Answering Comparison Questions: What's the Difference?

Silke Scheible

Supervisors: Bonnie Webber and Johan Bos

Master of Science

in

Speech and Language Processing

Theoretical and Applied Linguistics

School of Philosophy, Psychology and Language Sciences

University of Edinburgh

2005

Abstract

This Masters thesis is a first attempt at dealing with comparison questions in open-domain question answering. Research so far has only been conducted within closed domains. While closed-domain systems assume that all the information necessary to answer a question is contained in a structured database, open-domain systems have to cope with large collections of unstructured text.

The obvious advantage of open-domain systems is that they are able to deal with a wide variety of topics, while closed-domain systems can only answer questions within a limited topic area. A look at question logs on the Internet reveals that users quite frequently query the differences between two concepts, which is why an open-domain system would be most useful.

This study has multiple purposes. Firstly, it provides an overview of the task of answering difference questions in open domains. And secondly, some practical experiments are carried out to investigate the difficulty of carrying out the first step in answering difference questions, namely to identify the senses of the question terms. Although this represents only a small step towards a full implementation of a difference questions system, the foundations are thus laid for further research.

Acknowledgements

First and foremost, I would like to thank my dissertation supervisor, Professor Bonnie Webber, not only for her scholarly guidance and advice, but also for her encouragement and support throughout this project's development. I would also like to thank my parents for their constant support during my studies. Special thanks also go to Justin Waugh, who spent hours disambiguating WordNet senses for me, and to Evia, Michael and Agis, for supporting me during difficult hours at Appleton Tower.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text. This work has not been submitted for any other degree or professional qualification except as specified.

(*Silke Scheible*)

Table of Contents

Chapter 1

Introduction

Anyone who has used the web-service AskJeeves[1] before knows that the virtual butler does not provide actual answers to the questions he is asked. Instead, he returns snippets of text from the web which may include the answer to the question, just like a common web-based search engine. Users of AskJeeves are without doubt also familiar with the feeling of frustration when the excerpts of text returned by the butler are of little or no relevance to the question asked. In recent years there has been a growing interest in systems that are able to provide the user with precise answers to their questions. These systems are commonly referred to as QA (Question-Answering) systems and take large collections of texts or the World Wide Web as their data bases. The idea is that a QA system takes a question in everyday language as input (and not just keywords), and returns a precise answer (and not just documents which include the answer somewhere).

Research into question answering has been greatly encouraged by the annual TREC[2] conference, which added a question answering track to its program in 1999. Modern QA systems are able to deal with a number of question types. However, although more and more types are included in the QA competition every year, no research has yet been carried out to investigate questions like the following:

What is the difference between a boa and a python?

What are the similarities between the Earth and Mercury?

[1] `www.ask.com`.

[2] Text Retrieval Conference, `http://trec.nist.gov/`.

Comparisons in terms of similarities and differences are an important part of human perception. When we encounter new concepts in everyday life, we automatically compare them to concepts we already know. The present study is an attempt to tackle comparison questions in the framework of open-domain question answering.

The first chapter will be concerned with theoretical aspects of difference questions. A framework for this question type will be proposed which is based on psychological insights. After investigating the acceptability of difference questions and answers, real user questions taken from a question log on the Internet will be discussed.

The second chapter is concerned with previous work in the area, which includes a discussion of how two closed-domain systems deal with difference questions: the TEXT system and the SHAKEN system. Then, an overview of the most important processing steps in an open-domain system will be presented. A discussion of possible knowledge sources will conclude the theoretical part of the work.

The third and fourth chapters investigate how the first processing step in answering difference questions can be tackled, namely, how to ”make sense” of the questions. This step is important as question terms are often polysemous. Two ways are proposed for their disambiguation: in the third chapter, a simple edge-based algorithm is proposed and implemented, while the last chapter of this work describes how sophisticated similarity measures can be used for disambiguating the question terms.

Chapter 2

Difference questions

2.1 What are difference questions?

Consider the following set of ”What’s the difference” questions:

1. What’s the difference between a clock and a watch?
2. What’s the difference between a frog and a toad?
3. What’s the difference between a tin and a can?
4. What’s the difference between a dog and a cat?
5. What’s the difference between a fish and a bicycle?

As an initial experiment, I asked five native speakers of English how they would expect a question answering system to answer these questions. Here are the answers of three of the participants:

1. What’s the difference between a clock and a watch?

A: A watch is small and worn on your wrist. A clock is bigger and goes on the wall.

B: Both tell the time, a clock sits on the wall or a shelve, whilst a watch sits on wrist.

C: A watch is a small, portable timepiece normally worn on the wrist and attached with a strap. A clock is typically a larger timepiece which remains in one location.

2. What’s the difference between a frog and a toad?

A: A frog is smooth and lives in the water. A toad is bumpy and lives on the land.

B: Different categories of closely related amphibian. Toads are generally larger with bumpy skin.

C: I don't know. They seem to be very similar to me.

3. What's the difference between a tin and a can?

A: A tin usually has a lid that is replacable. A can is totally sealed and cannot be resealed once opened.

B: A tin refers to a special type of metal that container is made out of. Also a tin can contain both solids and fluids, a can generally contains only fluid.

C: A can is a metal cylinder enclosed with a metal top and bottom typically used for storing liquids of some sort, while a tin is a small metal container used to store candies.

4. What's the difference between a dog and a cat?

A: A dog is a canine and a cat is a feline.

B: Two common housepets. One being of the canine species, the latter being a member of the feline species.

C: They are both domesticated species of mammals, but cats are classified as feline and dogs as canine (a picture would definitely help).

5. What's the difference between a fish and a bicycle?

A: A fish is an animal that lives in the water. A bicycle is a metal apparatus that people use as a self-powered transport machine.

B: A fish swims in the water, a bicycle is a kinetically powered contraption for transporting humans.

C: These two items don't seem to have anything in common: a fish is an underwater animal that can be caught and eaten, while a bicycle is a non-motorised form of transportation for humans.

The answers of the participants demonstrate that dealing with difference questions is by no means a trivial task, and that there may be a number of possible answers. While the participants seem to have a clear opinion about what the differences between a clock and a watch are, they appear less clear about tins and cans. It is also questionable whether a person who wants to know about the differences between dogs and cats is content with an answer involving the terms canine and feline only. It clearly depends on the users and their prior knowledge how detailed an answer should be. Are all the answers given above equally acceptable, or is one of them better than the other? And if so, what is it that makes it a better answer? Furthermore, what can be considered a sensible question at all? While asking for the differences between a clock and a

watch or between a dog and a cat makes perfect sense, question 5 involving the fish and the bicycle appears slightly strange. As participant C notes, they ”don’t seem to have anything in common”. Another participant remarked that the question seems ”rather silly”, while a third one commented that a question answering system should really reject such a question. It should therefore be possible to draw a line between acceptable and unacceptable questions. However, as will be discussed in the following sections, none of these issues is straightforward to deal with.

2.1.1 A framework for discussing difference questions

The study of comparisons in terms of similarities and differences has long been a central issue in academic areas such as psychology, philosophy or linguistics. As the well-known philosopher and psychologist William James once pointed out, the human species differs from all other animals in their ability to extract common elements from comparisons. The ability to see what makes objects, thoughts or words similar to each other can be seen as the basis for our theories of perception and cognition, or as James observes, ”this sense of Sameness is the very keel and backbone of our thinking” (James, 1950). Quine puts it this way: ”similarity, is fundamental for learning, knowledge and thought, for only our sense of similarity allows us to order things into kinds so that these can function as stimulus meanings” (Quine, 1969).

In recent years there has been a renewed interest in the study of similarity, with a special focus on how it can be described in a formal framework. Several models have been proposed: geometric, feature-based, alignment-based and transformational models, to mention just a few (for an overview see Hahn (2003)). While all of these models have advantages and drawbacks, the alignment-based model seems to be most suitable for the purpose of describing difference questions, as it pays particular attention to the treatment of differences. Like the other models, the alignment-based view assumes that the similarity of two items increases with its commonalities, and decreases with its differences. Gentner and Markman (1994), whose work is most influential in this area, characterise similarity as a comparison of structured mental representations. Mental representations consist of hierarchical systems which contain objects, attributes of objects, relations between objects, and relations between relations. Markman and Gen-

tner illustrate this with the example displayed in Figure 2.1, where the configurations on the top can be described by the structural representations at the bottom (Gentner and Markman, 1994).

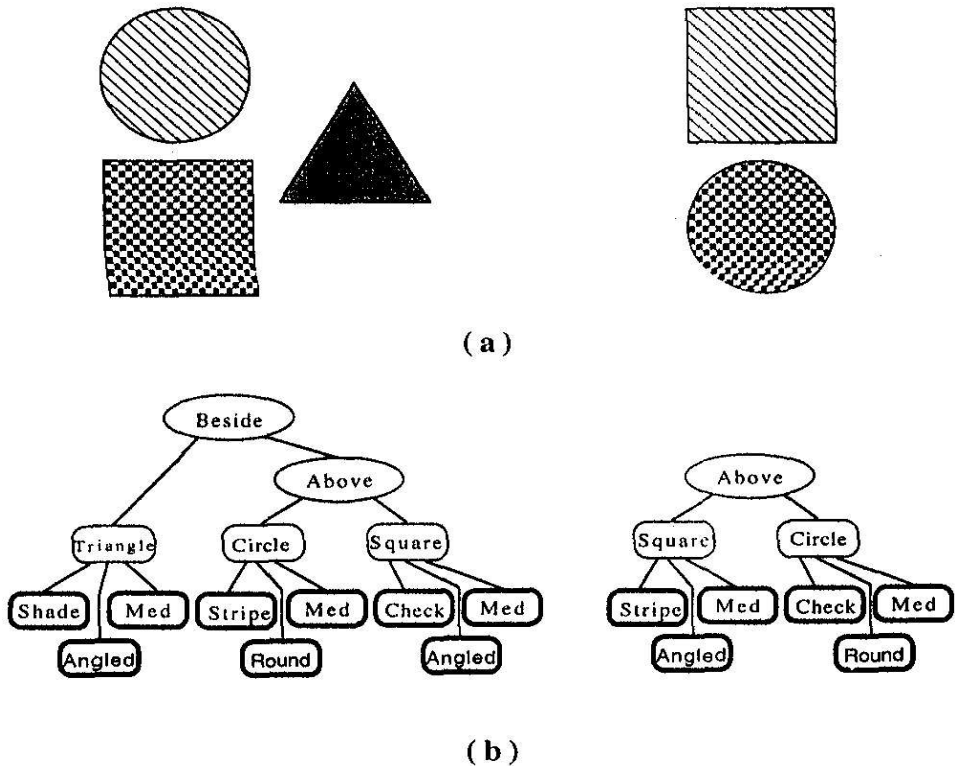

Figure 2.1: (a) Simple configurations of objects and (b) simple relational structure describing the configurations (adapted from Markman et al 1994)

The process of comparison can be described as ”one of structural alignment between two mental representations to find the maximal structurally consistent match between them” (Gentner and Markman, 1994). A structurally consistent alignment is one that obeys the constraints of *one-to-one mapping* and *parallel connectivity*. *One-to-one mapping* requires that an element in one representation corresponds to at most one element in the other representation. For example, the square on the left hand side in Figure 2.1 cannot correspond to both the square and the circle on the right hand side. The *parallel connectivity* constraint requires that if an element in one representation matches an element in the other representation, then the elements that are linked to them must also match. Therefore, if the ”above” relations in Figure 2.1 are matched,

this requires that the arguments of this relation must also match (i.e. the striped circle matches the striped square, and the checked square matches the checked circle).

Markman and Gentner draw attention to the fact that ”in many cases more than one structurally consistent interpretation is possible for a given comparison” (Gentner and Markman, 1994). For instance, in the example above, one interpretation may be based on the fact that both configurations have a circle in common. Another interpretation may be based on the fact that both have a shape above another shape. Therefore, when commonalities and differences of two items are investigated, one has to be aware of the fact that this is always with respect to a certain interpretation. This means that there can be several answers to difference questions which are all correct.

Although not without shortcomings, Markham and Gentner's model provides a framework that is most useful for a discussion of difference questions. This is due to the fact that in their model a comparison yields not only the commonalities of the items, but also two types of differences: those related to the commonalities (which they call *alignable* differences) and differences that are not related to the commonalities (*non-alignable* differences) (Markman and Gentner, 1996). Markman and Gentner carried out a series of experiments and found support for the following claims:

1. Similar pairs of concepts have many alignable differences, whereas dissimilar pairs have few alignable differences.

2. Alignable differences are often conceptually related to the commonalities from which they are derived. For example, Markman and Gentner noted that participants asked to list commonalities for the pair *car*/*motorcycle* often said ”Both have wheels”, while when asked for the differences between the pair they listed the alignable difference ”Cars have four wheels and motorcycles have two wheels”.

3. Alignable differences are more salient from a psychological perspective than non-alignable differences. As a consequence, people find it easier to list differences for similar pairs than for dissimilar pairs. This is against our general intuition that people should find it easier to list differences for dissimilar pairs.

These findings provide some important insights for the task of answering difference questions. The next two sections will discuss the issues of acceptable questions and acceptable answers within the framework provided by Markman and Gentner.

2.2 Acceptable difference questions

Not all difference questions are equally acceptable to native speakers of English. This is illustrated by the reactions of the participants to the question ”What’s the difference between a fish and a bicycle?”. A question answering system dealing with difference questions should therefore be able to reject unacceptable questions just like human speakers do. But how could the threshold be determined that separates acceptable from unacceptable questions? As discussed above, Markman and Gentner showed that people find it more difficult to list differences for dissimilar pairs than for similar pairs, because there are only few alignable differences for dissimilar pairs. This implies that the acceptability of difference questions may be directly related to the similarity of the question terms. I decided to carry out a small experiment to investigate this further.

In 1965, Rubenstein and Goodenough carried out an experiment with human subjects to determine the semantic similarity between nouns (Rubenstein and Goodenough, 1965). They presented their participants with 65 noun pairs (such as *coast-shore*), and asked them to rate the semantic similarity for each pair on a scale from 0 (not similar) to 4 (synonymous). In a follow-up experiment, Miller and Charles chose a subset of 30 of these noun pairs, to represent equal amounts of high, intermediate, and low levels of similarity (Resnik, 1998). They then asked 38 subjects to rate the similarity in the same fashion as Rubenstein and Goodenough had proposed, and found that the new results were highly correlated to the previous ones (0.97). Thus, the average ratings for these noun pairs can be taken as a reliable estimate of the similarity of the pairs.

To investigate whether acceptability of difference questions correlates with the semantic similarity of the question terms, I made use of Miller and Charles’s noun pairs. I adopted a procedure similar to the one applied by Resnik, who replicated the Miller-Charles experiment in 1998. Ten subjects were presented with a list of difference questions involving 28 of the Miller-Charles noun pairs. While the participants in the previous experiments judged the *similarity* of a pair like *gem-jewel*, the participants in my experiment had to rate the *acceptability* of the difference question involving that word pair, i.e. *What’s the difference between a gem and a jewel?* Like in Resnik’s experiment, five subjects received the questions in random order, while the other five

received them in reverse order. After a short explanatory introduction, the participants were asked to judge the acceptability of the questions on a three-point scale: *perfectly acceptable*, *somewhere between*, and *unacceptable*. A reproduction of the test is displayed in Appendix A.1.

The results strongly suggest that the acceptability of difference questions is correlated with the similarity of the question terms. The lower the similarity value of the two words, the less likely it is that the question is judged as acceptable by humans. The terms with the lowest similarity are judged as unacceptable by all speakers (*rooster*/*voyage*, *noon*/*string*, *glass*/*magician*, and *chord*/*smile*). On the other hand, the high similarity pairs are all judged as perfectly acceptable (*furnace*/*stove*, *midday*/*noon*, *magician*/*wizard*, *asylum*/*madhouse*, *coast*/*shore*, *boy*/*lad*, *journey*/*voyage*, *gem*/*jewel*, and *car*/*automobile*). However, defining a precise value for the acceptability threshold would require further carefully designed experiments involving more participants, which is not possible in the scope of this paper. Furthermore, it has to be borne in mind that basically we are dealing with a psycholinguistic question. The amount of commonality that two terms need for a difference question to make sense may differ for different speakers in different situations. Factors such as context or social background are likely to heavily influence the judgement of a speaker. Hence the threshold value should be set to as low as possible.

2.3 Acceptable answers

After discussing acceptable questions in some detail, let us now turn to acceptable answers. The notion of correct/incorrect answers is a notorious one in the discipline of Question Answering, which has led to many heated debates over the evaluation of Question Answering systems.

An evaluation of question types such as factoid questions is fairly straightforward, as there is generally only one correct answer. For instance, the question ”What year was Virginia Woolf born?” has only one correct answer, namely 1882. Definition questions, on the other hand, are considered to be more difficult to evaluate. Consider for example the question ”What is a mouse?”. It clearly depends on the context whether

the answer should involve the terms *rodent* or *computer equipment*. When evaluating answers to such questions these issues must be taken into account, for example by accepting multiple answers as correct.

Difference questions are even more complex than definition questions. In order to answer a question of the form ”What’s the difference between A and B?”, the following sub-questions need to be answered:

1. What is A?
2. What is B?
3. What commonalities do A and B share?
4. What are the differences between A and B?

First of all, it needs to be clarified what A and B are. Then the commonalities of A and B are established. That this step occurs before looking at the actual differences is supported by Markman and Gentner’s finding that people generally first list the alignable differences between concepts. As alignable differences are based on the commonalities of a pair, the third question needs to be tackled before establishing the differences. The fact that difference questions involve at least two definition questions indicates that a fair evaluation of answers is likely to be a very difficult undertaking.

What information should an acceptable answer contain? According to Markman and Gentner’s results, humans find alignable differences psychologically more salient than non-alignable differences. As the answer produced by the system should be as natural as possible to the user, this has to be taken into account, for example by listing the alignable differences first.

The questionnaire discussed in Section 2.1 shows that there is a some uncertainty among the participants as to what an answer to a difference question should involve. This is due to the fact that they do not have all the facts stored in their minds. Participant C, for example, admits that he does not know what the differences between frogs and toads are. I searched FAQ’s on the Internet for expert answers to this question. Here are two examples[1]:

[1]The search term in Google was ”difference between frogs and toads” and produced 823 results. `http://www.wildlifetrust.org.uk/suffolk/yp/try/amphibians.htm`, `http://www.ekpc.com/greenweb/answers.html`

```
1.
* Frogs have a smoother skin than toads, whose skin is 'warty'
* Toads look more stout than frogs
* Frogs hop but toads normally walk
* Frogs are seen near gardens ponds more often that toads because
  toads only go into water to breed and prefer larger ponds
* Frog eggs can normally be seen laid in clusters whereas toad spawn
  is laid in long strings

2.
Toads, typically have wartier skin and shorter legs that frogs.
Frogs, with their long legs jump and toads hop. Most toads lay their
eggs in long strands, while frog eggs are laid in large masses.
Frogs and toads both belong to the order Salientia or Ecaudata
(tailless). Toads are a member of the family Bufonidue, while the
rest of the frogs belong to one of the other six families within
the order Salientia.
```

The answers found on the Internet are more comprehensive than those in my questionnaire. The FAQ answers nicely demonstrate that the differences listed are alignable differences, which focus on the following commonalities shared by frogs and toads:

- **skin** (smooth vs. warty)
- **legs** (short vs. long)
- **movement** (hop/jump vs. walk/hop) (unclear)
- **eggs** (clusters vs. strings)
- **habitat** (garden ponds vs. land/larger ponds)
- **biological family** (Bufonidue vs. Salientia)

As frogs and toads are very similar, there are many alignable differences that can be listed. The large number of hits in Google implies that this is indeed a frequently asked question. The next section will look at more real user questions, in order to see what a question answering system will have to deal with.

2.4 Real user questions

I sifted through a long list of difference questions extracted from a number of question logs on the internet[2]. The pattern searched for was ”difference.*between” and produced a raw list of queries, each of which came with a time-stamp, representing the date and time the query was submitted. The search produced 1005 results, with dates ranging from 28/11/2003 up until 12/06/2004. Here are some examples:

```
2004-01-13 04:45:43 differences between microprocessor and
microcontroller
2004-01-13 06:24:40 difference between newton and kilogram
2004-01-13 12:52:19 difference between management and
financial accounting
2004-01-13 16:15:01 difference between impeller and propeller
```

Most of the results do not represent complete questions. Users seem to prefer elliptical phrases like ”difference between microprocessor and microcontroller” instead of ”What is the difference between a microprocessor and a microcontroller?”. This reflects their intuitive knowledge of how current search engines work: stop words are removed, and hence users focus on what they believe are the most crucial terms in their question. While the incomplete queries presented above are perfectly acceptable as instances of difference questions, this is not always the case, as the following cases demonstrate:

```
2004-01-25 03:38:00 difference between pci and
2004-05-08 19:57:41 whats the difference between the decendants and all
2004-05-18 07:52:40 what is the difference between a printer and a pri
2004-06-18 11:20:41 difference between animals and plants ce
```

Queries where question terms are missing or incomplete are clearly unacceptable, and were therefore removed from the list. However, cases where there were letters missing, or where terms were misspelled, were retained if the intended meaning was clear. While one could argue that such decisions are too a large degree idiosyncratic, it has to be borne in mind that the purpose of the procedure was not a precise statistical analysis of difference queries, but an investigation of the kinds of differences users are

[2]Thanks to Jochen Leidner for providing me with this list.

interested in. The retention of incomplete or misspelled items with clear meanings is therefore justified.

This preprocessing procedure left a list of 460 queries, which were critically analysed and categorised into three main groups according to the nature of the question terms A and B:

1. A and B are both simplex words
2. At least one of the terms is a 'compound', e.g. *prime minister*/*president* or *grizzly bear*/*brown bear*
3. Everything else, including the following subgroups:
    - Questions involving more than two terms, e.g. "What's the difference between internet, extranet and intranet?"
    - Questions involving one term only, e.g. "What's the difference between religions?"
    - Questions including additional information, e.g. "What's the difference between processes and applications *in the task manager*?"
    - Some other minor cases.

The majority of questions fall into Groups 1 and 2. Before discussing these two groups in further detail, a few words are in order about the third group. The first subgroup has nine members, additional examples being the differences between *hepatitis A*, *B* and *C*, between *Windows XP*, *Professional* and *Home*, or the differences between *Islam*, *Judaism* and *Christianity*.

The second subgroup has 13 members, all of which feature only one question term, usually in plural form. It is closely related to the first subgroup, as the question term involved is usually a superordinate term, requiring an explanation of the differences between its subordinate terms. For instance, an answer to the question "What's the difference between religions?" could theoretically involve an explanation of all the religions in the world. Thus, the question encountered in the first subgroup ("difference between Islam, Judaism and Christianity") can be seen as a subquestion, where only three religions are compared (rather than all of them).

The third subgroup comprises 11 members, and includes questions with additional information. There are two main ways in which this can occur. The first kind of additional information modifies the question terms themselves, e.g. by specifying a context in which the question terms should be considered. In the example given above, ”What’s the difference between processes and applications in the task manager?”, the processes and applications to be compared are those specific to the task manager rather than general terms. Or consider the question ”What’s the difference between toner and colour for hair”. Here, the postmodifier ”for hair” specifies what sort of toner and colour is meant, namely, hair toner and hair colour.

The second kind of additional information modifies the noun *difference* itself. This can either occur as a phrase premodifying the noun phrase *difference* (e.g. an adjectival premodifier, as in ”What’s the cultural difference between Korea and Canada?”), or as a postmodifier of *difference* (e.g. a prepositional phrase, as in ”What’s the difference in speed between dsl and cable internet connection?”). Like this, users specify a particular alignable difference they want information on.

Altogether, the third group contains 35 questions, which is a neglectable amount compared to the first two groups. Group 1 contains 207 questions involving simplex question terms, and group 2 contains 218 queries where at least one of the question terms is complex. The classification of questions into groups containing simplex or multiplex question terms is well motivated. The first reason concerns the automatic processing of the question. While an identification of question terms is no problem for simplex terms, this can be quite difficult for multiplex questions. This is due to the fact that the use of complex question terms can lead to a syntactic ambiguity which is commonly referred to as coordination ambiguity (Resnik, 1998). Compare for example the following two queries, where the first includes simplex question terms, and the second has a complex second term:

a) What’s the difference between (an adult) and (a baby)?

b) What’s the difference between (an adult and a baby) skull?

While the first sentence has only one structural analysis (indicated by the brackets), the second one has two: we could either be looking at the differences between an adult

in general and a baby skull, or at the differences between an adult skull and a baby skull. Humans intuitively resolve the ambiguity correctly, because they are aware that only a comparison of two types of skulls makes sense. For an automated system, however, coordination ambiguity poses a major problem.

A second important issue distinguishing Group 1 from Group 2 is the degree of lexicalisation of the question terms. The term lexicalisation refers to a ”gradual historical process, which changes regularly formed complex lexemes and tends to convert them into a single unit with specific content” (Lipka, 2002). A side-effect of this process is that lexicalised items can generally be found in dictionaries, while unlexicalised words are normally not listed. I went through the laborious process of looking up all members of Group 1 and Group 2 in the electronic dictionary WordNet (Version 2.1), i.e. all 850 question terms. I found that in Group 1 297 of 414 terms are listed (ca. 72%). In contrast, only 198 of the 436 terms in Group 2 were encountered in WordNet (ca. 45%, see Appendix A.2 and A.3 for the first 50 members of the lists). The low results for the second group come as no surprise, as multiword expressions are generally less likely to be lexicalised. They are only lexicalised when the expression is very frequent or when its meaning becomes opaque due to semantic change. However, most often, the meanings of the multiword expressions are simply the meaning of the sum of its components, and as this meaning is transparent there is no need for the expression to be listed in a dictionary. This makes the automatic processing of such question terms extremely difficult.

Chapter 3

Difference questions in QA

Having investigated the nature of difference questions and answers in some detail, this chapter will discuss difference questions in question answering. A first paragraph will look at previous work in the area. Then, an overview of the task of dealing with difference questions in open-domain QA will be provided.

3.1 Previous work

3.1.1 Open-domain vs. closed-domain QA

In Question Answering a distinction is commonly made between open-domain and closed-domain systems. While closed-domain systems generally assume that all the information necessary to answer a question is contained in a structured database, open-domain systems have to cope with large collections of unstructured texts.

There are advantages to both: while open-domain systems are able to deal with a wider variety of topics, the answers of closed-domain systems are more often correct and therefore more reliable. Nevertheless, there was a shift away from the traditional closed-domain systems to open-domain ones in the 1980s. This was because open-domain systems were considered more useful because of their potential of answering a wide variety of questions.

An important research impetus into open-domain systems is provided by the annual TREC conference. Even though more and more question types have been included

in the TREC QA competition over the years, no research has yet been carried out to investigate difference questions (or comparisons in general). However, there have been attempts to deal with difference questions in closed-domain systems.

3.1.2 The TEXT system

The first QA system that explicitly dealt with difference questions was Kathleen McKeown's TEXT system, developed in 1985. Its purpose was "to generate text in response to a limited class of questions about the structure of a military database" (McKeown, 1985). The class of questions includes the following: definitions (e.g. "What is a frigate?"), requests for information (e.g. "What do you know about submarines?") and difference questions ("What is the difference between an ocean escort and a cruiser?", all examples taken from McKeown 1985).

There are four main modules in the TEXT system: a semantic processor, a schema selector, a schema filler and a tactical component. As a first step in processing a question, the TEXT system chooses a set of possible schemata which can be used for the answer. Then, the semantic processor produces a pool of knowledge that may be relevant for the answer (referred to as "relevant knowledge pool"). The schema selector next chooses a single schema from the set of possible schemata, which is then filled by the schema filler and a message is constructed. As a last step, this ordered message is passed on to the tactical component, which translates it into English by making use of a functional grammar (McKeown, 1985).

The most crucial step in answering a particular question in TEXT is to decide which information should be included in the relevant knowledge pool. While the technique used to partition the knowledge base is fairly simple for definitions and information questions, the method used for difference questions is "slightly more complicated" (McKeown, 1985). For definitions and information questions the area around the questioned object is simply sectioned off. For difference questions, however, "the kind of information that is included in the relevant knowledge pool is dependent upon the conceptual closeness of the two entities in question" (McKeown, 1985). In other words, TEXT takes into account that different degrees of semantic similarity require different answers, as was discussed in detail in the previous chapter. But how does TEXT

determine how similar two items are?

In order to determine the conceptual closeness of a pair, the hierarchical structure of the knowledge base is exploited. All entities in the knowledge base are part of a so-called generalization hierarchy, which includes superordinates of entities that ”share common features and can be grouped together as a class” (McKeown, 1985). For example, the entities SHIP and SUBMARINE have the superordinate WATER-VEHICLE, and both WEAPON and PROJECTILE are generalised as DESTRUCTIVE-DEVICE.

The TEXT system distinguishes three different categories of closeness: *very close*, *very different*, and *class difference* (which represents an intermediate stage) (McKeown, 1985). Category membership is determined by the position of the entities in the generalization hierarchy: elements are classified as *very close* if they are siblings in the hierarchy, while *very different* elements have a common ancestor in the hierarchy which is too high up to provide useful information. Elements in the *class difference* category are somewhere between (i.e. they share a common ancestor but are not particularly close). The information TEXT includes in the relevant knowledge base depends on the category of the entities. McKeown argues that answers should be more detailed when listing differences for very similar items, while a discussion of differences for very different entities could be endless. Therefore, for a dissimilar pair only the ”most salient distinction between the two” is considered of importance, which is expressed by their ”generic class membership” (McKeown, 1985). That this is indeed natural is supported by the answers in the questionnaire (Chapter 2): the differences between *fish* and *bicycle* are compared by contrasting generic terms such as *animal* or *vehicle*.

Placed in the context of the alignment-based model of similarity discussed above, one could say that the TEXT system only takes alignable differences into account. If the entities are too dissimilar, there are no alignable differences, and the system resorts to generic class membership.

3.1.3 The SHAKEN system

A relatively recent approach to dealing with difference questions in closed domains is described by Nicholson and Forbus (2002). They have embedded a comparison system

within their knowledge-based system SHAKEN which can deal with questions of the form ”How are X and Y similar?” and ”How are X and Y different?”, given that X and Y are two concepts that are known to the system (Nicholson and Forbus, 2002). Their purpose was to provide domain experts building up a knowledge base with a means to test what their system understands (knowledge capture). Their techniques rely on insights of the structure-mapping theory, namely that comparisons are alignment-based. The algorithms proposed by them use a system called structure-mapping engine (SME) as their comparison mechanism (Nicholson and Forbus, 2002). In order to answer difference questions a so-called ”case construction” technique is used to ”extract a relevant subset of knowledge about the concepts to be compared from the knowledge base” (Nicholson and Forbus, 2002). This can be compared to the task of creating the relevant knowledge pool in the TEXT system. When the results of the comparison have been computed, they are simplified by summarisation techniques and arranged in an order that is assumed to be most natural for the user. This is a crucial step, where insights of the structure-mapping theory come in: Nicholson and Forbus state that ”in summarizing differences, since alignable differences are more salient we present them first” (Nicholson and Forbus, 2002). They mention property differences as an important kind of alignable difference, ”where two entities that play similar roles are of different types” (Nicholson and Forbus, 2002). Property differences are computed by investigating the attributes that hold for corresponding entities. Non-alignable differences are those statements that are ”true in one description but not the other, with no correspondences in common” (Nicholson and Forbus, 2002), and they are listed last.

An evaluation of the system by human assessors revealed the following problems (Nicholson and Forbus, 2002): Firstly, the users were presented with too much information, which made it difficult for them to find the information they wanted. Secondly, differences that were wrong got included in the results. And finally, similarities that were expected by the users were not listed. Although one might expect that answering difference questions in closed-domains is not too complicated due to the availability of a structured database, these results show that there are serious problems.

On the whole one can conclude that dealing with difference questions in closed-domain systems involves some difficulties, but is possible due to the existence of a

structured database. The obvious disadvantage, however, is that questions can only be asked about a limited domain.

3.2 Difference questions in open-domain QA

The task of a common open-domain QA system is to answer an input question by using a knowledge base, which normally consists of a large collection of texts in natural language. QA systems typically consist of several processing stages. First, the input question has to be analysed and the knowledge base preprocessed. Then, documents relevant for the question are identified and retrieved. Next, an analysis of these documents is carried out. The final two steps are extracting an answer and generating a response which is returned to the user. The following sections will provide an overview of implementing an open-domain difference questions system. As the scope of this paper does not permit an in-depth discussion of the various issues, the overview presented is only very rough.

3.2.1 Analysis of the question

The first step in answering difference questions is to make sense of the question asked. The section on ”real difference questions” (2.4) provided an overview of the kinds of questions the system will have to deal with. Clearly, there are many ways of asking for the difference between items, and the classification given in 2.4 is by no means exhaustive (consider e.g. questions of the form ”How do you distinguish between?” or ”In what ways are A and B different?”). Therefore, one could argue that as a first step a question has to be identified as a difference question. Assuming that this has already been carried out (e.g. by simple pattern matching), two major tasks remain: firstly, the question terms need to be extracted from the question, and secondly, their senses have to be identified.

Let us assume that the question includes the pattern ”difference(s) between A and B” (which is the most common). An extraction of the question terms A and B is fairly straightforward for simplex question terms, as the word occurring between ”between” and ”and” is A, and the word after ”and” is B (assuming that nothing else follows).

However, the situation proves to be more difficult for complex terms. First of all, each question term can be realised as a whole syntactic phrase, such as an NP or VP (e.g. ”What is the difference between living in Ireland and living in the States?”[1]. Even when leaving the other categories aside, the processing of NP question terms is still not straightforward. As discussed in 2.4, this is due to the problem of coordination ambiguity, and an algorithm will have to be implemented to deal with this issue.

Once the question terms are extracted, their senses have to be identified. In other words, the two sub-questions ”What is A?” and ”What is B?” need to be answered (cf. 2.3). Again, this task is more complicated for complex terms than for simplex terms. After analysing the question, the problem of finding an answer can be tackled. In the following, I will present a discussion of possible knowledge bases from which answers could be derived.

3.2.2 Answers and knowledge bases

The choice of resources is vital for the task of answering difference questions. Three kinds of collections offer themselves for the task: lexical databases/ontologies, computerised encyclopaedias, and the Internet. I will discuss the advantages and disadvantages of each of these possibilities in turn. In particular, I will consider WordNet as an example of a lexical database, and the Wikipedia as an online encyclopaedia.

3.2.2.1 WordNet

The WordNet project came into existence at Princeton University’s Cognitive Science Laboratory in 1985. The current version, 2.1, contains almost 150,000 English nouns, verbs, adjectives, and adverbs, which are organized into sets of synonyms[2]. Each synonym set represents a lexicalised concept, and short glosses are provided for each set. In addition, semantic relations such as antonymy, hypernymy, or meronymy link the sets together. The structural organisation of the database makes WordNet a popular resource for researchers working in the area of natural language processing, as it

[1] `http://www.irishemigrant.com/article.asp?iCategoryID=164&iArticleID=12981`
[2] http://wordnet.princeton.edu/man/wnstats.7WN

”provides a more effective combination of traditional lexicographic information and modern computing” than common machine-readable dictionaries (Miller, 1995).

As most difference questions involve nouns (see Chapter 2), WordNet is particularly attractive for the task of answering difference questions, because all nouns are part of a so-called ”IS-A”-hierarchy (or hypernym hierarchy). Recall that McKeown’s TEXT system uses a similar hierarchy (the generalization hierarchy) to answer difference questions. However, her system has major advantages over the present one. Firstly, the entities in her database are all unambiguous, i.e. there is a 1:1 relation between entities and words. In WordNet, many word forms are ambiguous and are therefore encountered at different locations within the hierarchy, depending on their senses. This means that a disambiguation procedure is required in WordNet.

A second advantage of TEXT is that all entities have a structured description which makes a comparison of common and different features a straightforward task. WordNet does not provide such a consistently structured description of entities. Apart from the semantic relations that hold between synsets, the only information available to investigate commonalities and differences are the glosses, which are generally very brief. This makes an analysis of differences between two concepts a difficult task, especially in terms of alignable and unalignable differences. A search for commonalities and alignable differences could be conducted by gloss overlap: words (apart from stop words) that occur in both glosses are likely to represent common features. Another idea would be to make use of the meronymy relation, the ”PART-OF” relation. How commonalities and alignable differences can be identified, and how unalignable differences could be extracted remains open for future work.

A third point where WordNet is at a disadvantage concerns answering difference questions. The TEXT system relies on the conceptual closeness of a pair, which is determined by distance in the hierarchy (cf. discussion in 3.1.2). However, only ”very similar” entities qualify for a comparison in terms of commonalities and differences. These entities are direct neighbours in the hierarchy, and can be compared as they share the same features. Such an approach does not work for comparisons within the WordNet hierarchy, as it should be possible to compare concepts further away from each other. An additional complication concerns the network density in WordNet. Jiang

and Conrath note that ”it can be observed that the densities in different parts of the hierarchy are higher than others” (Jiang and Conrath, 1997), like e.g. in the plant/flora section of WordNet. Jiangh and Conrath conclude that ”the greater the density, the closer the distance between the nodes” (Jiang and Conrath, 1997). Therefore, decisions based on the conceptual closeness of two items (as in TEXT) have to be very carefully considered. Note, however, that WordNet could do very well on comparing ”very different” concepts by contrasting their generic class membership, just as in TEXT.

Despite these drawbacks, the WordNet hierarchy has one major advantage over the TEXT hierarchy: it is open-domain and aims to include all members of the open word classes, which means that theoretically it is capable to deal with a great range of difference questions.

3.2.2.2 Wikipedia

The Wikipedia is a free, collaborative online encyclopedia. The project started in 2001 and is constantly updated by its users. It has some major advantages over WordNet. Firstly, there is a wealth of information available. In April 2005, the English language edition contained over 500,000 articles [3], most of which describe nouns. A comparison of the coverage of question terms shows that Wikipedia covers about 95% of the simplex terms, while only 73% are listed in WordNet (see Appendix A.2 and A.3). Secondly, most of the definitions provided in the Wikipedia are fairly long and provide an exhaustive or near-exhaustive definition of a queried concept. This often involves comparing it to related concepts. Consider for example the beginning of the definition of frog:

```
Frogs are amphibians in the Order Anura, which includes
frogs and toads. The term "frog" is a popular name for
animals that look like toads, but are generally more
slender, have a less warty and dry skin, have long
legs adapted for leaping and are more aquatic. It has
no meaning in animal systematics, since many anuran
families include both "frogs" and "toads".
```

[3] http://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia

Some of the words are hyperlinked (e.g. *amphibians*, *frogs*, *toads*). As these words are likely to be important for a definition of the term, it might be useful to compare the hyperlinked words of one question term (e.g. *frog*) to the hyperlinked words occurring in the definition of the other question term (*toad*).

However, the Wikipedia also has a number of disadvantages. It lacks the semantic relations between concepts that are provided in WordNet. Furthermore, even though co-edited by different authors, the information provided is not as reliable as in WordNet, and may consequently lead to wrong results.[4]. It appears that the strengths of Wikipedia are the weaknesses of WordNet. Hence, it is worthwhile to consider an approach that uses a combination of the two. The last section in this chapter deals with the Internet as a resource for the task of answering difference questions.

3.2.2.3 The Internet

Lita et al., who have conducted a resource analysis for question answering, note that a growing number of QA systems use the web as a resource (Lita and Nyberg, 2004). It has been shown that ”since the Web is orders of magnitude larger than local corpora, answers occur frequently in simple contexts, which is more conducive to retrieval and extraction of correct, confident answers” (Lita and Nyberg, 2004). The advantage of using the web for answering difference questions lies in the fact that questions may already have been answered somewhere on the web. Therefore, a search engine such as Google may be used to search for a string like ”difference between a frog and a toad”, as was done in section 3. However, while the wealth of information present on the web is a sure advantage to the Wikipedia or WordNet, it has the problem of being very unreliable. Most webpages are not peer-reviewed and often express a single person’s point of view (this is also an issue in Wikipedia, although there are guidelines).

[4]For an exhaustive account of the advantages and disadvantages of the Wikipedia, see the websites `http://en.wikipedia.org/wiki/Wikipedia:Why_Wikipedia_is_not_so_great` and `http://www.cooldictionary.com/words/Wikipedia:Why-Wikipedia-is-so-great.wikipedia`

3.3 Conclusion

This chapter has presented previous work on difference questions in closed domains, and some important issues concerning an implementation in an open domain were described. However, the overview was only a rough one, and there are clearly many more issues to be considered. These, however, remain open for further research. The remaining part of this work will deal with the first step in answering difference questions in an open domain, which is to ”make sense” of the question.

Chapter 4

Making sense of difference questions I

4.1 The task

Suppose someone asked the following question:

a) *What's the difference between a bat and a mouse?*

Native speakers of English will answer the question by explaining the differences between a small nocturnal mammal and a small animal that belongs to the family of rodents. However, now consider this question:

b) *What's the difference between a bat and a racquet?*

It is clear that now the word *bat* no longer refers to an animal, but to a club used in ball games. While humans are generally very efficient at disambiguating word senses of polysemous words, this poses a major problem for computers.

As George A. Miller observes, ”polysemy is a major barrier for many systems that accept natural language input” (Miller, 1995). In particular, he notes that ”in information retrieval, a query intended to elicit material relevant to one sense of a polysemous word may elicit unwanted material relevant to other senses of that word” (Miller, 1995). Likewise, if we ask a question answering system to list the differences between a bat and a mouse, we do not want results that compare the baseball bat to the little grey rodent, and neither do we want to know about the differences between the nocturnal mammal and the computer mouse.

The first step in answering difference questions automatically is therefore to ”make

sense" of the question asked. WordNet offers the possibility of doing this systematically, as it "lists the alternatives from which choices must be made" (Miller, 1995). WordNet is the main resource for word sense disambiguation tasks, and its glosses have been used to semantically annotate the SemCor corpus[1]. The remainder of this work will investigate how well WordNet can perform as a resource for disambiguating the question terms of difference questions.

4.2 Preliminaries

4.2.1 Assumptions

The preceding chapters offered a glimpse of the complex nature of difference questions. The tables in Appendix A.2 and A.3 illustrate the fact that many of the question terms asked by real users are not covered in WordNet, which is the case for complex terms in particular. However, this fact will be ignored for the time being. What is of interest is: if terms are listed, how can one go about identifying their senses?

As WordNet does not contain all question terms, some assumptions have to be made upon which the approach taken in this task can be based. Prager et al. (2001), who investigate the use of WordNet for answering "What-is" questions, also make such assumptions, e.g. that the question term to be defined is available in WordNet. In the following, I will make the following assumptions about the question terms A and B:

1. *A and B have been identified in a previous processing step*
2. *A and B are simplex nouns*
3. *WordNet lists the intended senses of A and B*
4. *At least one of the terms has more than one sense (otherwise task is trivial)*

Assumption 1 states that the question terms have already been identified, i.e. the preprocessing step of extracting the terms has already been carried out. Assumption 2 restricts the terms to simplex nouns. Theoretically, complex nouns can also be included, but since WordNet treats all its entries as single strings, complex words are disregarded for practical reasons.

[1] `http://www.cs.unt.edu/~rada/downloads.html\#semcor`

4.2.2 Main Idea

The main idea behind the attempt of ”making sense of difference questions” described in this chapter is based upon the psycholinguistic findings discussed in Chapter 2: the more similar two items are, the more acceptable the difference question involving the pair. This implies that if the question terms have more than one sense, the intended senses are those that maximise the similarity of the two words.

In her TEXT system, Kathleen McKeown determines the similarity of two entities by looking at the position of their closest ”common ancestor” (McKeown, 1985). This idea will be adopted in the present approach. For each pairwise combination of senses the closest common ancestor is determined. The two senses which share the closest common ancestor overall are then taken to be the intended senses. This approach represents our intuitions about difference questions, and has been discussed in various studies (Lesk, 1986). Before discussing the approach in more detail, the next section will provide a description of the data sets that were used in the task.

4.3 Data

4.3.1 Development set

In order to create a disambiguation system for difference questions, a development question set has to be provided. The purpose of this set is to test the accuracy of the system during development. The errors produced are analysed and used to refine the system step by step. It is therefore of importance to choose a question set that is representative of the task.

As the system relies on the choice of the lowest common hypernyms of words, the decision was made to base the choice of question terms on the potential amount of common hypernyms between the question terms. This amount (k) is determined by multiplying the number of senses of word 1 (m) with the number of senses of word 2 (n), i.e. $k = n \cdot m.$[2]. Noun pairs were generated according to intuition, and the value k was determined for each pair. 50 pairs were chosen altogether, 30 of which have a

[2]As will be shown later, k expresses an approximation and not the actual value

value k smaller than 10 (inclusive) and the remaining 20 have more than 10 possible lowest common hypernyms. This appeared to constitute a fair representation of the task. The complete set is displayed in Appendix B.1.

Unfortunately, it is not possible to create a development set that systematically covers all possible cases. One could argue that the set is biased in that it does not take dissimilar pairs into account. It would also have been a possibility to base the set on the similarity of the noun pairs, e.g. by using the Miller-Charles question set described in Chapter 2. However, this approach would also have two problems: firstly, for dissimilar pairs it is more difficult to establish the intended senses, and secondly, the task may be too easy since many of the nouns only have one or two senses.

After studying in depth some of the nouns and their sense relations, I identified some cases that were potentially problematic for ambiguation, and therefore I decided to include at least one example each in the development set:

- The question terms A and B have no hypernym in common (e.g. *infinitive*/*imperative*)
- A and B have senses that are synonymous (e.g. *tin*/*can*, *rock*/*stone*)
- A and B have senses where one is a hypernym of the other (e.g. *boa* is a hypernym of *python*)

For an evaluation of the algorithm during development, each noun in the set was assigned exactly one sense by two independent human annotators. They were furthermore asked to provide a confidence value for each decision (between A = confident and D = very unsure). This was done so that mistakes produced by the system could be judged to be more or less grave. Cases where their opinion differed were further reviewed by a third judge. The judgements of the annotators are also displayed in the table in Appendix B.1 (columns J1/J2 = Judge 1/2, C1/C2 = confidence value 1/2).

4.3.2 Test set

After the development is finished, the algorithm is tested on an unseen set of difference questions to provide an evaluation of how well it performs. This set should be

representative of what the algorithm is to expect in the real world. Therefore, a random selection of 50 questions was extracted from the question log described in 2.4, all of which fulfil the assumptions stated in 4.2.1. The selection was random in order to ensure that the questions were as varied as possible and not clustered around a time a particular user posed his or her queries.

For the evaluation, two independent judges assigned senses to the question terms. Again, they were required to choose exactly one sense, even when they felt that there was more than one possible solution, and indicated a confidence value as described above. The cases where the judges disagreed were presented to a third person, who made a final decision. Since these were in favour of Judge 1, the J1 column will generally be considered the gold standard. The annotators agree on 86% (43/50) of the test set. The complete set alongside the senses assigned by Judge 1 and Judge 2 are displayed in Appendix B.2.

4.4 Approach

4.4.1 Formalisation of the task

First, let us formalise the idea described in Chapter 4.2.2. Let w_1 and w_2 be the question terms of a difference question, and $S_1 = \{s_{11}, ..., s_{1m}\}$ be the set of senses listed in WordNet for w_1, and $S_2 = \{s_{21}, ..., s_{2n}\}$ be the set of senses in WordNet for w_2 (i.e. w_1 has m senses, and w_2 has n senses).

We are looking to determine the pair of intended senses (s_{word1}, s_{word2}). For each sense pair (s_{1i}, s_{2j}), let $H_{ij} = h_{ij1}, ..., h_{ijk}$ be the set of common hypernyms of that sense pair $(1 \leq i \leq m, 1 \leq j \leq n)$. Let further d be a distance metric that measures the distance of a hypernym from its sense pair, with $d : H_{ij} \longrightarrow \mathbf{N}$. The lowest common hypernym (*lch*) of the sense pair (s_{1i}, s_{2j}) can then be calculated as :

$(*)$

$$lch(s_{1i}, s_{2j}) = \arg\min_{h \in H_{ij}} d(h).$$

This lowest common hypernym will be referred to as ”local” *lch*. The pair of intended senses (s_{word1}, s_{word2}) is the pair with the lowest common hypernym *globally,*

i.e. if

$$\text{LCH} = \{lch(s_{1i}, s_{2j})\}(1 \leq i \leq m, 1 \leq j \leq n)$$

is the list of all local lowest common hypernyms, the intended senses are the pair (s_{word1}, s_{word2}) with

$(**)$

$$(s_{word1}, s_{word2}) = \arg\min_{x \in LCH} d(x).$$

Therefore, the following issues need to be dealt with:

1. Define and implement the distance metric d,
2. Implement a function that creates the list H_{ij} of common hypernyms of a sense pair (Function 2),
3. Implement a function that chooses the smallest element of a list of hypernyms according to the distance metric d (Function 3).

Note that Function 3 can be used twice: first, to choose the local *lch*, and then also to choose the global *lch*.

4.4.2 A first implementation

For the implementation of the algorithm I decided to use a Python interface to the WordNet database (version 2.0) offered by Sourceforge[3]. PyWordNet offers a number of possibilities to query the database for lexical relationships between word senses. The function `meet` (in wn_tools) appeared to be particularly attractive for my task. According to Oliver Steele, who developed PyWordNet, "the *meet* of two items is their most subordinate common concept" [4]. It takes as input two word senses and returns their lowest common hypernym. Steele illustrates its use with the following examples (where *dog* has been defined as the first sense of the noun *dog* in WordNet, and *cat*[0] refers to the first sense of the noun *cat*):

```
>>> meet(dog, cat[0])
{noun: carnivore}
>>> meet(dog, N['person'][0])
```

[3] `http://sourceforge.net/projects/pywordnet`
[4] `http://osteele.com/projects/pywordnet/examples.html`

```
{noun: life form, organism, being, living thing}
>>> meet(N['thought'][0], N['belief'][0])
{noun: content, cognitive content, mental object}
```

The list LCH (see above) could therefore easily be created by applying `meet` to every possible sense combination available for the two words, and then putting the results in a list. This saves one processing step, since the list of common hypernyms H_{ij} does not have to be created. What remains is defining the distance function *d* and an implementation of Function 3 (choosing the global lowest common hypernym).

However, during the implementation I discovered a major problem with Steele's `meet` function. When calculating the `meet` of the nouns *cup* and *mug*, the program output the following results for senses 3 (*cup*) and 4 (*mug*):

```
>>> concept1 = N['cup'][2]
>>> concept2 = N['mug'][3]

>>> meet(concept1, concept2)
{noun: artifact, artefact}
>>> meet(concept2, concept1)
{noun: container}
```

This is clearly wrong, as a function that calculates the lowest common hypernym of two concepts should produce the same results irrespective of the order in which the concepts are entered. The problem lies in the fact that concepts in WordNet are sometimes allocated more than one hypernym, as illustrated by the hypernym tree of *cup*/3:

```
['cup' in {noun: cup},
 [{noun: crockery, dishware},
  [{noun: tableware},
   [{noun: ware},
    [{noun: article},
     [{noun: artifact, artefact},
      [{noun: object, physical object}, [{noun: entity}]],
      [{noun: whole, whole thing, unit},
       [{noun: object, physical object}, [{noun: entity}]]]]]]]],
 [{noun: container},
  [{noun: instrumentality, instrumentation},
```

```
   [{noun: artifact, artefact},
    [{noun: object, physical object}, [{noun: entity}]],
    [{noun: whole, whole thing, unit},
     [{noun: object, physical object}, [{noun: entity}]]]]]]]
```

The tree of *mug*/4, on the other hand, looks like this:

```
['mug' in {noun: mug},
 [{noun: drinking vessel},
  [{noun: vessel},
   [{noun: container},
    [{noun: instrumentality, instrumentation},
     [{noun: artifact, artefact},
      [{noun: object, physical object}, [{noun: entity}]],
      [{noun: whole, whole thing, unit},
       [{noun: object, physical object}, [{noun: entity}]]]]]]]]]
```

Obviously, the correct *lch* is *container*, as it is 3 steps away from the concept `mug` and a direct hypernym of `cup`. `meet` does not recognise the direct hypernym. A closer look at its implementation reveals why this is the case:

```
def meet(a, b, pointerType=HYPERNYM):

    return (intersection(closure(a, pointerType), closure(b, pointerType))
    + [None])[0]
```

The functions `closure` and `intersection` (wn_tools) require further explanation. `Closure` returns ”the transitive closure of source under the pointerType relationship” (wn_tools), which in this case is the hypernymy relation. The following lines show the source code of `closure`, and the lists produced when applying `closure` to the nouns *cup*/3 and *mug*/4:

```
def closure(source, pointerType, accumulator=None):
    if isinstance(source, Word):
        return reduce(union, map(lambda s, t=pointerType:tree(s,t),
source.getSenses()))
    _requireSource(source)
    if accumulator is None:
        accumulator = []
    if source not in accumulator:
```

```
        accumulator.append(source)
        for target in source.pointerTargets(pointerType):
            closure(target, pointerType, accumulator)
    return accumulator

>>> closure(word1, HYPERNYM)
['cup' in {noun: cup}, {noun: crockery, dishware}, {noun: tableware},
{noun: ware}, {noun: article}, {noun: artifact, artefact},
{noun: object, physical object}, {noun: entity},
{noun: whole, whole thing, unit}, {noun: container},
{noun: instrumentality, instrumentation}]

>>> closure(word2, HYPERNYM)
['mug' in {noun: mug}, {noun: drinking vessel}, {noun: vessel},
{noun: container}, {noun: instrumentality, instrumentation},
{noun: artifact, artefact}, {noun: object, physical object},
{noun: entity}, {noun: whole, whole thing, unit}]
```

The function `intersection` then returns the intersection of these two lists, and the lowest common hypernym (or "most subordinate common concept", as Steele calls it) is taken to be the first element of the resulting list (indicated by the [0] in `meet`). Depending on whether concept 1 or concept 2 is entered as a first argument of intersection, the elements of the intersection list occur in a different order:

```
>>> intersection(closure(word1, HYPERNYM), closure(word2, HYPERNYM))
[{noun: artifact, artefact}, {noun: object, physical object},
{noun: entity}, {noun: whole, whole thing, unit}, {noun: container},
{noun: instrumentality, instrumentation}]

>>> intersection(closure(word2, HYPERNYM), closure(word1, HYPERNYM))
[{noun: container}, {noun: instrumentality, instrumentation},
{noun: artifact, artefact}, {noun: object, physical object},
{noun: entity}, {noun: whole, whole thing, unit}]
```

As a result, the pair concept 1/concept 2 is assigned {`artifact, artefact`} as their *lch*, while the result for concept 2/concept 1 is {`container`} (which is the correct *lch*). The crux of the problem is that `closure` does not take the structure of the hypernym tree into account. It simply flattens the tree into a list and then traverses it from left to right, not noticing that the tree has two branches. Instead of going depth-first down

the branches of the tree, the concepts should actually be listed breadth-first. However, this leads to the next problem: when moving along the tree breadth-first, should this be from left to right, or right to left? This is a crucial point, as the elements are put into a list according to the order in which they are encountered, and it is the first element of this list that is chosen by `meet` as the *lch*.

As a matter of fact, the problem of branching hypernym trees was not taken into account in the formalisation of the task. The left hand side of equation (*) should therefore be a list, which most of the time contains one *lch*, but sometimes, like in the case of *cup*/3 and *mug*/4, contains more than one:

I performed an error analysis of the `meet` function by applying it to the noun pairs of the development set, and found that the number of cases where `meet(concept1, concept2)` does not equal `meet(concept2, concept1)` is actually very small. This might explain why the problem went unnoticed so far. There are 8 sense pairs where `meet` produces wrong results:

- *rug*/1 and *mat*/3
- *town*/1 and *city*/2
- *town* /2 and *city*/1
- *tablet*/4 and *pill*/3
- *magazine*/1 and *bo*ok/1
- *magazine*/1 and *book*/2
- *magazine*/1 and *book*/4
- *mug*/4 and *cup*/3

I was furthermore able to show that the `meet` function is incomplete 53 times for the development set (i.e. `meet` only calculates one *lch* for a sense pair, but in fact there is more than one).

Considering that the overall number of sense pairs for the 50 development set questions is 925, one could argue that an error rate of 0.0086% (8/925) is neglectable. However, the consequences of one mistake can be profound, as the example *cup*/3 *mug*/4 illustrates. Therefore, a better implementation of the task is in order, which will be discussed in the next section.

4.4.3 A better implementation

Returning to the original plan of implementing the algorithm, the first step was to define the distance measure. This is the function *d* that maps the set of common hypernyms of two given concepts to the set of natural numbers. There are a number of ways in which this can be done. I decided to use the most straightforward method, which also reflects our intuitions about the distance of the hypernym from its concepts: the shortest path from concept to concept via the hypernym node, which is calculated as

$d(h) = n_1 + n_2$,

where n_1 is the number of edges between concept 1 and the hypernym and n_2 is the number of edges between concept 2 and the hypernym (note that this function is not injective, as it may map more than one argument to the same natural number. The fact that there can be more than one lowest common hypernym for two concepts is a consequence of this).

In order to find the lowest common hypernym between two concepts, all their common hypernyms are put into a list and the lowest one is chosen according to the distance measure. For this purpose, the list of hypernyms of a given concept needs to be ”indexed”, i.e. for each concept a list is created that contains all its hypernyms and their distances from the concept. The implementation of the index function makes use of the `wn_tools` function `flatten`, which flattens the hypernym trees step by step. The items that emerge in every new round are added to an IndexList, together with their index, which is incremented by 1 in each round. This is what the function produces for *cup*/3 and *mug*/4:

```
IndexList1 (cup/3):
[[1, {noun: crockery, dishware}], [1, {noun: container}],
[2, {noun: tableware}], [2, {noun: instrumentality,
instrumentation}], [3, {noun: ware}], [3, {noun: artifact,
artefact}], [4, {noun: article}], [4, {noun: object,
physical object}], [4, {noun: whole, whole thing, unit}],
[5, {noun: artifact, artefact}], [5, {noun: entity}],
[5, {noun: object, physical object}], [6, {noun: object,
physical object}], [6, {noun: whole, whole thing, unit}],
[6, {noun: entity}], [7, {noun: entity}],
[7, {noun: object, physical object}], [8, {noun: entity}]]
```

```
IndexList2 (mug/4):
[[1, {noun: drinking vessel}], [2, {noun: vessel}],
[3, {noun: container}], [4, {noun: instrumentality,
instrumentation}], [5, {noun: artifact, artefact}],
[6, {noun: object, physical object}],
[6, {noun: whole, whole thing, unit}], [7, {noun: entity}],
[7, {noun: object, physical object}], [8, {noun: entity}]]
```

The list of common hypernyms H (referred to as H_{ij} in 4.4.1) can then be generated for each sense pair by going through every possible combination and adding each hypernym that occurs in both IndexLists to H.

Next, an implementation of Function 3 is needed. This function chooses the smallest element of a list of hypernyms according to the distance metric d. I implemented a function `chooseLCH` which performs this task. Essentially it is a common ”find minimum” function which takes as input a list of lists containing tuples of the form [sense1, distance1, sense2, distance2, hypernym], where the first and third element are sense numbers, the second and fourth the distances n_1 and n_2 (obtained by the `index` function), and the last element is the common hypernym h of word1/sense1 and word2/sense2. The distances n_1 and n_2 are added up to obtain $d(h)$, and the tuple with the shortest path is put into a list (referred to as `first_lch_list` in the code). This is the step that Steele does not take into account: by putting the smallest element into a list, we ensure that all alternatives are kept if there is more than one shortest path. All the local *lch*'s are put into a list (the LCH list in the formalisation above), and the `chooseCCH` is applied once again to choose the ”global” lowest common hypernym (which again is a list that may contain more than one element).

Finally, an evaluation of the results is carried out, where the calculated senses are compared to the correct senses (as labelled by the human judges), and the results are presented in terms of correct and incorrect answers. Three different cases are distinguished: firstly, if there is more than one *lch* in the final list, secondly, if there is no *lch* at all, or thirdly, if there is exactly one. If there is more than one *lch*, another decision process is needed. Again, it was opted to apply a simple measure, namely the depth of the hypernym within the WordNet hierarchy. The function `chooseLowest` was implemented to calculate the distance of the *lch*'s from its root node in WordNet (which is one of 11 unique beginners, cf. Miller (1995)). For this purpose the hypernym lists of

the *lch*'s are indexed, and the hypernym with the highest index is chosen as the root. The function then chooses the *lch* with the hypernym with the highest index, since the index describes the depth of the *lch* in the hierarchy. If the index is the same for some (or all) members of the final list, then by default the function chooses the first element of the final list, and then the corresponding senses are compared to the human labelled ones.

If there is no *lch* at all for two words, then the system chooses by default the first sense for each word. This is motivated by the fact that WordNet often lists the most frequent sense of a word first. If two words do not share a common hypernym, they are likely to be dissimilar, and therefore, the most frequent sense is likely to be the intended sense (although experiments would have to be carried out to support this claim). And finally, the last possibility is straightforward. If the system has calculated exactly one *lch*, then its corresponding senses are compared to the human judgements.

The final results of the system for the development set are displayed in Appendix B.3 (column marked "R"), alongside the human judgements. The last column furthermore displays the *lch*'s calculated by the system (0 indicates no *lch*). The sense pairs highlighted in bold in the R column are the ones that differ from the human judgements, and those that are italicised differ from the votes of Judge 2 only. Taking Judge 1 as a benchmark, the accuracy of the system is 66% (33 out of 50 questions correct). When taking Judge 2 as the benchmark, the system performs considerably worse with 58% (29 out of 50 questions correct). Per question term, the results are as follows: taking Judge 1 as benchmark, 75% (75 out of 100) of the question terms were labelled correctly by the system, and with Judge 2 as the gold standard, the system's accuracy is 70%. The results by question term are not necessarily representative though, as some of the question terms only have one sense in WordNet anyway. While there is without doubt still room for improvement especially in the case of question terms that have more than one *lch*, it was decided to stop the development here, as the system was at a stage where the potential of the *lch* approach could be investigated in more detail.

4.5 Results

The performance of the system on the test set is quite similar to its performance on the development set: treating Judge 1 as the benchmark, the accuracy of the algorithm is 64% (32 out of 50 questions correct), and for Judge 2 it is 54% (27 correct). As discussed above, the results for Judge 1 will be considered the gold standard. Taking into account that the accuracy for human judgements is 86% (the judges agreed on 43 of the 50 test set questions), the system can be said to achieve about 74.4% of the performance of human annotators (taking Judge 1 as the gold standard). An analysis in terms of question terms also shows similar results to the ones discussed for the development set: compared to Judge 1's results, the system assigns 74% of the senses correctly, and 70% compared to Judge 2. The results are displayed in Appendix B.4. In the following, an error analysis of the results is carried out, which will show up some major problems of the present approach. The analysis will distinguish three different cases: firstly, the question terms have no *lch* at all; secondly, they have exactly one *lch*; and thirdly, there is more than one *lch* to choose from.

There are only three noun pairs that do not share a common hypernym: (23) *marketing*/*sales*, (24) *condensation*/*dampness* and (25) *weather*/*climate*. As discussed above, the system assigns as a default the first senses of the words. This works well in the case of *weather*/*climate*, but not for *condensation*/*dampness*, where the correct sense pair should be 3/1. For *marketing*/*sales* the result is in accordance with Judge 1's choice, but not with Judge 2, who believes 2/1 to be the correct sense pair. The fact that only 6% of all questions have no hypernym in common supports the assumption that humans tend to compare similar rather than dissimilar items.

The second group contains noun pairs that share exactly one lowest common hypernym. Containing almost two thirds of all noun pairs (31/50), this group does not only constitute the largest set of nouns, but it also contains the most interesting cases in this discussion, as it illustrates how well the *lch* approach works. Table 3 shows that 23 out of 31 questions (74.2%) are labelled correctly according to Judge 1 (but only 61.3% for Judge 2). A result of 74.2% is quite good, and a look at the errors produced in this group reveals that most of them are not grave mistakes. The pairs labelled incorrectly compared to Judge 1's results are:

(4) *art*/*science*
(13) *apostle*/*disciple*
(19) *spread*/*interest*
(32) *project*/*product*
(33) *life*/*death*
(34) *enlightenment*/*romanticism*
(37) *plant*/*tree*
(38) *license*/*certification*

The four pairs that do not match Judge 2's assessment are:

(6) *probation*/*parole*
(12) *lease*/*licence*
(39) *original*/*copy*
(49) *science*/*technology*

What is striking is that (with the exception of *plant*/*tree*) all concepts are abstract, as opposed to other members of the set such as *star*/*planet*, *wasp*/*hornet*, *shark*/*ray*, or *margarine*/*butter*. Furthermore, the confidence values of the human assessors reveal that they were uncertain about at least half of the pairs. In addition, on top of not agreeing on pairs 6, 12, 39 and 49, they also did not agree on pairs 32 and 33. Since one should not expect a machine to outperform a human, these mistakes should be considered as not grave. A closer look at the system's choices shows that (with the exception of two pairs) they are not unreasonable. In the case of the pair *art*/*science*, the system chooses senses 3 and 2 which share the hypernym {superior skill}, instead of the *lch* of the intended sense pair 2/1 {creation, creative activity}. The system further suggests sense 3 for *apostle*, "(New Testament) one of the original 12 disciples chosen by Christ to preach his gospel", and sense 1 for *disciple* "someone who believes and helps to spread the doctrine of another", which does not reflect the human judgements of the "best sense", but is definitely not wrong. The case of *life* and *death* further supports this view. Determining the senses of *life* and *death* is a philosophical issue, and whether the senses chosen by the system (*life*/13 and *death*/2) are better or worse than the human judgements is a matter of debate:

> 13. (12) life – (the organic phenomenon that distinguishes living organisms from nonliving ones; ”there is no life on the moon”)
>
> 2. (311) death – (the permanent end of all life functions in an organism or part of an organism; ”the animal died a painful death”)

Only two results produced in this group are totally wrong: the pair *plant*/*tree*, and to a lesser degree *spread*/*interest*. For *plant*/*tree* the system computes the *lch* {actor, histrion, player, thespian, role player} and the following senses:

> 4. plant – (an actor situated in the audience whose acting is rehearsed but seems spontaneous to the audience)
>
> 3. Tree, Sir Herbert Beerbohm Tree – (English actor and theatrical producer noted for his lavish productions of Shakespeare (1853-1917))

The correct senses of *plant* and *tree* have as their *lch* the synset {organism, being}, which is a direct hypernym of *plant*, but four steps away from *tree*. Between *tree* and {organism, being} are the synsets {woody plant}, {vascular plant}, and {plant}, which illustrates the problem of network density discussed in Section 3.2.2.1.

On the whole, the results produced by the systems in this group are satisfactory. However, it has also become clear that a fair evaluation of answers in terms of correct/incorrect is not always possible, as WordNet’s senses are very fine grained. This is in accordance with the finding that disambiguation using WordNet’s senses is a lot more difficult than disambiguating to the level of homographs (Resnik, 1998). Therefore, an implementation of a difference QA system should be able to accept more than one sense pair as correct, as there may be more than one correct answer.

Turning to the last group of noun pairs, namely those sharing more than one *lch*, the situation is as follows. Of the 16 members of this group, 11 could be disambiguated correctly, as their correct *lch* (as judged by the human assessors) is included in the final LCH list. They are (including their number in the table in Appendix B.4 and their correct *lch*):

> (5) *stress*/*depression* (psychological state), (7) *male*/*female* (animal), (8) *newton*/*kilogram* (unit of measurement), (9) *marketing*/*advertising* (commerce), (10) *star*/*planet* (celestial body), (11) *poetry*/*prose* (writing style), (22) *boy*/*girl* (person, individual), (28) *marketing*/*selling* (commerce), (35) *democrat*/*republican* (legend/fable), (42) *college*/*university* (educational institution).

Consequently, another 22% (11/50) of questions have the potential of being labelled correctly if their correct *lch* is identified. The success of this depends on the implementation of a decision making function, which in our case is the `chooseLowest` function described above, which chooses the *lch* according to its depth in the hierarchy. At a first glance, this method appears to work reasonably well: of the 11 questions, 7 are identified correctly. However, a second glance reveals that the decisions are more or less random: many of the noun pairs have two or more identical *lch*'s, which consequently have the same depth in the taxonomy. For example, the nouns *marketing* and *advertising* share the *lch* {commerce} twice, once for sense pair 1/2, and another time for 2/2. In cases like this, `chooseLowest` picks the first sense pair listed (as default), which in this case is 1/2. In fact, of the 7 correctly labelled pairs only four are not chosen in a random fashion, and, likewise, of the three incorrectly labelled noun pairs, two have been assigned the default sense pair. Obviously, the `chooseLowest` function is a very ineffective way of making a decision.

Looking at the results of the approach on the whole, it seems that the *lch* method works well for cases where there is exactly one *lch* (23/31 correct). This supports the assumption that similarity acts as a strong indicator for the correct senses. However, the remaining 27 noun pairs (i.e. over half of them) can be considered to be assigned senses more or less randomly. Therefore, the conclusion of this section must be that a disambiguation approach based on *lch*'s and simple edge counting is not good enough, and that other factors have to be taken into account. The next chapter will discuss a range of methods, all of which have in common that they base their decision on the similarity of the items.

Chapter 5

Making Sense of Difference Questions II

5.1 WordNet::Similarity

This chapter investigates to what degree measures of semantic similarity can be useful in disambiguating the question terms of ”What’s the difference” questions. The previous chapter showed that the concept of similarity works well as an indicator for a correct sense choice, but that a similarity measure based on lowest common hypernyms and edge counting alone is not sufficient. In this chapter, some other measures of similarity will be discussed and their usefulness for the disambiguation task will be assessed.

As Budanitsky and Hirst note, ”the problem of formalizing and quantifying the intuitive notion of similarity has a long history in philosophy, psychology, and artificial intelligence, and many different perspectives have been suggested” (Budanitsky and Hirst, 2001). Apart from the theoretical discussions in psychology presented in Chapter 2, there has also been a lot of research into computational measures of similarity. In the late nineties, `WordNet::Similarity` was published, which is a freely available software package currently containing six measures of similarity (and three measures of semantic relatedness), all of which are implemented in Perl[1]. The measures take as

[1] `WordNet::Similarity` is available from http://sourceforge.net/projects/wn-similarity.

input two words (or concepts) listed in the WordNet 2.0 database, and return a numerical value for their similarity (or relatedness). The next section will give an overview of the measures available in this package.

5.2 Overview of the measures

Generally, there are two major approaches to measuring similarity in `WordNet::Similarity`. The first approach is based on path lengths between the concepts, an idea familiar to us from the disambiguation algorithm described in Chapter 4. Three path-based measures are available: LCH[2] (Leacock and Chodorow, 1998), WUP (Wu and Palmer, 1994) and PATH. The LCH measure determines the shortest path between two concepts and ”scales that value by the maximum path length in the *is-a* hierarchy in which they occur” (Pedersen et al., 2004). WUP calculates the depth of the ”least common subsumer”, or *lch*, to the root node in terms of path length, and scales this value by the sum of the path lengths from the two concepts to the root. The PATH measure is the simplest of the three, as it calculates the similarity value as the inverse of the shortest path between two concepts. All three measures share basic ideas with the approach suggested in Chapter 4, which can be considered a mixture of PATH and WUP.

The second approach is based on information content, which is ”a corpus-based measure of the specificity of a concept” (Pedersen et al., 2004). The three measures available in `WordNet::Similarity` are LIN (Lin, 1998), JCN (Jiang and Conrath, 1997) and RES (Resnik, 1998). Resnik was the first to suggest such an approach in 1995, stating that ”intuitively, one key to the similarity of two concepts is the extent to which they share information in common, indicated in an IS-A taxonomy by a highly specific concept that subsumes them both” (Resnik, 1998). In `WordNet::Similarity`, the information content of concepts is by default derived from the sense-tagged SemCor corpus, but there are options available that also allow using untagged corpora such as the Brown corpus or the BNC. Like RES, LIN and JCN both measure the information content of the least common subsumer, but they also take into account the information content of the two concepts themselves.

[2]Not to be confused with the abbreviation for ”Lowest Common Hypernym”.

Apart from the similarity measures, there are also measures of relatedness available in `WordNet::Similarity`. Semantic relatedness must be distinguished from semantic similarity. As Banerjee and Pedersen observe, a common type of relatedness between two concepts is when ”one is a more general instance of the other (e.g., a car is a kind of vehicle) or because one is a part of another (e.g., a tire is a part of a car)” (Banerjee and Pedersen, 2003). It is clear that while *tires* and *cars* are related, they can not be classified as similar. In `WordNet::Similarity`, three relatedness measures are available: HSO (Hirst and St-Onge, 1998), LESK (Banerjee and Pedersen, 2003), and VECTOR (Patwardhan et al., 2003). The HSO measure assumes that two concepts are related if their senses in Wordnet are connected by a path that is as short as possible, and it also assumes that WordNet relations have direction. The VECTOR and LESK measures, on the other hand, are not path-based but focus on word-overlaps in the defining glosses of the concepts.

5.3 Disambiguating senses with WordNet::Similarity

The measures available in `WordNet::Similarity` open up new possibilities of disambiguating the senses of difference question terms. As they are intended for determining the similarity of noun pairs, a disambiguation procedure (based on the assumptions stated in 4.2.1) can simply be based on the similarity values calculated for all possible sense pairs: the sense pair with the highest similarity (or relatedness) index is declared the winner. To investigate how well the individual measures perform, I first ran all three information-content based similarity measures (LIN, JCN and RES) on both the test set and the development set. I then also applied HSO, VECTOR and LESK to both sets in order to see how measures of relatedness perform compared to measures of similarity. Before discussing the results, one section will be devoted to the results obtained by the path-based measures LCH and WUP.

In an evaluation of the measures, Judge 1 will be treated as the gold standard. Results for each member of the test set (and development set) will be presented in a table containing:

- The noun pairs (columns WORD 1 and WORD 2)

- The labels of the two human judges (columns J1 and J2)
- The first three results of the measure as ranked by the system, where each of the three columns (Result 1, Result 2, Result 3) contains the calculated sense pair and their calculated similarity value (in brackets)
- An evaluation in terms of correct/incorrect.

The entry *yes* in the last column indicates that the first result is the correct result according to Judge 1. If *yes* is set in brackets *(yes)*, this means that there is no clear winner, i.e. there are further results with the same similarity value as the first. Likewise, *no* indicates that the first result output by the system is not the correct result, and *(no)* means that while the correct result is not ranked as the first, it nevertheless has the same value as the first result.

5.4 Results

5.4.1 LCH and WUP

The results for the path-based measures are similar to those obtained in Chapter 4 (cf. also Appendix B.3 and B.4):

	Test set				Development set			
Measure	yes	(yes)	no	(no)	yes	(yes)	no	(no)
LCH	30	3	20	7	29	2	21	3
WUP	32	2	18	1	32	1	18	0

LCH achieves 30 correct answers for the test set, and 29 correct answers for the development set (for more details see Appendix C.1.1 and C.1.2 for LCH, and C.2.1 and C.2.2 for WUP). WUP reaches 32 correct answers in both sets. The formulas by which LCH calculates the results is:

$$\text{sim}_{LCH}(c_1, c_2) = max[-log(\frac{ShortestLength(c_1, c_2)}{(2 \cdot D)})]$$

$ShortestLength(c_1, c_2)$ is "the shortest path length (having minimum number of nodes) between the two concepts and D is the maximum depth of the taxonomy (distance of the farthest node from the root node"(Patwardhan et al., 2003). The values are

on a logarithmic scale, and thus the minimum value of 0 is reached when

$$ShortestLength(c_1, c_2) = 2 \cdot D$$

which is the case when concepts 1 and 2 are at the very bottom of the hierarchy, and share no common hypernym apart from the ”dummy” root node inserted at the top of the hierarchy. The 0 value is reached neither in the test set nor in the development set, which reflects the fact that totally dissimilar items are generally not compared. The maximum value depends on the depth of the hierarchy. For the noun hierarchy the depth is 16, and therefore the maximum value of the formula is 3.5835 (which is assumed for concepts which are in the same synset, e.g. pair 41 in the test set, *frog*/*toad*).

WUP is quite similar to LCH, as it also uses the shortest path. Instead of counting the number of concepts on that shortest path, it counts the edges, which results in a ”shortest path” value that is always 1 less than the path length in nodes. The shortest path between the ”least common subsumer” (henceforth *lcs*) and the root node is ”scaled by the sum of the path lengths from the individual concepts to the root” (Pedersen et al., 2004). Therefore, the maximum value is 1 (when the concepts are in the same synset), and the minimum value is 0.

As the measures are based on an idea similar to the one implemented in Chapter 4, an in-depth error analysis will not be carried out. As expected, both measures suffer from the same problems as encountered previously, namely that there may be more than one *lcs* (which results in identical values for the results), or none at all. Incidentally, these are the only results where the measures differ from the one described in Chapter 4. However, their decision making function is no better than `chooseLowest`, as illustrated by the results of 60% (LCH) and 64% (WUP) (as opposed to 64% for the Chapter 4 algorithm). Furthermore, the three pairs in the test set with no common hypernym (pairs 23, 24, and 25) are also disambiguated incorrectly by both measures.

5.4.2 RES

The measure proposed by Resnik is based on the information content of the lowest common subsumer. His idea is to associate probabilities *p(c)* with each concept *c* in

the taxonomy, which in `WordNet::Similarity` are derived from frequencies in the sense-tagged SemCor corpus. As each concept counts as an instance of its hypernym concepts, the probability *p* is "monotonically nondecreasing as one moves up the taxonomy: if c1 IS-A c2, then $p(c_1) \leq p(c_2)$" (Resnik, 1998). Resnik notes that as a consequence of this the probability of the dummy top node in the hierarchy is 1.

In Resnik's approach, the similarity of two concepts c_1 and c_2 is then taken to be equal to the maximum information content (calculated as the negative log likelihood) achieved by any of the common hypernyms of the concepts:

$$\text{sim}_{RES}(c_1, c_2) = IC(lcs(c_1, c_2)),$$

The *lcs* that maximises the above equation is referred to as the "most informative subsumer" (Resnik 1998). This is in accordance with our intuition: the lower the probability of a word, the greater its information content. The minimum similarity value is 0 (when the most informative subsumer is the dummy root node), while the maximum value is unbounded (as it depends on the frequency of the concept).

The advantage of this approach is that it can deal with concepts that have multiple lowest common hypernyms: the *lcs* with the highest information content is chosen. However, this only works if the *lcs*'s are different. Patwardhan et al. note that "the potential limitation of this approach is that quite a few concepts might have the same least common subsumer, and would have identical values assigned to them" (Patwardhan et al., 2003), a problem which also occurs in the path-based approach. Moreover, the information based approach has another great practical disadvantage. Higher information content means lower probability, and lower probability means lower frequency, and considering the size of the SemCor corpus one can easily see that the approach is prone to a "zero-probability" problem. Resnik himself works with the Brown corpus (Resnik 1998) to avoid this problem. However, using an untagged corpus means trading in the zero-probability problem for skewed results, as the frequency estimates are likely to be inaccurate.

The results of the Resnik measure for both question sets are disappointing (cf. Appendix C.3.1 and C.3.2):

	Test set				Development set			
Measure	yes	(yes)	no	(no)	yes	(yes)	no	(no)
RES	29	3	21	5	29	3	21	5

The measure has an accuracy of 58% for both sets, which is slightly worse than the path-based measures. Surprisingly enough, this is not due to the zero-probability problem, as most nouns in the two sets are words of higher frequency. The only pairs whose first results are 0 are *marketing*/*sales*, *condensation*/*dampness*, *weather*/*climate*, and *meiosis*/*mitosis* for the test set (i.e. pairs 23, 24, 25 and 48), and *infinitive*/*imperative*, *software*/*hardware*, and *gaelic*/*welsh* for the development set (pairs 1, 2 and 7). Of these, only *meiosis*/*mitosis* and *garlic*/*welsh* are a consequence of the zero probability problem, as neither the terms themselves nor their ”most informative subsumers” (*cell division* and *Celtic*) occurs in SemCor. The results for the other pairs, on the other hand, are correct, as their most informative subsumer is the root node, whose information content is indeed 0.

An investigation of the 21 errors produced by RES on the test set shows that for only 6 of them the correct answer is included within the first three results produced by the measure (i.e. less than one third). It can be concluded that the RES measure is not suitable for this problem. However, it may come in useful at another stage of the difference questions task: Resnik has shown that his measure produces good results for resolving coordination ambiguity, which represents a problem for complex question terms (as discussed in Chapter 2.4).

5.4.3 JCN and LIN

The results of JCN and LIN are even more disappointing: JCN has accuracies of 50% on the test set and 56% on the development set, while LIN is even worse with 46% and 56% (see Appendix C.4.1 and C.4.2 for JCN, and C.5.1 and C.5.2 for LIN).

	Test set				Development set			
Measure	yes	(yes)	no	(no)	yes	(yes)	no	(no)
JCN	25	4	25	7	28	3	22	10
LIN	23	4	27	12	28	4	22	11

What is striking is that there is now a big problem with zero probabilities. All *(yes)* and *(no)* results in the table represent cases where all sense pairs receive the same similarity value, namely 0. Apart from the pairs 23, 24 and 25 (test) and 1 and 2 (development) which are correctly assigned a 0 similarity, there are now also pairs like *probation*/*parole*, *apostle*/*disciple* or *wasp*/*hornet* with a zero probability. Taking this fact into consideration, the performance of the measures is even worse: when not counting the *(yes)* cases, the overall accuracy of JCN is only 42% (test) and 50% (development), while LIN drops to 38% (test) and 48% (development). Such results are clearly not satisfying, and it seems safe to conclude that the use of information-based measures are inappropriate for the task of disambiguating senses of difference question terms.

A final word is in order to explain the zero values of the two measures. In `WordNet::Similarity`, JCN is calculated as:

$$\text{distance}(c1,c2) = IC(c_1) + IC(c_2) - 2 \cdot IC(lcs(c_1,c_2))$$
$$\text{sim}_{JCN}(c_1,c_2) = \frac{1}{\text{distance}(c_1,c_2)},$$

where c_1 and c_2 are the two concepts, IC is the information content of the concept, and $lcs(c_1,c_2)$ is the lowest common subsumer of c_1 and c_2. I investigated the behaviour of JCN for the zero-value pairs of the test set (6, 13, 16, 29, 31, 43, 44, 48 and 50) by switching on the `trace` function and found that the similarity of the concepts is set to 0 when $\text{distance}(c_1,c_2)$ becomes negative. The most common cause of this is when one of the concepts has an IC of 0, or when both are 0 while $IC(lcs) \geq 0$. On the other hand, it is also problematic when $distance(c_1,c_2)$ equals 0, as $sim(c_1,c_2)$ is then undefined (or infinite). This is the case for concepts within the same synset, i.e. $IC(c_1) = IC(c_2) = IC(lcs(c_1,c_2))$, and for cases where $IC(c_1) = IC(c_2) = IC(lcs(c_1,c_2)) = 0$ (e.g. for the pair *meiosis*/*mitosis*). The first case is dealt with by placing an upper bound on similarity, which is represented by the value 29590099.4292 (see e.g. Result 1 for pair 41 in the test set, *frog*/*toad*). In the second case, the similarity value is set to 0, as for the RES measure.

The LIN measure calculates similarity with this formula:

$$\text{sim}_{LIN}(c_1, c_2) = 2 \cdot \frac{IC(lcs(c_1, c_2))}{IC(c_1) + IC(c_2)}$$

There are 16 zero-probability cases in LIN, including all 9 JCN cases listed above. For LIN, the similarity value is 0 when the IC value of one of the concepts is 0 (like in the case of JCN), but in addition the measure is also 0 when $IC(lcs(c_1, c_2)) = 0$.

5.4.4 HSO

Hirst-St.Onge is the first measure of semantic relatedness to be discussed. What further distinguishes it from the other measures discussed is the fact that it ”considers many other relations in WordNet and is not restricted to nouns” (Patwardhan et al., 2003). The idea behind it is that ”two lexicalized concepts are semantically close if their WordNet synsets are connected by a path that is not too long and that ’does not change direction too often’” (Budanitsky and Hirst, 2001). The formula used to calculate the relatedness value is:

$$\text{rel}_{HSO}(c_1, c_2) = C - path\_length - k \cdot d,$$

where C and K are constants, and d is the number of changes of path direction. The relatedness value is 0 if no such path exists (Budanitsky and Hirst, 2001). Obviously, HSO is an extremely complex measure, which does not pay off in the case of our test sets (cf. Appendix C.6.1 and C.6.2):

	Test set				Development set			
Measure	yes	(yes)	no	(no)	yes	(yes)	no	(no)
HSO	29	8	21	10	32	5	18	8

Values of 58% and 64% are better than the information based results, but still no improvement to the path-based measures. What is striking is the high amount of ”undecided” results (*(yes)* and *(no)*) in the test set, 18 noun pairs altogether are affected by this (including many cases where the first three results are all 0). Subtracting the number of undecided results, it leaves only 21 clearly correct results. As 42% is again a very poor a result, HSO is also rejected as a disambiguation measure.

5.4.5 VECTOR

The VECTOR measure is another complex relatedness measure. The approach used, however, is totally different from the ones discussed so far. Instead of making use of the semantic relations that hold between concepts in WordNet, VECTOR bases its decision on the text of the glosses of these concepts. This is done by treating the WordNet glosses as a text corpus, and creating a co-occurrence matrix from this corpus. Each open-class word in that corpus is assigned a context vector, and each WordNet gloss is represented by a gloss vector, which is ”the average of all the context vectors of the words found in the gloss” (Pedersen et al., 2004). The relatedness between two concepts is calculated by determining the cosine between the gloss vectors representing the concepts. The results are as follows (Appendix C.7.1 and C.7.2):

	Test set				Development set			
Measure	yes	(yes)	no	(no)	yes	(yes)	no	(no)
VECTOR	28	0	22	0	25	0	25	0

The measure is therefore no better than the ones discussed so far. When interpreting the results, it has to be borne in mind that there are no cases that are ”undecided”. Furthermore, there are no ”zero” results, which is an advantage to the measures discussed so far. The results are also disappointing in that only half of the incorrectly disambiguated noun pairs include the correct result within the top three. The next section will describe the last of the WordNet::Similarity measures, which is also based on WordNet glosses, and which provides a pleasant surprise.

5.4.6 LESK

The LESK measure is based on an algorithm for word sense disambiguation first proposed by Lesk (1986). This algorithm ”assigns a sense to a target word in a given context by comparing the glosses of its various senses with those of the other words in the context” (Banerjee and Pedersen, 2003). The correct sense is then calculated as the one whose gloss has the most words in common with the glosses of its context words. While Lesk’s algorithm only considers glosses of the target word and the

context words, the ”extended gloss overlap” measure, proposed by Banerjee and Pedersen (2003) and implemented in `WordNet::Similarity`, also takes into account the glosses of concepts related to these words. Their reason for extending the algorithm in this way is the fact WordNet glosses are generally very short and ”do not provide sufficient vocabulary to make fine grained distinctions in relatedness” (Banerjee and Pedersen, 2003).

Benerjee and Pedersen's scoring mechanism also differs from the original Lesk algorithm. While Lesk assigned the same scores to single and multiple word overlaps, the extended measure does differentiate between these cases. This is motivated by the Zipfian relationship that holds between lengths of phrases and their frequencies in large text corpora: the longer a phrase is, the lower its frequency in the corpus (Banerjee and Pedersen, 2003). Therefore, the (adapted) LESK measure assigns an *n* word overlap a score of n^2, and the scores of all overlaps of the two glosses (and also of their related glosses) are added up to obtain the final result.

The results produced by this measure are astonishing. The accuracy for both sets is at least 10% higher than for any of the other measures (see also Appendix C.8.1 and C.8.2):

	Test set				Development set			
Measure	yes	(yes)	no	(no)	yes	(yes)	no	(no)
LESK	37	0	13	0	38	0	12	0

What is more, an analysis of the incorrect results reveals that for the test set, 9 out of 13 incorrectly labelled noun pairs include the correct result among the top three (and 9 out of 12 for the development set). In fact, 7 of them are ranked second in both cases:

Test set:
(3) *heat*/*temperature*, (5) *stress*/*depression*, (7) *male*/*female*, (11) *poetry*/*prose*, (24) *condensation*/*dampness*, (35) *democrat*/*republican*, (38) *license*/*certification*;

Development set:
(7) *gaelic*/*welsh*, (30) *tin*/*can*, (34) *rock*/*stone*, (38) *atom*/*molecule*, (39) *solid*/*liquid*, (46) *pea*/*bean*, (50) *mug*/*cup*). An investigation of answers proposed by the measure shows that most often they are also acceptable. Consider e.g. the pair (7) *male*/*female*,

where LESK proposes the senses 2 ”a person who belongs to the sex that cannot have babies” and 2 ”a person who belongs to the sex that can have babies”, while the Judges agreed it was 1 ”an animal that produces gametes (spermatozoa) that can fertilize female gametes (ova)” and 1 ”an animal that produces gametes (ova) that can be fertilized by male gametes (spermatozoa)”. However, Judge 1 is not entirely confident in this decision (as indicated by a confidence value of B). In fact, a closer look at the six test set pairs and the 5 development test pairs that were *not* ranked first or second shows that they are mainly cases where the judges themselves either disagree or are unsure. Therefore, they cannot be considered as grave mistakes.

5.5 Conclusion

This chapter investigated the use of measures of semantic similarity and relatedness to disambiguate the question terms of difference questions. As it turns out, it is a measure of semantic relatedness that performs best on both test sets, namely the LESK measure. Its use of gloss overlaps is more efficient than both the edge-based approaches and the measures that take information content into account. In the context of word sense disambiguation 76% is a very good result. However, it has to be borne in mind that there are a number of terms where a disambiguation is not necessary, as they only have one sense. On the other hand, the evaluation considers only complete sets of question terms, i.e. a question is counted as wrong even if one of the question terms has been disambiguated correctly. The results should therefore only be compared to other word sense diambiguation tasks by bearing these issues in mind.

Chapter 6

Conclusion

This study set out to provide an overview of answering comparison questions in QA, with a special focus on ”What’s the difference” questions. During the course of the research it emerged that implementing a system that can deal with difference questions is an extremely complicated task.

The first part of this work investigated the nature of difference questions. Studies of comparison processes in psychology provided a suitable framework for discussing difference questions, but they also revealed how complex human processing of comparisons is. An automated system has to be able to imitate these processing steps. This means that the system first has to identify the items which are to be compared and provide a definition of them. Then it needs to establish what commonalities they share, before finally being able to look for the differences. In closed-domain systems, an implementation of these steps is straightforward due to the existence of a structured knowledge base. A further advantage of closed-domain systems is that they do not have to deal with the polysemy inherent in natural language, since every item in the knowledge base is well defined.

In an open-domain system, on the other hand, even the first processing step, namely defining the question terms, is a difficult issue. This was demonstrated in the second part of this work. Several measures were applied to disambiguate the senses of the question terms. The best performance (76%) was achieved by a measure of semantic relatedness based on gloss overlap in WordNet. While this is a good result in the area of word sense disambiguation, the approach was based on various assumptions, for ex-

ample that the correct senses are included in WordNet. Also, so far only simplex nouns have been taken into account, while an investigation of real user questions showed that over half of the difference questions asked on the Internet involve complex question terms.

It is clear that a lot of research remains to be done before a complete open-domain difference questions system can be presented to users. A major problem will be how the so-called ”alignable differences” can be identified, which are those kinds of differences that are psychologically most salient to the user. Neither WordNet nor the Wikipedia are structured enough to make this a straightforward task. Research into which semantic relations are most important in comparisons should be carried out, and into whether it is possible to enrich a given knowledge base with these relations. It could furthermore be investigated in what way multi-knowledge base approaches could be beneficial (e.g. combining WordNet with Wikipedia or the Internet).

On the whole, answering comparison questions in open domains is a fascinating though difficult endeavour. The path to a properly functioning system is bound to be long and rocky, but the need for it is apparent when looking at the great numbers of difference questions asked on the Internet. And when the system finally works properly, it will of course be able to provide the user with the only correct answer to the question ”What’s the difference between a fish and a bicycle?”:

They can both swim, except for the bicycle.

Appendix A

Appendix A

A.1 Questionnaire

A.2 Simplex terms in WordNet and Wikipedia

A.3 Complex terms in WordNet and Wikipedia

Appendix B

Appendix B

B.1 Development set

B.2 Test set

B.3 Development set results

B.4 Test set results

Appendix C

Appendix C

C.1 LCH

C.1.1 LCH: Test set results

C.1.2 LCH: Development set results

C.2 WUP

C.2.1 WUP: Test set results

C.2.2 WUP: Development set results

C.3 RES

C.3.1 RES: Test set results

C.3.2 RES: Development set results

C.4 JCN

C.4.1 JCN: Test set results

C.4.2 JCN: Development set results

C.5 LIN

C.5.1 LIN: Test set results

C.5.2 LIN: Development set results

C.6 HSO

C.6.1 HSO: Test set results

C.6.2 HSO: Development set results

C.7 VECTOR

C.7.1 VECTOR: Test set results

C.7.2 VECTOR: Development set results

C.8 LESK

C.8.1 LESK: Test set results

C.8.2 LESK: Development set results

Bibliography

Banerjee, S. and Pedersen, T. (2003). Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 805–810.

Budanitsky, A. and Hirst, G. (2001). Semantic distance in WordNet: an experimental, application-oriented evaluation of five measures. In *Workshop on the WordNet and Other Lexical Resources. The North American Chapter of the Association of Computational Linguistics*.

Gentner, D. and Markman, A. (1994). Structural alignment in comparison: no difference without similarity. *Psychological Science*, 5(3):152–158.

Hahn, U. (2003). Similarity. *Encyclopedia of Cognitive Science*.

Hirst, G. and St-Onge, D. (1998). *Lexical chains as representation of context for the detection and correction of malapropisms*, pages 305–332. MIT Press, Cambridge, MA.

James, H. (1890/1950). *The principles of psychology*. Dover: New York. Original work published 1890.

Jiang, J. J. and Conrath, D. W. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International Conference on Research in Computational Linguistics*.

Leacock, C. and Chodorow, M. (1998). *Combining local context and WordNet similarity for word sense identification*, pages 265–283. MIT Press, Cambridge, MA.

Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone. In *Proceedings of SIGDOC '86*.

Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*.

Lipka, L. (2002). *English Lexicology*. Narr, Tubingen.

Lita, L.V., H. W. and Nyberg, E. (2004). Resource analysis for question answering. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.

Markman, A. and Gentner, D. (1996). Commonalities and differences in similarity comparisons. *Memory and Cognition*, 24:235–249.

McKeown, K. (1985). *Text generation: using discourse strategies and focus constraints to generate natural language text*. CUP, New York.

Miller, G. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

Nicholson, S. and Forbus, K. (2002). Answering comparison questions in shaken: A progress report. In *AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases*, Palo Alto, CA. AAAI.

Patwardhan, S., Banerjee, S., and Pedersen, T. (2003). Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*.

Pedersen, T., Patwardhan, S., and Michelizzi, J. (2004). Wordnet::similarity - measuring the relatedness of concepts. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (Intelligent System Demo)*.

Prager, J., Chu-Carroll, J., and Czuba, K. (2001). Use of WordNet hypernyms for answering What-Is questions. In *Proceedings of the TREC-2002 Conference*.

Quine, W. (1969). *Ontological Relativity and Other Essays*. Columbia University Press, New York.

Resnik, P. (1998). Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*.

Rubenstein, H. and Goodenough, J. (1965). Contextual correlates of synonymy. *Communications of the ACM*, 8:627–633.

Wu, Z. and Palmer, M. (1994). Verb semantics and lexical selection. *32nd Annual Meeting of the Association for Computational Linguistics*.