.d.). (`http://nlp.stanford.edu/nlp/javadoc/jwnl-docs/`)

Kaisser, M. (2006). *Web Question Answering by Exploiting Wide-Coverage Lexical Resources.* (`http://staff.science.uva.nl/~katrenko/stus06/images/kaisser.pdf`)

Korhonen, A., Krymolowski, Y., & Briscoe, T. (2006). A Large Subcategorization

Lexicon for Natural Language Processing Applications. In *Proceedings of Language Resources and Evaluation (LREC 2006).*

Leidner, J., Bos, J., Dalmas, T., Curran, J. R., Clark, S., Bannard, C., et al. (2004). The QED Open-Domain Answer Retrieval System for TREC 2003. In *Proceedings of the Twelfth Text Retrieval Conference (TREC 2003)* (pp. 595–599). Gaithersburg, MD.

Li, X., & Croft, W. (2001). *Incorporating Syntactic Information in Question Answering* (Tech. Rep. No. CIIR-239). Amherst, Massachusetts, USA: University of Massachusetts. (`http://ciir.cs.umass.edu/pubfiles/ir-239.pdf`)

Li, X., & Roth, D. (2005). Learning Question Classif ers: The Role of Semantic Information. *Journal of Natural Language Engineering*, *11*(4).

Lin, C.-S., & Smith, T. (2006). A Tree-based Algorithm for Predicate-Argument Recognition. *Association for Computing Machinery New Zealand Bulletin*, *2*(1), online journal.

Lin, D. (1998). Dependency-based Evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems.* Granada, Spain.

Lin, D., & Pantel, P. (2001). DIRT - Disovery of Inference Rules from Text. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining.*

Linguistics in SIL. (n.d.). (`http://www.sil.org/linguistics`)

Macleod, C., Grishman, R., & Meyers, A. (1998). *COMLEX Syntax Reference Manual.* (`http://nlp.cs.nyu.edu/comlex/refman.ps`)

Macleod, C., Grishman, R., Meyers, A., Barret, L., & Reeves, R. (1998). NOMLEX: A Lexicon of Nominalizations. In *Proceedings of EURALEX'98.* Liège, Belgium.

Makkonen, J., Ahonen-Myka, H., & Salmenkivi, M. (2003). Topic Detection and Tracking with Spatio-Temporal Evidence. In *Proceedings of 25th European Conference on Information Retrieval Research (ECIR 2003)* (pp. 251–265). Berlin: Springer-Verlag.

McCarthy, K., Zabar, B., & Weiss, G. (2005). Does Cost-Sensitive Learning Beat Sampling for Classifying Rare Classes? In *Proceedings of the 1st International Workshop on Utility-based Data Mining* (pp. 69–77). Bronx, NY, USA: ACM Press.

Mengel, A., & Lezius, W. (2000). An XML-Based Encoding Format for Syntactically Annotated Corpora. In *Proceedings of the Second International Conference on Language Resources and Engineering (LREC 2000)* (pp. 121–126). Athens.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., et al. (2004). *The Cross-Breeding of Dictionaries.* (`http://nlp.cs.nyu.edu/meyers/papers/dict-breed.pdf`)

Miller, G., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. (1990). Introduction to Wordnet: An On-line Lexical Database. *International Journal of Lexicography*, *3*(4), 235–244.

Moens, M., & Steedman, M. (1988). Temporal Ontology and Temporal Reference. *Computational Linguistics*, *14*(2), 15–28.

Monz, C. (2004). Minimal Span Weighting Retrieval for Question Answering. In R. Gaizauskas, M. Greenwood, & M. Heppel (Eds.), *Proceedings of the SIGIR Workshop on Information Retrieval for Question Answering* (pp. 23–30). Sheff eld, UK.

Müller, K. (2004). Semi-Automatic Construction of a Question Treebank. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004).*

Papka, R., & Allan, J. (1998). *On-Line New Event Detection Using Single Pass Clustering* (Tech. Rep. No. UM-CS-1998-021). Amherst, Massachusetts, USA: University of Massachusetts.

Parsons, T. (1990). *Events in the Semantics of English: A Study in Subatomic Semantics*. Cambridge/MA: MIT Press.

Passonneau, R. (1988). A Computational Model of the Semantics of Tense and Aspect. *Computational Linguistics*, *14*(2), 44–60.

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). *WordNet::Similarity - Measuring the Relatedness of Concepts.* Appears in the Proceedings of the Nineteenth National Conference on Artif cial Intelligence (AAAI-04).

Pustejovsky, J., Castano, J., Ingria, R., Saurí, R., Gaizauskas, R., Setzer, A., et al. (2003). TimeML: Robust Specif cation of Event and Temporal Expressions in Text. In *Proceedings, AAAI Spring Symposium on New Directions in Question Answering.* Tilburg, Netherlands.

Radford, A. (1997). *Syntactic Theory and the Structure of English*. Cambridge: Cambridge University Press.

Rijsbergen, C. J. van. (1979). *Information Retrieval* (2 ed.). London: Butterworths.

Sam's String Metrics. (n.d.). (`http://www.dcs.shef.ac.uk/sam/~simmetrics.html`)

Saurí, R., Knippen, R., Verhagen, M., & Pustejovsky, J. (2005). Evita: A Robust Event Recognizer for QA Systems. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing* (pp. 700–707). Morristown, NJ, USA: Association for Computational Linguistics.

Sekine, S. (2006). On-Demand Information Extraction. In *International Committee on Computational Linguistics and the Association for Computational Linguistics* (pp. 731–738). Morristown, NJ, USA: Association for Computational Linguistics.

Shen, D., & Klakow, D. (2006). Exploring Correlation of Dependency Relation Paths for Answer Extraction. In *Proceedings of the ACL 2006.* Morristown, NJ, USA: Association for Computational Linguistics.

Shen, D., Kruijff, G.-J., & Klakow, D. (2005). Exploring Syntactic Relation Patterns for Question Answering. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP)* (pp. 889–896).

Shen, D., Leidner, J., Merkel, A., & Klakow, D. (2006). The Alyssa System at

TREC 2006: A Statistically-Inspired Question Answering System. In *Workshop notes of the Text REtrieval Conference (TREC 2006).* Gaithersburg, MD, USA: National Institute of Standards and Technology.

Siegel, E., & McKeown, K. (2000). Learning Methods to Combine Linguistic Indicators: Improving Aspectual Classif cation and Revealing Linguistics Insights. *Computational Linguistics*, *26*(4), 595–627.

Sinha, S., & Narayanan, S. (2005). Model-based Anwer Selection. In *Proceedings of the Twentieth National Conference on Artif cial Intelligence 2005 (AAAI-05).* Pittsburgh, Pennsylvania, USA: American Association for Artif cial Intelligence.

TIGER API 1.8 - A Java Interface to the TIGER Corpus. (n.d.). (`http://www.tigerapi.org/`)

TREC Answer Patterns. (2005). (`http://trec.nist.gov/data/qa/2005_qadata/KL/factoid-docs.litkowski.txt`)

Vendler, Z. (1967). Verbs and Times. *Linguistics and Philosophy*, 97–121.

Voorhees, E. (2000). Overview of the TREC-9 Question-Answering Track. In *Proceedings of the Nineth Text Retrieval Conference (TREC 2000).* Gaithersburg, MD.

Voorhees, E., & Harman, D. (2005). *TREC : Experiment and Evaluation in Information Retrieval*. Cambridge, Massachusetts, USA: The MIT Press.

Voorhees, E., & Tice, D. (2000). Building a Question Answering Test Collection. In *Proceedings of the 23rd Annual International ACM ISGIR Conference on Research and Development in Information Retrieval* (pp. 200–207). ACM Press.

Weiss, G., & Provost, F. (2003). *The Effect of Class Distribution on Classif er Learning: An Empirical Study* (Tech. Rep. No. ML-TR 43). New Jersey, USA: Department of Computer Science, Rutgers University.

Wikipedia - The Free Encyclopedia. (n.d.). (`http://en.wikipedia.org/`)

Witten, I., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques* (Second ed.). San Francisco, California, USA: Morgan Kaufmann.

Wu, Z., & Palmer, M. (1994). Verb Semantics and Lexical Selection. In *32nd Annual Meeting of the Association for Computational Linguistics (ACL)* (p. 133-138). Morristown, NJ, USA: Association for Computational Linguistics.