Classification of Speech Acts in Discussion Threads

ERIK GUSTAFSON

Master’s Thesis in Computer Science (30 ECTS credits) at the School of Computer Science and Engineering
Royal Institute of Technology year 2011
Supervisor at CSC was Ola Knutsson
Examiner was Stefan Arnborg

TRITA-CSC-E 2011:128
ISRN-KTH/CSC/E--11/128--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication
KTH CSC
SE-100 44 Stockholm, Sweden
URL: www.kth.se/csc
I’d like to thank TN20 inc for providing an interesting and challenging thesis subject. Thank should also be given to the staff at NADA especially Ola Knutson.
Abstract

In this thesis the prospect of classifying parts of internet discussion threads as different speech acts, such as questions and answers, are examined. The approach is using different machine-learning algorithms such as decision trees and support vector machines (SVM) coupled with different kind of feature selections.

Most of the work was focused on finding an appropriate set of features that would be on the right level of complexity for determining the speech act. Methods that are examined are N-grams of part of speech, word patterns, ratios of common words and various statistical features.

The result showed that with a relatively small training set it was possible to get fairly good results (about 60% correct classifications) depending on the conditions. It was also found that there are quite big performance differences for individual speech acts and classifiers.
Abstract

Klassificering av talhandlingar i diskussionstrådar

Det här exjobbet undersöker hur klassificering av delar diskussionstrådar i olika talhandlingar som frågor och svar kan göras. Metoden som används är olika typer av maskininlärningsalgoritmer, exempelvis beslutsträd och Support Vector Machine (SVM) tillsammans med olika typer av egenskaper.

Den största delen av arbetet fokuserades på att hitta en bra uppsättning egenskaper som ger rätt nivå av komplexitet för att klassificerar talhandlingar. Typer av egenskaper som undersöks är N-gram av ordklasser, ord-mallar, ratios av ord och olika statistiska mät.

Resultatet visade att med en relativt liten mängd träningsdata kunde man få hyfsade resultat (omkring 60% korrekt klassificeringar) beroende på förutsättningarna. Ytterligare ett resultat var att relativt stora prestanda skillnader fanns mellan olika typer av tal handlingar och klassificerare.
Contents

1 Introduction ................................................. 1
   1.1 Definitions ........................................... 2
   1.2 Goals .................................................. 2
   1.3 Limitations ............................................ 3
   1.4 Thesis outline .......................................... 3

2 Previous work ............................................. 5
   2.1 Text categorization ..................................... 5
   2.2 Genre Detection .......................................... 6
   2.3 Question detection ....................................... 7
   2.4 Sentiment analysis ...................................... 7
   2.5 Subjectivity analysis .................................... 8

3 Problems, corpus and performance ......................... 9
   3.1 Selection of speech acts ............................... 9
   3.2 Performance ............................................ 9
      3.2.1 Recall and Precision .................................. 9
      3.2.2 Domain transferability .............................. 10
      3.2.3 Ease of Training .................................... 10
      3.2.4 Amount of Text Needed for Classification .......... 10
      3.2.5 Robustness .......................................... 11
      3.2.6 Computation resources .............................. 11
   3.3 Corpus .................................................. 11
   3.4 Comparision of results to others ..................... 12

4 Algorithms and Design .................................... 15
   4.1 General setup of classifier ............................ 15
      4.1.1 Training module .................................... 15
      4.1.2 Analysis module .................................... 15
   4.2 Feature Selection ....................................... 16
   4.3 Different feature sets ................................... 17
      4.3.1 Words ............................................... 17
      4.3.2 N-grams ............................................. 17
## CONTENTS

4.3.3 Word classes ................................................. 18
4.3.4 Common Words .............................................. 18
4.3.5 Word Patterns ................................................ 18
4.3.6 Lexical Markers ............................................. 18
4.3.7 Statistical features .......................................... 19

4.4 Machine Learning Algorithms .................................. 19
4.4.1 Decision Trees .............................................. 19
4.4.2 Support Vector Machine .................................... 21
4.4.3 Bayesian networks .......................................... 21
4.4.4 Committee classifiers ....................................... 21

4.5 Discourse classifiers ............................................ 22
4.5.1 Markov Model ............................................... 22
4.5.2 Hidden Markov Model ....................................... 24

4.6 Chunking and subdividing of text ............................... 24

5 Implementation ..................................................... 25
5.1 Software package ................................................ 25
5.2 Java 1.5 from SUN .............................................. 25
5.2.1 Weka3 .......................................................... 25
5.2.2 Lingpipe ....................................................... 26
5.3 Chunking of text .................................................. 26
5.4 Feature Extractions .............................................. 26
5.4.1 Bag of Words ................................................ 26
5.4.2 Word Class N-grams ........................................ 27
5.4.3 Word Pattern ................................................ 27
5.4.4 Common Words ............................................. 28
5.4.5 Statistical features .......................................... 28
5.5 Other implementation issues ..................................... 29
5.5.1 Committee classifier implementation ....................... 29
5.5.2 Parameter optimization ..................................... 29
5.5.3 Optimizing the combination of feature selection and Machine Learning Algorithms ................. 30

6 Results ................................................................. 31
6.1 The classifiers ..................................................... 31
6.2 Results in a single domain ....................................... 32
6.3 Results over multiple domains .................................. 34
6.4 Results in unknown domains .................................... 34

7 Discussion and Conclusions ....................................... 39
7.1 Discussion ......................................................... 39
7.1.1 Performance of the classifiers ............................... 39
7.1.2 Differences between domains ............................... 41
7.1.3 Differences between the speech acts ......................... 41
### Explanations of words and acronyms

<table>
<thead>
<tr>
<th>Word</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corpus</td>
<td>A collection of texts. Commonly used for training and evaluation in Natural Language Processing</td>
</tr>
<tr>
<td>BNC</td>
<td>British National Corpus</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>ML</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>MM</td>
<td>Markov Model</td>
</tr>
<tr>
<td>NB</td>
<td>Naïve Bayes</td>
</tr>
<tr>
<td>NLP</td>
<td>Natural Language Processing</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>TC</td>
<td>Text Categorization</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Today due to the success of Internet, there are enormous amounts of information available on-line. One problem that arises is that the information on Internet is not ordered or sorted, which makes it very hard to find the right information. One of the big challenges for computer science and the Internet service companies is to find effective ways for users to find the information they want. To overcome this problem for regular web pages, Internet search engines such as Yahoo, AltaVista and Google tries to index the content of Internet to make it searchable with keyword searching. These search engines have become better and more efficient over recent years and are today among the most popular websites on Internet.

One area on Internet that contain lots of information are Internet discussion groups, a concept that originates from USENET newsgroups. These discussion groups often consists of people with similar interests that discuss topics related to that interest. These discussion groups are based on threads where a specific topic is discussed. A thread can sometimes contain tens, if not hundreds, of different messages. One problem for a user is to quickly identify what in a thread that are questions and answers. A service that could mark these types in a thread could facilitate browsing a thread and could improve searching in discussion threads by only including questions or answers in the search.

Some other examples how classification in type-of-text could be used to improve knowledge discovery:

**Improved search engines** Make search engines more effective by allowing the user to specify the type of text he or she is interested in. For example; only look in questions.

**Summarization of texts** Use the type of text classification to create better summarization of texts.

**More specialized NLP algorithm** By being able find the type of text one could choose a more specialized algorithm for the given task, hence getting better performance.
CHAPTER 1. INTRODUCTION

1.1 Definitions

When classifying a text it is needed as a prerequisite to determine at what level classification should be done. Should classification be done for entire texts, sections of the text or single sentences. What level is chosen affects the methods and strategy a lot. One of the requirements from my assigner was that the classifications should be discrete, each piece of text should have just one category assigned. With the basis of this requirement I came to use an object called speech act, that would fill my needs.

A speech act is defined as piece of text formed by the author with a specific intent such as greeting someone, asking a question or answering a question. The speech act is somewhat more akin to a reply in a discussion between two or more persons, than to a piece of more traditionally written text. Below there is an example of the transfer of a post in a discussion thread into a speech acts:

Hi

Yes molding in that way would be a workable approach.

I am curious as to the types of ceramic businesses you operate. If you have a shop that offers classes, a bisque wholesale business, or finish work. Also, where are you located.

Transfers into the following three speech acts:

Other

Hi

Answer Yes molding in that way would be a workable approach.

Question I am curious as to the types of ceramic businesses you operate. If you have a shop that offers classes, a bisque wholesale business, or finish work. Also, where are you located.

The definition worked well since this thesis concerns internet discussion threads. Discussion threads are usually informal and have a structure that can be described as a mix between verbal communication and traditional written communication. The term speech act is also used by Fung et al.\[7\] with the same meaning.

Even though the domain of the problem has been reduced to just discussions within Yahoo group using statistical methods will make it possible to change area of application. To do such a change would mean that a new set of training data would have to collected for optimal performance and depending on domain some optimizations might need to be added or removed.

1.2 Goals

These are the goals of this thesis:
1.3. LIMITATIONS

- Finding and identifying relevant and useful categories for the speech acts in internet discussion groups.
- Find algorithms that will be able to classify texts of the above categories with as good precision as possible.

1.3 Limitations

To narrow down the work the following limitations will be imposed:

- The only language that will be considered is American English.
- The approach will be to use statistical methods.
- Only texts from Yahoo Groups will be considered.
- Different machine learning algorithms and internal workings will not be described or discussed in detail, even though they will be frequently used (an excellent resource for machine learning algorithms is Sebastiani’s Machine Learning in Automated Text Categorization [23]).

1.4 Thesis outline

The rest of the thesis is laid out as follows:

Chapter 2 surveys earlier and related work that has been done by others and will be built upon in this thesis. In chapter 3 the problem is discussed more in detail along with definitions, performance measures and what corpus to use. Algorithms and heuristics that are used are described in detail in chapter 4. The actual implementation and how it is done is described in chapter 5. Chapter 6 contains information about the tests and results. In chapter 7, the results are evaluated and conclusions and further work in the area are discussed.
Chapter 2

Previous work

It is first in recent years that the need to categorize text according to their genre (article, letter, etc.) has been proposed. Karlgren et al. [4] proposed to improve web searches with genre classification and it was proposed to use it in summarization of texts. The work that have been done are genre classification of British National Corpus (BNC) [13] and newspaper articles from the Wall Street Journal Corpus[8]. Furthermore, specialized work has been put into identifying questions and answers in conversations, chat logs and email conversation.

This chapter will go through previous work that relates to the work in this thesis.

2.1 Text categorization

To automatically categorize a text according to some criteria or group with a computer is an area where a lot of work has been done over the years. Text Categorization (TC) was first used in the 60’s to automatically assign a text to a category based on its topic and content.

The early approaches used a rule based expert system that relied on humans for creating the rule set. These systems achieved good results in class with human experts and where reasonably efficient. The downside was the effort of building the rule set and the fact that even small expansions and changes in the area of use could require huge changes to be done to the rule set by a human experts.

From the beginning of the 1990s systems based on machine learning gained popularity. The techniques could perform almost as well as human experts and with the advantage that human experts were not needed as much for the development. Most common machine learning methods have been tried for text categorization with varying degrees of success. In his survey Sebastini [23] mentions Decision Tree classifier, Decision Rules classifiers that generates a rule set, regression methods based on static regressions models, Rocchio classifiers that builds a profile for the documents, neural networks, nearest neighbor algorithms and Support Vector Machines (SVM).

As a complement to the different machine learning techniques, a lot of work has
also been put into the identification of useful characteristics to train the classifier on. Some features used are N-grams of characters, words, part of speech tags, frequencies and ratios of different punctuation and part-of-speech tags.

### 2.2 Genre Detection

It should be noted before looking into genre detection that there exists no gold standard or universal corpus with defined genres to use as comparison, so the different result can not easily be translated to each other.

In the middle of the nineties scientists started to pick up interest in how to automatically detect genres of texts, at first the common machine learning methods where used. As genres can be seen as orthogonal to topics, classifiers that are successful for topic classification might not be as good at determining genres and therefore different techniques had to be used. One of the first attempts to classify text genres where made by Karlgren and Cutting [13]. They used counts of certain lexical features such as adverbs, characters and certain expressions to classify texts from British National Corpus with Discriminant Analysis. The results they achieved were between 70% and 95% correct classifications, depending on the setup and the number of categories tested.

Kessler et al. [14] used structural, lexical and character level properties, or cues. They trained neural networks and logistic regression classifiers on them and managed to achieve 79% precision with the aid of feature selection to avoid over fitting.

One problem that arises is when dealing with genre categorization on Internet is that in contrast to literature and newspapers there are no clearly defined genres. Roussinov et al. [20] tried to identify suitable genres. They managed to identify different genres that matched user needs, and then grouped them together in five major genres; Topics, Publications, Products, Educational material and FAQ. They also concluded that all users will not always agree to the assignment of genre, and that fuzzy genre limits shall be used.

Wolters and Kirsten [28] tried to classify text from the German LIMAS corpus in four different genres over five different domains. As features they used function words and part-of-speech frequencies. For classifier they used a k-Nearest-Neighbour algorithm. They managed to achieve a precision of over 80% but found that their corpus was to small to build a general classifier.

More work in the field of domain transferability was done by Finn et al. [8]. They tested different techniques for feature selection; bag-of-word techniques, part-of-speech statistics and handcrafted features. The findings where that handcrafted features and bag-of-word performed best in classification within a single domain, over multi domain it performed considerably worse than part-of-speech that achieved an success rate of between 70 and 90% correct classifications. Other interesting work was done by Stamaratos et al. [25] who used frequencies of the 50 most common words (extracted from BNC) to determine genre of texts in the Wall Street
2.3. QUESTION DETECTION

Journal Corpus (when doing topic classification these 50 words are often ignored). They managed to achieve an error rate of approximately 4% using four different genres. They also managed to increase their result by adding frequencies of punctuation marks. Santini [22] in a study used POS-trigrams to train a Naive Bayes classifier. She used ten different genres and managed to get a success rate of 79% correct classifications. One interesting conclusion from the results where that including punctuation in the trigrams in general didn’t improve performance, which was somewhat contrary to what Stamaratos et al. had found. Argamon et al. [3] tried to combine the use of both the most frequent words and part-of-speech trigrams on texts from New York times. They reached a levels of success between 68 and 80% correct classifications, combining both techniques when distinguishing between two different categories.

An alternative technique using Word Patterns were tested by Ghanem et al. [9] who managed to achieve results around approximately 80%. The technique centers around identifying strings of the from <noun> is good and use them as training features.

Most of the work that have been done has either used the Reuters Corpus [18] or the Wall Street Journal Corpus [11]. Both of those corpora contains texts that are from the newspaper sphere and typical genres that have been identified are editorials and news. One example that relates to the web sphere is Elgersma et al. [6] that tried to identify whether a web page was a blog or not. With the aid of handpicked features, such as keywords and different statistical measures, they managed to achieve a precision of over 94% with their test setup and a relatively limited training corpus of only 200 texts. This result gives a hint that it is possible to achieve good results without the use of a very large training corpus.

2.3 Question detection

Detecting questions is a field that is part of summarization, especially summarization of e-mail threads, where it is important to find both questions asked and their answers. Shrestha et al. [24] created decision rules with the Ripper algorithm with part-of-speech n-grams as features. A precision of 96% was archived with a recall of 72%. They found that the recall was limited by declarative questions and that research should be put into that area.

2.4 Sentiment analysis

A sub field of text categorization that also borders to semantic analysis is sentiment analysis, to derive whether or not a text is positive or negative. This is usually tackled with a mix of statistical and lexical tools. In contrast with genre categorization, there are a lot of available test and training data from internet web sites that centers around reviewing different things. Pang et al. [17] compared a rule based classifier constructed by a human to a rule based classifier trained by different
machine-learning techniques. The best result for the human classifier where 69% while the best performing machine trained classifier archived 82.9% using unigrams of words as features.

One problem that arises when using unigrams is that some words like *bad* can have very different meanings based on the context, for example 'is bad' and 'is not bad'. Unigrams are unable to cope with these facets because of their simplicity. A solution for this is the use of appraisal groups [27] [26]. Adjectives are grouped together with polarity words as *not* and modifiers as *very* to get a sentiment score. Using the approach of appraisal groups in combination with bag-of-words (essentially unigrams), a score of 90.2% correct classifications was achieved.

### 2.5 Subjectivity analysis

One area that is closely related to sentiment analysis is subjectivity analysis, to detect if a sentence is subjective or objective. Pang et al. [17] found that their sentiment classification scored better on the subjective extracts than on the entire review. A classic machine learning approach to the problem was tried by Yu and Hatzivassiloglou [29] who detected subjective sentences with Naive Bayes classifiers and found that it performed better on finding opinions than on finding facts.

Pang et al. [17] used the fact that subjective and objective sentences are usually grouped together. With the use of a function for comparing the similarity between sentences they transferred the problem to finding the minimum cut in a graph.

Rilof et al. [19] used a high precision classifier to identify subjective sentences. These sentences where then used to create word patterns that where then used to identify more subjective sentences that could be used in training. They managed to obtain high results for both recall and precision with their experiments and concluded that word patterns are linguistically richer than single words and fixed phrases.
Chapter 3

Problems, corpus and performance

This section covers the initial problems that will have to be solved in this thesis such as defining a good set of speech acts to use, and how to evaluate and measure performance. Other prerequisites as obtaining and compiling a corpus will also be discussed.

3.1 Selection of speech acts

One particular problem that has to be solved before starting was to select the speech act that would be considered. As noted by Roussinov et al [20] an assignment of genres or categories will always be fuzzy, and the classifications of a text will not always result in a definite answer that would be universally true.

Finding a good set of speech acts that is complete enough, not overlapping and that make sense in a real world situation for classification of texts in internet discussion groups goes well beyond the scope of this thesis. After an initial study of the material that was to be used, the set of speech acts was selected as Other, Questions and Answers.

3.2 Performance

To construct a good classifier it is important to define what characteristics you want from a classifier. Even though some of the criteria discussed below are not absolute and can not explicitly be measured, they will be considered for all solutions.

3.2.1 Recall and Precision

Two very common concepts that are used to evaluating Natural Language categorization and classification systems are precision and recall [12]. Precision measures how good the given answers were is defined as:

\[
\text{Precision} = \frac{\#\text{CorrectAnswers}}{\#\text{Answers}}
\]
CHAPTER 3. PROBLEMS, CORPUS AND PERFORMANCE

Recall is defined as how many correct answers the system found and is defined as:

\[ \frac{\#CorrectAnswers}{\#TotalpossibleCorrectAnswers} \]

These two measurements are often in opposition to each other, increased recall often leads to decreased precision and increased precision usually leads to decreased recall. One simple example is the system that wants to maximize precision and therefore only answer for the cases where it is very confident, precision will be high but recall will be very low, since a lot of cases will be missed. Therefore precision and recall must be measured together to get an accurate result. In this thesis, precision and recall will be part of the measures used to measure the result of the output.

3.2.2 Domain transferability

Domain transferability is a common performance measure for a classifier. It means that the classifier will work well on several topical domains, examples are football, politics and handicrafts. It is especially interesting if a classifier works well in a domain that it has not been trained on. The reason for this measure is the huge amount of different domains and the impracticality of building a classifier for each domain.

3.2.3 Ease of Training

A golden rule in Machine Learning is that more training data will increase the result of the trained algorithm. Still collecting and annotating training data can be very time consuming and expensive. It has been shown that even with pretty small sets of training data (see [6]), good results can be achieved by clever construction of the classifier. Therefore a good performance measure is that the amount of required training data is small. As the available training data for practical reasons are limited, this is a performance measure that will be important when designing. For example it is unwise to select very rare features for training. If a feature can be found in only one of 10,000 documents, there would probably be needed at least 50,000 documents in the training set for the training to be useful.

3.2.4 Amount of Text Needed for Classification

An ideal classifier would be able to accurately classify the speech act of a text just from a single word or sentence. In reality such precision is often not possible, even for humans. It is also a well known fact that it is easier to classify a large text than a small one by statistical measure, since random noise (examples misspelled words and grammatical errors) have a higher impact on small texts. A good classifier should be able to accurately classify small texts.

The importance of this measure are very dependent on the task in hand. For example when classifying entire books, the amount of text for each item to classify
3.3. CORPUS

is huge so the amount of text needed for classification is not a problem. But when categorizing small text messages, it is one of the major problems faced.

3.2.5 Robustness

It is desirable for any good computer system to be tolerant of input that is not correctly formatted, misspelled, damaged or in other ways faulty. We call this feature robustness.

For a classifier of human language, robustness can be translated to managing misspelled words, grammatically bad formed sentences, unknown words and bad use of non alphabetical characters such as; @,$ or <. As the focus on this thesis is on texts from internet groups where language in general are not spell-checked or grammatically checked and new words or pseudo-words have a tendency of being quite common, robustness will be a big issue to keep in mind. Bigert et al.[10] found that relatively few errors(10%) could decrease the accuracy of a parser quite a lot.

3.2.6 Computation resources

To be able to use an algorithm or method in real life outside strictly controlled and limited research domains it is important that it does not require to much computing resources, such as memory and cpu-time. Of course to much resources is hard to clearly define and also varies according to several different parameters such as results, hardware and application. For this thesis, performance will not be explicitly evaluated and measured. It will be taken into account so that methods that are too computationally complex to use will not be considered.

3.3 Corpus

When doing statistical Natural Language Processing one of the main components is the set of annotated training data, usually referred to as the corpus. The difficulties in creating a corpus consist of finding a set of suitable texts and annotating the text with good precision. To be effective it is important that the corpus is both big enough and accurate enough to train your machine learning algorithm. To help solve this challenge, universities and other research centers have been collecting large corpora that are available for research. Unfortunately there does not exist any such large corpus covering the area of this thesis.

Therefore a corpus had to be created. The source used were discussion groups on Yahoo!(for the specific group see Appendix 2.) using the following method:

1. Select 6 different groups, with different domains.
2. The 20 most recent discussion threads are selected.
3. Threads containing only one post is ignored and not included in the above 20.
4. Threads that are spam are ignored. This was determined by the person collecting the data, on the criteria that the thread had just one post and that the topic was distant from that of the group.

5. The text from all posts in the message are selected. Headlines and copies of earlier posts are removed as well as signatures and marketing for the service provider.

6. Different posts are separated by one blank line.

An example of a cleaned discussion can be seen in Figure 3.1.

As an additional boost, parts of other corpora were used to improve the training where it could be applicable. For boosting when training on questions, AnswerBus question corpus was used[2].

### 3.4 Comparision of results to others

It is important to be able to compare results of this thesis with what others have done and let other compare their results to those found in this thesis. A regular
3.4. COMPARISON OF RESULTS TO OTHERS

approach for research in Natural Language Processing is to use a publicly available corpus like the British National Corpus (BNC)\[5\] to test on and then compare the results obtained.

Since there are no publicly available corpus that satisfies the need for this thesis, this standard approach will not work. Instead a comparison based approach will be used. A simple base line classifier will be implemented, so that others can compare the results with the base line classifier and at least get a relative comparison to the results of other more complex classifiers.
Chapter 4

Algorithms and Design

In this section it will be described how the classification works in all steps for the different techniques that will be used. The first part will be a description of the structure of a complete classifier and the later parts will look more into detail of the different components.

4.1 General setup of classifier

To simplify testing of different algorithms and methods of machine learning and feature selection, the design of the classifier has been modularized in to different parts, as shown in figure 4.1. Additionally the entire classifier has been divided into two different modules, training and analysis. One should note that several components appear in both modules. Also it is important to note that with this design, the analysis module can not run without the output from one run of the training module.

4.1.1 Training module

The training module takes a set of text pieces annotated with speech acts has input. As output it produces a classifier, consisting of a set of features and a machine learning algorithm using those features, trained on the set of annotated text pieces.

The module works in two steps(a)analyzing the annotated texts to get a set of features and (2) training a machine learning algorithm.

4.1.2 Analysis module

The analysis module has as requirement, a trained machine learning algorithm and a set of selected features. As input it takes a text. First the chunker splits the text into several smaller parts. It is important that the smaller parts are independent of each other in the sense that a speech act is not divided in two or more parts. These parts are then individually classified by the classifier using the feature set and the
machine learning algorithm. As output there is either the most probable category for each chunk or the probability for belonging to each category for every chunk.

Figure 4.1. General setup of a classifier

### 4.2 Feature Selection

The need for a feature selection arises from two needs: (1) the level of the information that is considered and (2) the size of the corpus.

Depending on what you classify, different levels of abstraction in the text needs to be considered. Looking for specific keywords might be a very good idea if the job is classifying news articles such as sports or economy. Looking for quotations marks would be good if you were trying to find questions, but would not work quite as
well if the problem was categorizing a text as different languages. Because of this, the features used will have to be carefully selected so that they work for the right level of abstraction.

The other reason is the size of the corpus. If we define a marker for a category as a feature that is more probable to appear in texts with that category than in texts with other categories. A marker can be considered strong if it is far more likely to appear in one category than in others. One good example is the question mark, it is far more probable to appear in a text that is categorized as question than in a text categorized as answer. If our corpus was infinitely large, all features that where markers for a category, regardless of strength, would exist more often on that category and less on other categories. Features that weren’t marker for any category would be evenly distributed over all categories.

But since we have a limited corpus, probability is bound to distribute some features that are not markers for a category unevenly so that they appear to be markers. The term for this is over-training. Although this phenomena will always occur, we can limit it by placing a cap on how many times a feature must exist in our corpus to be considered a marker. A feature that only appeared once would by our definition be considered a strong marker, but it might as well be a feature that isn’t a marker but only a rare feature.

This will of course have the side effect that some rare features that really are markers for a category will be lost. Finding a good balance between over-training and loss of features one are important way to optimize a classifier.

To summarize, feature selection can be split up in to parts: (1) the actual features being selected and (2) sorting away features that are not markers or are to rare in our corpus.

4.3 Different feature sets

There exists a vast multitude of different features that can be considered when analyzing texts. This section will give an overview of the ones used in this thesis.

4.3.1 Words

Trying to find words that are typical for a certain category, is a commonly used method. Usually the most common words like the, he, is are ignored, since they exist in all texts regardless of category. The most common way is by simply using a stop list for words to ignore that contains the most common words. Words as a feature generally works best for determining the topic of a text and not for deciding the speech act of a piece of text.

4.3.2 N-grams

N-grams are not really a feature set, but rather a way to extend different feature sets. A N-gram of words is just n words combined together, for example the text
bacon cheese sandwich would result in the 2-gram of words, bacon cheese and cheese sandwich. N-grams is typically used to extract a deeper semantical meaning from a text than just single words. One problem with n-grams is that there will be fewer observations of each n-gram in a given corpus for bigger n’s since there are more possible n-grams. The solution for this problem is using a larger corpora which sometimes can be troublesome to obtain.

### 4.3.3 Word classes

Using the word class of the words can be a good feature for determining the overall structure and type of individual sentences. Since there are few word classes compared to words, the use of N-grams of word classes often works well even if the corpus is not very large.

### 4.3.4 Common Words

One different way of using the words in a piece of text to find the speech act is using the frequencies for the most common words. The assumption is that the frequencies will be the same for a text of the same speech act regardless of the topics and contextual content. So if a text is question, the frequencies of the most common words will be the same if the text concerns handicraft, cars or babies, since the most common words are used regardless of topic.

### 4.3.5 Word Patterns

Word patterns is a mix between, words, word classes and N-grams. A word pattern is built using a N-gram consisting both of words, and word classes. An example could be *I like adjective noun*. Word patterns are interesting because they can carry more information about the actual structure of the sentence than a N-gram of word classes without being as closely tied to the topic of the text as regular N-grams of words.

### 4.3.6 Lexical Markers

The use of lexical markers stem from how people learn languages. According to Nattinger and DeCarrico [15], when people learn a language, they begin with small lexical phrases that they use as building components. One example is *I want to have ...* describing want for something. The theory is that the use of these markers remain even when someone get more proficient in the language. These markers, when identified, can then be used to identify what goal the author of a text has, and could be useful for finding questions and answers.
4.4. MACHINE LEARNING ALGORITHMS

4.3.7 Statistical features

One different set of features that can be used is statistical features. These are numbers computed from the text, without looking at the individual words of the text but rather by looking at the entire text. Examples of this are text length, average word length or words per sentence. Statistical features usefulness is that they can tell us a lot about the speech act of the text while ignoring the actual content of a text.

One example of the usefulness of these kind of features is computing how hard a text is to read, this is often done by computing the average sentence length and the ratio of words with many characters in them. One example is the Swedish LIX-index (Readability index).

The key challenge with statistical features is that finding suitable statistical measurements can be hard to automate, so a lot of manual labor is needed. Another disadvantage is that transferability between different languages might make a set of features virtually useless and more manual work might have to be applied.

4.4 Machine Learning Algorithms

There exists a multitude of different machine learning algorithms. Below, the main type of algorithms will be described in general. For a more detailed description, Fabrizio Sebastiani’s *Machine learning in automated text categorization* [23] is a good place to start. The different classifiers used in this thesis were chosen on basis of being proven to be effective in machine learning tasks, as well as being available in third party software packages.

There exists a lot of different implementations of the general ideas with minor optimizations and tweaks to get better performance.

4.4.1 Decision Trees

A decision tree tries to build a tree where in each node there is a decision to choose what branch to follow. A classification starts in the root of the tree and follows the branches, using the condition in each node to determine what branch to follow until you get to a leaf. The leaf then determines the category.

The main work in constructing this algorithm is finding good conditions to use in the nodes. A condition is good if it there is one category where most of the text get the same result from the condition. A perfect condition would pass all texts of category down one branch in the tree and all others to other branches. In Fig 4.2 there is an example of a decision tree. The condition in the root node is if there are less than 2 sentences follow the first branch. The next condition is if there are no question marks, if thats true follow the first branch. By continuing this way, a leaf will be reached, each leaf is marked by a classification and a score of number of correct classifications versus the number of in correct classification.
A special type of decision tree classifier is the rule based classifier. A rule classifier consists of ordered set of rules that determines what rule you should go to next, until you have a classification. There is no distinct border between decision tree classifiers and rule based classifiers and they perform more or less the same. The main advantage that have made rule based classifiers preferred over decision tree classifiers, are that they are more easy for humans to interpret. In Figure 4.4.1 there is an example of a rule based classifier. It shows two commons rules, the first one can be described as: more then zero question marks and no exclamation marks and not more than one sentence is a question, this will give 33 correct and 3 incorrect classifications of speech act.

A good example is the decision tree(see 4.2) and a corresponding rule based classifier(see 4.4.1), the later is much more simple to understand and describe in human language.

Number of ? > 0 AND
Number of ! <= 0 AND
Number of Sentences <= 1: question (33.0/3.0)

Number of ? > 0 AND
Count of think,best,opinion <= 1 AND
Number of ! <= 0 AND
Number of I,my <= 1 AND
Ratio of nominal pronoun > 0.02381: question (16.0/1.0)

Figure 4.3. Example of a Rule based classifier
4.4. MACHINE LEARNING ALGORITHMS

4.4.2 Support Vector Machine

Support Vector Machines (SVM) is an entirely different approach to machine learning than decision trees. When creating a SVM for two categories you have to calculate a value for the selected features \( F_1, F_2, \ldots, F_n \), to create a n-dimensional feature vector for all different texts. All vectors are then mapped in a n-dimensional space. The goal is then to find a mapping of the feature vectors to a 2 dimensional space. In a perfect mapping all feature vectors for the first category will have a non zero value on the first dimension and a 0 value on the second dimension, and vice versa for the other category.

4.4.3 Bayesian networks

One commonly known technique for machine learning is Bayesian networks. These are based on the Bayes theorem

\[
P(A|B) = \frac{P(B|A)P(A)}{P(B)}
\]

The probability of A given B, is the same as the probability probability of B given A times the probability of A divided by the probability of B. An simple example: We want to compute the probability of seeing rain in the afternoon(A) given we have clouds in the morning(B). According to the theorem the probability is, the probability of clouds in the morning on days it rains in the afternoon times the probability of rainy afternoons divided by the probability of cloudy mornings.

This theorem is used to construct a network of evidence variables \( F_1, F_2, \ldots \) and a variable C that we want to find out. The network is created so that evidence variables that are dependent on each other are connected.

One problem with Bayes networks are that they can be hard to construct, needs a lot of training data and are often more complicated than really needed. A simplified version of the Bayes Network is the Naïve Bayes (NB). When constructing a NB, the assumption is made that all evidence variables are conditionally independent of each other. This makes it easier to construct the network and the computation and training data needed is considerably smaller. Because of their ease to use, NB are quite common in Natural Language Processing, and despite their simplified assumption often, almost as good as a Bayes Network.

4.4.4 Committee classifiers

Different classifiers excel on classifying different types of text and different types of categories. The idea with a committee classifier is to use several different classifiers that have been trained for the same task, using different training sets and/or techniques and combine them to improve the overall result of the classification[23].

In the most straightforward implementation of a committee classifier, all classifiers are used to classify the text and then the most common classification is the one used. In more sophisticated implementations, the different classifiers are given
different weights that are optimized to maximize performance over a training set of texts.

4.5 Discourse classifiers

The previous discussed classifiers have only looked at the single piece of text they where given to classify. Information on what came before the piece is not considered. In the real world we know that a question is more likely to be followed by an answer than by a question. We call classifiers that use information about other related pieces of text to classify a text piece as Discourse Classifiers.

The Discourse Classifiers require a somewhat changed structure in the model of how the classifiers work. First of all, as training input it is not enough that random annotated text pieces are sent in. Instead entire chains of annotated text pieces needs to be sent in. From these chains a regular classifier, that classify a chunk of text without information about other related chunks. The information about from the annotated chains are used to compute probabilities for a sequence of chunks leading to a chunk of a specific type.

One other key difference is that the Discourse Classifier themself are not enough to work as a classifier, an implementation of a regular classifier is also needed. The extended model for Discourse Classifiers (A Hidden Markov Model Classifier, see figure 4.5) shows how the regular classifier is used. It is good to note that all Discourse Classifiers do not use all of the steps displayed in the model.

4.5.1 Markov Model

One way to analyze and model sequences of events are the Markov Model, A Markov Model is based on the Markov assumption, stating that the value of any state is only influenced by a limited sequence of previous states. This assumption actually fits classification of Internet discussion threads very well as an author can only base a response on what are already written in the thread.

A Markov Model consists of only three things:

- A set of states $\Sigma$
- Probabilities of starting on each different state. $S_\alpha$, $\alpha \in \Sigma$
- Probabilities $P_{\alpha|\beta_1\beta_2..\beta_n}$ of the next state $\alpha$ given a string of previous states $\beta_1\beta_2..\beta_n$ where $\alpha, \beta_1, \beta_2, ..., \beta_n \in \Sigma$.

A Markov Model is said to have an order, meaning that the length of the sequence of previous states that are considered. As with N-grams, a higher order means that more training data will be needed to avoid over-learning.

The Markov Model itself is not sufficient for classification of a text, but needs to have the aid of a regular classifier. For a first order classifier the calculation of
Figure 4.4. Model of the different steps for the design of a Hidden Markov Model Classifier.
classification of state $\alpha$ with the last classified step $\beta$ and $Q_x$ probabilities from the regular classifier of $\alpha$ being classified $x$, and a Markov classifier $(\Sigma, S, P)$.

$$P_{\alpha\beta} \cdot Q_\alpha, \alpha \in \Sigma$$

Thus the assignment of $\alpha$ that maximizes the expression will be the classification.

### 4.5.2 Hidden Markov Model

The Markov Model has a limitation that it can not handle uncertainty and errors in previous states. An extension of the Markov Model called Hidden Markov Model (HMM) are therefore often used. A HMM differs from a standard MM in that the states are not directly visible for an observer. This is done by extending the Markov Model with $E_{\alpha\beta}$ being the probability of emitting $\alpha$ in state $\beta$. Therefore, from an observed trail of emissions, the internal states must be computed. The result of such an computation will be the most likely internal states. The use of the emission probabilities make it possible to model imperfections of the classifiers and makes it possible to achieve better results.

### 4.6 Chunking and subdividing of text

A written text often consists of several parts or sections formed with different intent. It is not uncommon for a text to begin with an answer to a question and to end with a follow up question. In real-life situations these different parts of the text is not marked. This situation raises the need for subdividing a text into smaller chunks that only contains one speech act to accurately determine what the text should be classified as. The goal is to separate text formed with different intents in to to parts that are as lexically independent from each other as possible.
Chapter 5

Implementation

This chapter describes the implementation of the classifiers that are used in this thesis. Software packages that are used are also briefly described. For more complete information follow the references to individual websites.

The overall structure of the classifier (see 4.1) consists of two main components, extraction of features and a machine learning algorithm. Due to the modular design these components can be changed or combined at will. As the machine learning algorithms are all from the Weka Software package[16], no detailed description will be made of each classifier implementation.

5.1 Software package

5.2 Java 1.5 from SUN

As main programming language for the implementation, Java 1.5 from SUN was selected. The reason was mainly personal preferences and the good availability of useful third party software packages. Since the importance of optimal performance is not a main issue for this thesis, most programming languages could have been used.

5.2.1 Weka

Weka[16] is an open source software package developed at the University of Waikato. The intended use is data mining, and machine learning. Weka incorporates a large number of implementations of different machine learning algorithms (see section 4.4), as well as tools for pre-processing data.

In this thesis Weka was used because it offered a simple and convenient way of testing a multitude of different machine learning algorithms and to compare them in order to see which algorithm or implementation of algorithm that produced the best results.
5.2.2 Lingpipe

LingPipe[1] is a commercial software package for use in the area of Natural Language Processing and data mining. It contains several tools for these areas such as entity-tracking, document clustering after topic part-of-speech tagging and a lot more.

Although LingPipe offers a lot of tools that can be used, only the part of speech tagger was used. The part of speech tagger is an implementation of the BrownTagger[1].

5.3 Chunking of text

There exists a lot of different trivial and non trivial ways to chunk up texts in smaller lexically independent parts. Since the main scope of this thesis is not how to subdivide text in lexically independent chunks, just one simple heuristic for chunking is used.

The selected heuristic chunks up the text after the text's own paragraphs. This heuristic was selected because it is straightforward to implement and because the function of paragraphs. Paragraphs are in general used to separate parts of text that have different intent or lexical context into smaller parts, to improve readability and to mark different parts of the text as separate.

5.4 Feature Extractions

For most of the different algorithms there exist one or more parameters that can be optimized to achieve maximum performance. These will only be referred to as letters, example \( n, m, \ldots \).

5.4.1 Bag of Words

The Bag of Words classifier is a simple classifier where the features are counts of the words in the text, along with a list of stop words to be ignored (usually the most common words). The major advantage with this classifier is the straightforward implementation. The implementation follows these steps:

1. All words are counted
2. The words are sorted, in order after number of occurrences
3. The 50 most common words (as found in BNC) in English text are removed from the list,
4. The \( m \) most common words are selected, to be used for classification
5.4. Feature Extrainations

5.4.2 Word Class N-grams

The entire Word class N-grams algorithm are based around the part of speech tagger from LingPipe. The tagger is trained on the BNC (British National Corpus) and uses word recognition and the context of words close to the word being tagged to determine word class. This leads to problems with malformed texts, with errors and words unknown to the tagger.

As the tagger is not always able to tag all words, it marks them as not tagged. This implementations choose not to include any N-grams that contains words that could not be tagged. With a better and more robust tagger it would be possible that higher levels of accuracy could be achieved.

1. Part of speech tag all words in the text
2. Count all N-grams of length $n$
3. Remove all N-gram that contains words, the tagger was unable to tag
4. Select the $m$ most common N-grams

5.4.3 Word Pattern

The word pattern algorithm can in most parts be seen as a mix between word class N-grams and regular word N-grams, so the same limitations as discussed with the word class N-gram implementation applies as well. The size of the N-grams are typically 3-5, depending on task and amount of training data. The implementation uses the following steps:

1. Part of speech tag all words in the text
2. For all N-grams up to size $n$, count all word patterns
3. Remove word patterns where there was classification errors in word class tags (the word class tagger failed to classify).
4. Select the $m$ most common word patterns

Examples of found word patterns:

I like <adjective> <noun>
<noun> are very <adjective>
He <verb> a lot

In a text they would match these sequences of words.

I like pale ale
Cars are very boring
He sleeps a lot
If the corpus used for training is not very limited, the counting of all possible word patterns will require very large amounts of memory. To counter this only certain word classes were allowed be use as Part of Speech tags. The selected word classes where the most frequent, verb, adverb, adjective and noun. These classes were chosen because they are common in text and therefore required a smaller training corpus when training.

5.4.4 Common Words
The principle for this classifier is to use the occurrence of the most common words to determine speech act(see4.3.4). The common word classifier offers a quite straightforward implementations. The main task is actually the creation of a list of the most common words used in the language. The list used for this thesis was created from British National Corpus (BNC) that contains 100-million words from different domains. For a common word classifier, no real feature selection needs to be performed. The only thing that needs to be decided is how many of the most common words that should be used for the classification.

In addition to common words, punctuations were also used in the implementation in the same way as words.

5.4.5 Statistical features
A classifier that uses statistical features do not perform any real feature selection when it runs. Instead the feature selection is done during the implementation and design time, when the developer decides what statistical measurements to include in the implementation. The features selected where:

- Text length
- Number of sentences
- Average number of words per sentence
- Number of ! and ?
- Ration of different word classes
- Count of opinion markers
- Count of Topic markers
- Count of Evaluators
- Count of Summarizers
- Count of Opinion markers

As in other implementations where world class tagging is used, there can be error sources from errors in the implementation in the word class tagger as well as error in the training data.
5.5. OTHER IMPLEMENTATION ISSUES

5.5 Other implementation issues

5.5.1 Committee classifier implementation

The implemented committee classifier uses weighting of the different classifiers used to symbolize that different classifiers performances vary. The optimization is done with a probabilistic algorithm, the Hillclimbing algorithm[21]. The steps of the algorithm is as follows:

1. Give all classifiers weight of 1
2. Measure the result of the classifier
3. Randomly decrease the weight of on classifier with some small value $n$ and add the weight to another classifier
4. Measure the result with the new weights
5. If the new weight distribution do not decrease the result, keep it.
6. Go to 2 unless this is the $m$ iteration

Since this is a probabilistic algorithm it is not guaranteed to find a maximum, even if $m$ is a very large number. Therefore the Hillclimbing algorithm must run several times to find an optimal weight distribution.

It should be noted that if the classifiers are slow when classifying or many training texts or classifiers are used, this process will become very slow and become impractical. To decrease running time the values for $m$ and $n$ can be decreased, but this would be likely to decrease performance of the optimization and hence the performance of the classifier.

5.5.2 Parameter optimization

On many of the described implementations, there are variables, denoted as $m$, $n$... These values are points of optimization that will greatly affect the performance for each classifier. By tuning these parameters, the result can be tuned to get better performance both overall and on different speech acts or categories.

For parameter optimization, local search is used. The algorithm is basically the same as the one used for weight optimization in the committee classifier but with the basic difference that when a value $a$ of parameter is randomly selected (step 3) it can either be increased or decreased and no other parameters are affected since they are all independent. There is no constraint of the total sum of the parameters as in the weight optimization problem.
5.5.3 Optimizing the combination of feature selection and Machine Learning Algorithms

As a further point of optimization are matching the feature selection algorithm to the optimal machine learning algorithm. As Sebastini\[23\] noted, that most types of machine learning algorithms have roughly the same performance, this is not an area where a lot can be gained. Still some classifier are better then others given the task at hand and the amount of training data\(\text{(about 5-10\% differens was noted)}\). By trying out a feature selection algorithm with different machine learning algorithms, a small increase in performance can be made.

For this thesis a simple program that just tested different implementations of classifiers \(\text{(supplied from the WEKA package)}\) for a specific feature selection algorithm was used.
Chapter 6

Results

In this part different implementations are evaluated against each other for a number of different tests. In the tests the ration between training set and evaluation set is kept to 80:20. The total size of the combined evaluation and training set was about 3000 items, from about 160 discussion threads (for more details see Appendix 3). For the discourse classifiers the discussion threads had to stay in one part.

This chapter contains a lot of tables and results. All evaluation is done in the next chapter.

6.1 The classifiers

To have a benchmark and a point of reference a naive classifier will be included in the tests. The naive classifier just guesses what speech act a chunk belongs to without looking at it. Hence it will always make 33% correct classifications.

These are the classifiers being used for all the tests:

Naive classifier  Guesses the speech act.

Bag-of-Words SVM classifier  This is a baseline classifier, because of it easy implementation, included for others to use as comparison. It uses regular words as features and a Support Vector Machine as ML algorithm.

Word class n-gram classifier  As features set, N-grams of word classes are used.

Word Pattern classifier  Uses word patterns as feature sets.

Common words classifier  Uses frequencies of the 100 most common words as feature set.

Statistical classifier  Uses a set of selected statistical values as feature set.

Committee classifier  Uses the Word Class N-gram, word pattern, common words and statistical classifiers as a committee with different weights.
CHAPTER 6. RESULTS

**Markov classifier** Builds a Markov Model of the discussions. Uses the common word classifier as classifier.

**HMM classifier** Builds a Hidden Markov Model of the discussions. Uses the common word classifier as classifier.

For more details of parameters and setup of the classifiers see Appendix 1.

### 6.2 Results in a single domain

One important measure of the classifiers are how good they perform over a single domain. It is also important to note if there are differences over how the classifier performs over several domains.

For this test the classifiers where tested and trained on just one domain. The results can be seen in table 6.2.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>D 1</th>
<th>D 2</th>
<th>D 3</th>
<th>D 4</th>
<th>D 5</th>
<th>D 6</th>
<th>D 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>32,9</td>
<td>42,4</td>
<td>41</td>
<td>39,1</td>
<td>43,1</td>
<td>66,7</td>
<td>38,9</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>32,9</td>
<td>24,2</td>
<td>41</td>
<td>32,6</td>
<td>41,7</td>
<td>66,7</td>
<td>38,9</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>32,9</td>
<td>69,7</td>
<td>23,1</td>
<td>30,4</td>
<td>44,4</td>
<td>66,7</td>
<td>38,9</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>44,3</td>
<td>33,3</td>
<td>53,8</td>
<td>34,8</td>
<td>44,4</td>
<td>61,9</td>
<td>48,1</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>32,9</td>
<td>45,5</td>
<td>51,3</td>
<td>60,9</td>
<td>47,2</td>
<td>61,9</td>
<td>38,9</td>
</tr>
<tr>
<td>Ensemble classifier</td>
<td>38</td>
<td>39,4</td>
<td>41</td>
<td>32,6</td>
<td>44,4</td>
<td>61,9</td>
<td>38,9</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>43</td>
<td>30,3</td>
<td>43,6</td>
<td>32,6</td>
<td>45,8</td>
<td>66,7</td>
<td>38,9</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>44,3</td>
<td>33,3</td>
<td>51,8</td>
<td>34,8</td>
<td>44,4</td>
<td>61,9</td>
<td>48,1</td>
</tr>
</tbody>
</table>

**Table 6.1.** The percentage of correct classifications over the seven different domains used.

All classifiers varied over the different domains (see table 6.2 for the domains). To illustrate the difference a table with minimum, maximum and mean percent of correct classifications has been put together (see table 6.3).

<table>
<thead>
<tr>
<th>Aberration</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Musical</td>
</tr>
<tr>
<td>D2</td>
<td>Soup making</td>
</tr>
<tr>
<td>D3</td>
<td>Sailing</td>
</tr>
<tr>
<td>D4</td>
<td>Pottery</td>
</tr>
<tr>
<td>D5</td>
<td>Computers</td>
</tr>
<tr>
<td>D6</td>
<td>Cooking</td>
</tr>
<tr>
<td>D7</td>
<td>Toys</td>
</tr>
</tbody>
</table>

**Table 6.2.** The content of the different domains. For exact content of the domains see Appendix 2.
6.2. RESULTS IN A SINGLE DOMAIN

More detailed results from the test can be seen in tables 6.4 and 6.5.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>min</th>
<th>max</th>
<th>mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>32,9</td>
<td>66,7</td>
<td>38</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>24,2</td>
<td>66,7</td>
<td>34,8</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>23,1</td>
<td>69,7</td>
<td>38,3</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>33,3</td>
<td>61,9</td>
<td>40,1</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>32,9</td>
<td>61,9</td>
<td>42,3</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>32,6</td>
<td>61,9</td>
<td>37</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>30,3</td>
<td>66,7</td>
<td>37,6</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>33,3</td>
<td>61,9</td>
<td>39,8</td>
</tr>
</tbody>
</table>

Table 6.3. The maximum, minimum, mean percentage of correct classifications from table 6.1.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Question</th>
<th>Answer</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>22</td>
<td>41</td>
<td>47</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>23</td>
<td>43</td>
<td>42</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>27</td>
<td>51</td>
<td>43</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>73</td>
<td>56</td>
<td>55</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>27</td>
<td>45</td>
<td>51</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>48</td>
<td>52</td>
<td>51</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>45</td>
<td>56</td>
<td>54</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>73</td>
<td>56</td>
<td>55</td>
</tr>
</tbody>
</table>

Table 6.4. Percentage of precision over each category for the classifiers.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Question</th>
<th>Answer</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>10</td>
<td>42</td>
<td>62</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>24</td>
<td>39</td>
<td>45</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>35</td>
<td>17</td>
<td>68</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>29</td>
<td>53</td>
<td>74</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>14</td>
<td>57</td>
<td>51</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>20</td>
<td>52</td>
<td>62</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>51</td>
<td>53</td>
<td>54</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>28</td>
<td>53</td>
<td>74</td>
</tr>
</tbody>
</table>

Table 6.5. Percentage of recall over each category for the classifiers.
6.3 Results over multiple domains

Since it is impractical to train different classifiers for all possible domains, it is important to see how well the classifiers behave when they are trained for use with several domains. In this section the classifiers have been trained on a different number of domains and then evaluated for the same domains. A summary of the results can be seen in table 6.6.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>43</td>
<td>45</td>
<td>47</td>
<td>50</td>
<td>49</td>
<td>48</td>
<td>45</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>38</td>
<td>47</td>
<td>43</td>
<td>44</td>
<td>43</td>
<td>43</td>
<td>46</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>40</td>
<td>45</td>
<td>42</td>
<td>45</td>
<td>49</td>
<td>44</td>
<td>47</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>58</td>
<td>46</td>
<td>45</td>
<td>48</td>
<td>51</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>68</td>
<td>45</td>
<td>47</td>
<td>50</td>
<td>53</td>
<td>52</td>
<td>40</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>51</td>
<td>48</td>
<td>47</td>
<td>49</td>
<td>53</td>
<td>50</td>
<td>48</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>71</td>
<td>48</td>
<td>45</td>
<td>48</td>
<td>53</td>
<td>50</td>
<td>46</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>58</td>
<td>46</td>
<td>45</td>
<td>50</td>
<td>51</td>
<td>51</td>
<td>51</td>
</tr>
</tbody>
</table>

Table 6.6. Percentage of correct classifications for each classifier over different numbers of domains

More detailed results for each category can be found in the following tables; 6.7, 6.8, 6.9, 6.10, 6.11, 6.12.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>22</td>
<td>22</td>
<td>0</td>
<td>50</td>
<td>25</td>
<td>0</td>
<td>50</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>24</td>
<td>14</td>
<td>21</td>
<td>23</td>
<td>26</td>
<td>22</td>
<td>31</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>28</td>
<td>18</td>
<td>30</td>
<td>24</td>
<td>34</td>
<td>35</td>
<td>37</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>73</td>
<td>31</td>
<td>37</td>
<td>34</td>
<td>39</td>
<td>55</td>
<td>48</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>27</td>
<td>22</td>
<td>54</td>
<td>36</td>
<td>49</td>
<td>58</td>
<td>38</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>48</td>
<td>26</td>
<td>55</td>
<td>36</td>
<td>49</td>
<td>58</td>
<td>69</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>45</td>
<td>36</td>
<td>34</td>
<td>36</td>
<td>43</td>
<td>57</td>
<td>47</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>73</td>
<td>31</td>
<td>37</td>
<td>28</td>
<td>39</td>
<td>55</td>
<td>48</td>
</tr>
</tbody>
</table>

Table 6.7. Percentage of precision for category Question over different number of domains

6.4 Results in unknown domains

An important measure is to see how well a classifier does when it is used for a domain it has not been trained on. In real life situation this can be important since it is very hard to cover all possible domains when gathering training data. It is also interesting to see how much training for several different domains affects the result of classifying an unknown domain. The classifiers are trained for one or
### 6.4. RESULTS IN UNKNOWN DOMAINS

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>10</td>
<td>17</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>24</td>
<td>6</td>
<td>10</td>
<td>18</td>
<td>19</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>35</td>
<td>8</td>
<td>13</td>
<td>14</td>
<td>19</td>
<td>15</td>
<td>24</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>39</td>
<td>43</td>
<td>48</td>
<td>42</td>
<td>47</td>
<td>45</td>
<td>67</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>14</td>
<td>17</td>
<td>43</td>
<td>38</td>
<td>42</td>
<td>41</td>
<td>43</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>30</td>
<td>16</td>
<td>34</td>
<td>35</td>
<td>29</td>
<td>28</td>
<td>24</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>51</td>
<td>54</td>
<td>41</td>
<td>34</td>
<td>51</td>
<td>38</td>
<td>54</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>39</td>
<td>43</td>
<td>48</td>
<td>46</td>
<td>47</td>
<td>45</td>
<td>67</td>
</tr>
</tbody>
</table>

**Table 6.8.** Percentage of recall for category Question over different number of domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>42</td>
<td>45</td>
<td>43</td>
<td>52</td>
<td>47</td>
<td>50</td>
<td>44</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>43</td>
<td>47</td>
<td>40</td>
<td>48</td>
<td>43</td>
<td>50</td>
<td>47</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>52</td>
<td>44</td>
<td>39</td>
<td>54</td>
<td>52</td>
<td>55</td>
<td>51</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>57</td>
<td>46</td>
<td>41</td>
<td>51</td>
<td>49</td>
<td>52</td>
<td>46</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>45</td>
<td>45</td>
<td>41</td>
<td>53</td>
<td>50</td>
<td>51</td>
<td>38</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>52</td>
<td>47</td>
<td>41</td>
<td>53</td>
<td>50</td>
<td>51</td>
<td>44</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>57</td>
<td>49</td>
<td>41</td>
<td>50</td>
<td>53</td>
<td>53</td>
<td>44</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>57</td>
<td>45</td>
<td>41</td>
<td>54</td>
<td>49</td>
<td>52</td>
<td>46</td>
</tr>
</tbody>
</table>

**Table 6.9.** Percentage of precision for category Answer over different number of domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>42</td>
<td>45</td>
<td>63</td>
<td>48</td>
<td>64</td>
<td>53</td>
<td>55</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>39</td>
<td>43</td>
<td>44</td>
<td>34</td>
<td>57</td>
<td>35</td>
<td>40</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>17</td>
<td>20</td>
<td>37</td>
<td>28</td>
<td>35</td>
<td>31</td>
<td>31</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>88</td>
<td>42</td>
<td>44</td>
<td>44</td>
<td>50</td>
<td>49</td>
<td>37</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>58</td>
<td>44</td>
<td>52</td>
<td>52</td>
<td>54</td>
<td>59</td>
<td>37</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>52</td>
<td>50</td>
<td>44</td>
<td>43</td>
<td>56</td>
<td>50</td>
<td>49</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>53</td>
<td>50</td>
<td>44</td>
<td>54</td>
<td>56</td>
<td>55</td>
<td>48</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>53</td>
<td>43</td>
<td>44</td>
<td>48</td>
<td>50</td>
<td>49</td>
<td>37</td>
</tr>
</tbody>
</table>

**Table 6.10.** Percentage of recall for category Answer over different number of domains

---

35
more domains and then evaluated for other domains to see how well they handle these. There is a summary of the results in table 6.13 and more detailed results with precision and recall in tables 6.14, 6.15, 6.16, 6.17, 6.18 and 6.19.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>47</td>
<td>50</td>
<td>52</td>
<td>49</td>
<td>53</td>
<td>46</td>
<td>46</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>42</td>
<td>50</td>
<td>49</td>
<td>46</td>
<td>49</td>
<td>44</td>
<td>48</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>43</td>
<td>47</td>
<td>45</td>
<td>45</td>
<td>51</td>
<td>41</td>
<td>48</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>55</td>
<td>53</td>
<td>54</td>
<td>51</td>
<td>59</td>
<td>49</td>
<td>57</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>52</td>
<td>50</td>
<td>52</td>
<td>52</td>
<td>57</td>
<td>52</td>
<td>45</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>51</td>
<td>55</td>
<td>51</td>
<td>49</td>
<td>57</td>
<td>47</td>
<td>49</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>54</td>
<td>45</td>
<td>53</td>
<td>49</td>
<td>59</td>
<td>45</td>
<td>47</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>55</td>
<td>50</td>
<td>54</td>
<td>51</td>
<td>59</td>
<td>49</td>
<td>57</td>
</tr>
</tbody>
</table>

Table 6.11. Percentage of precision for category Other over different number of domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>62</td>
<td>54</td>
<td>49</td>
<td>68</td>
<td>52</td>
<td>65</td>
<td>56</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>45</td>
<td>63</td>
<td>53</td>
<td>64</td>
<td>38</td>
<td>68</td>
<td>66</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>68</td>
<td>79</td>
<td>56</td>
<td>75</td>
<td>76</td>
<td>74</td>
<td>75</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>74</td>
<td>50</td>
<td>46</td>
<td>55</td>
<td>54</td>
<td>56</td>
<td>56</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>51</td>
<td>54</td>
<td>44</td>
<td>52</td>
<td>55</td>
<td>49</td>
<td>43</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>62</td>
<td>55</td>
<td>55</td>
<td>64</td>
<td>58</td>
<td>60</td>
<td>58</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>54</td>
<td>45</td>
<td>46</td>
<td>46</td>
<td>50</td>
<td>50</td>
<td>40</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>74</td>
<td>50</td>
<td>46</td>
<td>53</td>
<td>54</td>
<td>56</td>
<td>56</td>
</tr>
</tbody>
</table>

Table 6.12. Percentage of recall for category Other over different number of domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>44</td>
<td>47</td>
<td>42</td>
<td>48</td>
<td>47</td>
<td>48</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>41</td>
<td>46</td>
<td>45</td>
<td>46</td>
<td>46</td>
<td>48</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>43</td>
<td>46</td>
<td>45</td>
<td>47</td>
<td>49</td>
<td>49</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>48</td>
<td>52</td>
<td>51</td>
<td>53</td>
<td>54</td>
<td>51</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>45</td>
<td>51</td>
<td>48</td>
<td>54</td>
<td>47</td>
<td>48</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>46</td>
<td>53</td>
<td>53</td>
<td>52</td>
<td>56</td>
<td>53</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>46</td>
<td>52</td>
<td>48</td>
<td>50</td>
<td>53</td>
<td>49</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>48</td>
<td>52</td>
<td>51</td>
<td>53</td>
<td>54</td>
<td>51</td>
</tr>
</tbody>
</table>

Table 6.13. Percentage of correct classifications under different number of training domains.
### 6.4. RESULTS IN UNKNOWN DOMAINS

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>34</td>
<td>35</td>
<td>44</td>
<td>0</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>29</td>
<td>28</td>
<td>28</td>
<td>27</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>32</td>
<td>30</td>
<td>29</td>
<td>30</td>
<td>41</td>
<td>39</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>41</td>
<td>42</td>
<td>47</td>
<td>44</td>
<td>60</td>
<td>48</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>45</td>
<td>43</td>
<td>58</td>
<td>53</td>
<td>60</td>
<td>51</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>46</td>
<td>50</td>
<td>54</td>
<td>48</td>
<td>71</td>
<td>55</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>39</td>
<td>44</td>
<td>42</td>
<td>42</td>
<td>64</td>
<td>50</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>41</td>
<td>42</td>
<td>47</td>
<td>44</td>
<td>60</td>
<td>48</td>
</tr>
</tbody>
</table>

**Table 6.14.** Percentage of precision for category Question over different number of training domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>12</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>17</td>
<td>18</td>
<td>14</td>
<td>25</td>
<td>23</td>
<td>29</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>9</td>
<td>12</td>
<td>22</td>
<td>22</td>
<td>21</td>
<td>28</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>52</td>
<td>49</td>
<td>51</td>
<td>65</td>
<td>48</td>
<td>50</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>38</td>
<td>52</td>
<td>49</td>
<td>59</td>
<td>37</td>
<td>45</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>37</td>
<td>37</td>
<td>43</td>
<td>43</td>
<td>37</td>
<td>29</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>48</td>
<td>49</td>
<td>39</td>
<td>45</td>
<td>45</td>
<td>44</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>52</td>
<td>49</td>
<td>51</td>
<td>65</td>
<td>48</td>
<td>50</td>
</tr>
</tbody>
</table>

**Table 6.15.** Percentage of recall for category Question over different number of training domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>45</td>
<td>46</td>
<td>44</td>
<td>46</td>
<td>43</td>
<td>44</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>42</td>
<td>50</td>
<td>49</td>
<td>48</td>
<td>45</td>
<td>45</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>46</td>
<td>51</td>
<td>56</td>
<td>50</td>
<td>51</td>
<td>55</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>48</td>
<td>51</td>
<td>53</td>
<td>53</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>43</td>
<td>53</td>
<td>47</td>
<td>55</td>
<td>41</td>
<td>43</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>45</td>
<td>54</td>
<td>55</td>
<td>51</td>
<td>52</td>
<td>49</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>46</td>
<td>52</td>
<td>46</td>
<td>48</td>
<td>46</td>
<td>45</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>48</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>48</td>
<td>48</td>
</tr>
</tbody>
</table>

**Table 6.16.** Percentage of precision for category Answer over different number of training domains
### Table 6.17. Percentage of recall for category Answer over different number of training domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>45</td>
<td>44</td>
<td>27</td>
<td>44</td>
<td>52</td>
<td>44</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>36</td>
<td>35</td>
<td>29</td>
<td>31</td>
<td>36</td>
<td>33</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>14</td>
<td>25</td>
<td>24</td>
<td>29</td>
<td>37</td>
<td>37</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>39</td>
<td>43</td>
<td>37</td>
<td>46</td>
<td>56</td>
<td>49</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>45</td>
<td>43</td>
<td>28</td>
<td>39</td>
<td>53</td>
<td>51</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>33</td>
<td>41</td>
<td>37</td>
<td>43</td>
<td>51</td>
<td>48</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>45</td>
<td>47</td>
<td>41</td>
<td>52</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>39</td>
<td>43</td>
<td>37</td>
<td>46</td>
<td>56</td>
<td>49</td>
</tr>
</tbody>
</table>

### Table 6.18. Percentage of precision for category Other over different number of training domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>44</td>
<td>47</td>
<td>41</td>
<td>49</td>
<td>50</td>
<td>51</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>43</td>
<td>48</td>
<td>46</td>
<td>50</td>
<td>49</td>
<td>53</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>43</td>
<td>46</td>
<td>46</td>
<td>49</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>52</td>
<td>57</td>
<td>53</td>
<td>58</td>
<td>58</td>
<td>55</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>46</td>
<td>54</td>
<td>46</td>
<td>55</td>
<td>51</td>
<td>53</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>48</td>
<td>52</td>
<td>51</td>
<td>54</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>50</td>
<td>55</td>
<td>49</td>
<td>55</td>
<td>53</td>
<td>51</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>53</td>
<td>58</td>
<td>53</td>
<td>58</td>
<td>58</td>
<td>55</td>
</tr>
</tbody>
</table>

### Table 6.19. Percentage of recall for category Other over different number of training domains

<table>
<thead>
<tr>
<th>Classifier</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag-of-Words SVM classifier</td>
<td>57</td>
<td>68</td>
<td>76</td>
<td>68</td>
<td>65</td>
<td>74</td>
</tr>
<tr>
<td>Word class n-gram classifier</td>
<td>56</td>
<td>69</td>
<td>77</td>
<td>68</td>
<td>67</td>
<td>71</td>
</tr>
<tr>
<td>Word Pattern classifier</td>
<td>86</td>
<td>81</td>
<td>80</td>
<td>73</td>
<td>73</td>
<td>72</td>
</tr>
<tr>
<td>Common words classifier</td>
<td>54</td>
<td>62</td>
<td>67</td>
<td>55</td>
<td>56</td>
<td>53</td>
</tr>
<tr>
<td>Statistical classifier</td>
<td>48</td>
<td>60</td>
<td>69</td>
<td>68</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>Committee classifier</td>
<td>63</td>
<td>70</td>
<td>73</td>
<td>65</td>
<td>71</td>
<td>69</td>
</tr>
<tr>
<td>Markov classifier</td>
<td>46</td>
<td>57</td>
<td>59</td>
<td>48</td>
<td>61</td>
<td>51</td>
</tr>
<tr>
<td>HMM classifier</td>
<td>64</td>
<td>62</td>
<td>67</td>
<td>55</td>
<td>56</td>
<td>53</td>
</tr>
</tbody>
</table>
Chapter 7

Discussion and Conclusions

In this chapter the results from chapter 6 will be discussed in detail as well as conclusions from the thesis and further work that could be done in the area.

7.1 Discussion

7.1.1 Performance of the classifiers

The first test that was done using single domains showed that there is some difference in the results of the classifiers. The differences when looking on the mean values of the results (see table 6.3) were quite small. The Statistical classifier, Common words classifier and HMM classifier showed slightly better results than the others. If the results are studied in details there seems to be quite a big difference between the classifiers for specific categories. For one of the domains (D2, Soup making), the Word Pattern classifiers result outperforms every other classifier even tough the total mean result is in line with the other classifiers (see table 6.2). A conclusion from this is that the performance of a classifier over different domains can vary very much and that no single classifier can be named the best.

When comparing the classifiers over several different domains, see figure 7.2, it can be noted that, as expected, the results drops at first when going from a single domain to more domains and then very slowly starts to get better as more domains are added. The first drop is most likely accounted to the fact that fewer features or markers are valid across several domain than on one. The slow increase of the results after the drop is most likely due to the bigger training set that is available. After the first initial drop (from one to two domains) it is likely to compensate more than the addition of new domains.

When testing over unknown domains (see figure 7.2), the trend is that the result actually increases with the number of training domains. The reason for this is probably that over-training on domain specific markers should decrease when using more domains, giving more weight to domain independent markers.

A conclusion is that training over multiple domains does not necessarily improve the result over the trained domains but seems to improve the results on unknown
domains. When looking at the different classifiers, the ones that performs best over multiple domains both when classifying over unknown and known domain, are the Common Words Classifier and the Committee Classifier.

When comparing all classifiers to the naive classifier, it is clear that they all perform better than it. The biggest difference is in the test with training over one domain where the difference is very large for some classifiers (Markov classifier and Statistical classifier). As more domains are added the performance of the classifiers decrease, but they are still better than the baseline. It is hard to speculate on the cause of this decrease in performance but one cause could be that the increase in training data when using multiple domains is not enough to overcome the increased complexity of the problem.

![Figure 7.1. Classifier results when training and testing over multiple domains](image1)

![Figure 7.2. Classifier results when training over multiple domains and testing on unknown domains](image2)
7.1. DISCUSSION

7.1.2 Differences between domains

As can be seen in table 6.2, there are notable differences in the results between different domains. By only looking at the best result, the risk of the classifiers being unsuitable for the current domain is minimized. We can see that there are quite big differences between the results from about 44% up to 70%.

For example D1 has considerably worse results than D6 for all classifiers. As noted above many categories with generally poor results have a few classifiers that performs well above the others. It is therefore likely to assume that some domains such as D1 is harder to classify since no single classifier managed to get very good results.

Of course other error sources could be that dome domains require a larger training corpus or that the set of categories did not fit very well for that domain.

7.1.3 Differences between the speech acts

If the results are analyzed according to the different speech acts, notable differences can be found. In general, questions seems to be the category that is by far the hardest to find. For all tests it performs a bit below the other two categories.

The reasons for this could be that questions are more scare in discussions, so the available corpus is smaller than that of the other two categories. Another major problem with questions is the way the text is chunked into smaller parts. The chunks are created after just the disposition and layout of the text, which could potentially cause problems. If we had a chunk consisting of an answer and then appended a question in the end as often is done, it should be classified as a question and not an answer. This is a problem that is not well handled by statistical methods since they handle the general characteristics of the chunks and don’t look in to small sequences. One solution to this problem could be to look at a deeper semantical level when searching for questions.

7.1.4 Error sources

In general the results was worse than what I had expected during development. Error sources for this could be the following:

Unsuitable set of speech acts The selection of speech acts might not be optimal. If the speech acts overlap each other to much, good classification might be hard to achieve even for a human classifier, let alone make it harder for an automated classifier.

Small corpus size The size of the used corpus was quite small, it contained about 2000 chunks of text. A bigger corpus would more likely get better results. The small corpus used would also favor classifiers that do not require large corpora for good performance and punish classifiers that needs large corpora to achieve good performance that might be better with a large enough corpus.
7.2 Conclusions

One main conclusion is that classifying texts after different speech acts is definitely a possible and viable approach even though the results are not extremely good, depending on the application. An application where the user could help supply the training data (krautsourcing) would over time overcome the problem with a small corpus and would be a good way to test the limits.

What can be learned from the work in this thesis is that there seems to be no single best feature set when classifying over just one single domain. The variable differences of the domains make different classifier and feature set perform well. When looking on multiple domains the Committee Classifiers along with the Common Words classifiers seems to be the best.

There also seems to be quite a lot of differences between different speech acts, some speech acts appear to be easier to classify than others for varying reasons, good selection of speech acts should be able to improve results.

7.3 Future work

For further research in the area of speech acts classification there are a few useful ideas that came up while working on this thesis. These I have been unable to pursue.

While working on this thesis, the lack of an already existing corpus that could be used has been a limitation. Due to the time frame and scope of the thesis, the corpus created was not very big. Obviously this has been a problem and most likely affected the results of the constructed classifiers. Creation of a bigger corpus or tagging an existing one with suitable speech acts would be very useful and a major step towards constructing better and more efficient speech act classifiers.

Usually when discussing and comparing different methods and algorithms for classification and NLP tools in general statistics of achieved results are deemed to be very important. But one important question that rises when usability for speech act classification is discussed, is how good does a system have to be to be useful in a real life situation?

There is a well known fact that some kind of tasks require different characteristics of the result. For example when it is very important to find all errors, high recall with acceptance of bad precision can be tolerated. But what are the priorities for a user, how good must a system performs to be considered useful, what is most important recall or precision? Research in this area would be useful to help focus research and development to benefit user experience.

There has been one problem that have been very real when working on this thesis, the lack of a good and widely accepted set of categories for use on the web, even though finding a good set of categories is a project that would have to be evolving and revised since the web always changes. It would have been a great aid in creating good speech act classifiers and also essential to building a large corpus that would be useful for many different researchers and developers.
Bibliography


BIBLIOGRAPHY


Appendix A

Appendix 1 - Parameters for classifiers

Bag of word classifier
Machine Learning implementation
LibSVM

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max number of selected words</td>
<td>90</td>
</tr>
<tr>
<td>Minimum number of occurrences</td>
<td>3</td>
</tr>
</tbody>
</table>

Word class n-gram classifier
Machine Learning implementation
J48

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-gram max length</td>
<td>5</td>
</tr>
<tr>
<td>Minimum number of occurrences</td>
<td>20</td>
</tr>
</tbody>
</table>

Word pattern classifier
Machine Learning implementation
SMO

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-gram max length</td>
<td>5</td>
</tr>
<tr>
<td>Minimum number of occurrences</td>
<td>3</td>
</tr>
</tbody>
</table>
Common words classifier
Machine Learning implementation
SMO

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of common words used</td>
<td>100</td>
</tr>
</tbody>
</table>

Statistical classifier
Machine Learning implementation
SMO

Committee classifier
Classifiers used
- Common word classifier
- Pos N-gram classifier
- Statistical classifier
- Word pattern classifier

Markov classifier
Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order of classifier</td>
<td>4</td>
</tr>
</tbody>
</table>

Regular classifier
Common word classifier

HMM classifier
Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order of classifier</td>
<td>4</td>
</tr>
</tbody>
</table>

Regular classifier
Statistical classifier
Appendix B

Appendix 2 - Selected groups

- Bands United - http://launch.groups.yahoo.com/group/bandsunited/
- Soup Craft - http://groups.yahoo.com/group/Soap_Craft/
- Beachcats - http://groups.yahoo.com/group/beachcats/
- Ceramic Lovers - http://groups.yahoo.com/group/CeramicLovers/
- Java Official - http://tech.groups.yahoo.com/group/Java_Official/
- Distillers - http://groups.yahoo.com/group/Distillers/
- Lego - http://groups.yahoo.com/group/Lego/

All of the groups was visited on 2007-09-10.
Appendix C

Appendix 3 - The data set

<table>
<thead>
<tr>
<th></th>
<th>Number of entries</th>
<th>Number of words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Questions</td>
<td>508</td>
<td>20572</td>
</tr>
<tr>
<td>Answers</td>
<td>1126</td>
<td>46460</td>
</tr>
<tr>
<td>Other</td>
<td>1391</td>
<td>50225</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>3025</strong></td>
<td><strong>117 257</strong></td>
</tr>
</tbody>
</table>