NOTE TO USERS

Université d'Ottawa · University of Ottawa

Université d'Ottawa · University of Ottawa

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Magdalena WIDLAK

AUTEUR DE LA THÈSE - AUTHOR OF THESIS

M. Sc. (Computer Science)

GRADE - DEGREE

School of Information, Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT - FACULTY, SCHOOL, DEPARTMENT

TITRE DE LA THÈSE - TITLE OF THE THESIS

Influence of Word Sense Disambiguation Performance of Texte Classification

S. Matwin

DIRECTEUR DE LA THÈSE - THESIS SUPERVISOR

CO-DIRECTEUR DE LA THÈSE - THESIS CO-SUPERVISOR

EXAMINATEURS DE LA THÈSE - THESIS EXAMINERS

N. Japkowicz

F. Oppacher

J-M. De Koninck, Ph D

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

DEAN OF THE FACULTY OF GRADUATE AND POSTODORAL STUDIES

Influence of Word Sense Disambiguation on Text Classification

by **Magdalena Widlak**

Thesis submitted to the Faculty of Graduate and Postdoctoral Studies in partial fulfillment of the requirements for the degree of

Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Your file Votre référence
ISBN: 0-494-01643-4
Our file Notre référence
ISBN: 0-494-01643-4

Acknowledgements

I want to thank my supervisor, Dr. Stan Matwin, for his continuous support and help during my graduate study, and for his valuable advice in writing this thesis. I want to also thank the faculty, staff and students from the Computer Science Department for their help and support.

Abstract

Word sense ambiguity is a pervasive characteristic of natural language. The discrimination of word senses, word sense disambiguation, is considered to be of prime importance for many areas involving computerized language analysis, from machine translation to information retrieval. Text classification, as a growing subfield of information retrieval, is also believed to suffer from the effects of word sense ambiguity.

The purpose of this thesis was to evaluate how word sense disambiguation affects text classification. The intuitive hypothesis is that word sense disambiguation aids the task of text classification.

In order to evaluate the influence of word sense disambiguation on text classification three different corpora of text documents were disambiguated manually. Classification of both original and corresponding disambiguated data was performed using four different classification systems employing four different learning approaches: decision trees (C5.0), decision rules induction (Ripper), Naive Bayes (Rainbow) and support vector machines (LibSVM). Results obtained from the classification were compared using various evaluation methods.

The results do not support the stated hypothesis very strongly. In some cases word sense disambiguation improved the results of text classification, in other cases there was no improvement or the results were worse. The difference in classification results obtained on original and disambiguated data are in most cases insignificant, that is, even though there is a slight difference in average errors, we cannot conclude that this difference is statistically significant. Some general tendencies can be observed when it comes to performance of specific classification systems. We can also infer which of the corpora were "easier" to classify than other.

The contribution of this Thesis

The main contribution of this thesis is evaluation of a hypothesis stating that word sense disambiguation has a beneficial effect on text classification. Word sense disambiguation is considered to be a potential source of error for many of the natural language processing fields. This thesis reports on results obtained from classifying both disambiguated and non-disambiguated text documents.

The second contribution is manual disambiguation of three corpora of on-line documents. The process of manual disambiguation is described in detail.

The third contribution is performing extensive empirical study involving classification of both original (non-disambiguated) and disambiguated corpora. Four different classification systems were used in order to evaluate the influence of word sense disambiguation thoroughly. The results obtained from classifying original and disambiguated texts were compared using various evaluation methods.

Table of Contents

List of Figures

List of Tables

1. Introduction

Along with the enormous growth of on-line information, grows a need of developing methods to help organize and navigate through such amounts of data. One of the areas that has drawn a lot of interest of researchers in the recent years is that of an automated text categorization. Text categorization (a.k.a. text classification) is the activity of labeling natural language texts with thematic categories from a predefined set. It can be applied to document indexing, document filtering, populating hierarchical catalogues of Web resources and other applications requiring document organization, such as putting documents in specific mailboxes or newsgroups.

One of the problems that almost any field involving natural language processing faces is word sense ambiguity – where one word can depict more than one meaning. It is a potential source of error in machine translation, information retrieval ([LEW97], [KRC92]) or speech recognition. Text classification, as a subfield of information retrieval, is also considered to suffer from the effects of word sense ambiguity. Word sense disambiguation is one of the fields in natural language processing dealing with this particular challenge.

The purpose of this thesis is to discover how word sense disambiguation influences text classification. In order to do that, both original (ambiguous) and same, but disambiguated, documents were classified, and results of the classification compared. For the lack of reliable automatic disambiguation tools, the documents were disambiguated manually, with some help of WordNet. Then, both types of corpora were classified using several available classification systems – C5, Ripper, Rainbow and LibSVM to see whether the choice of a classifier may affect the classification results.

The initial classification experiments handled balanced or mildly imbalanced data sets. The results were compared in three different ways: (1) difference between average errors on original data versus errors on disambiguated data, (2) difference between average errors and standard deviations on original and disambiguated data in order to see whether

the difference is significant, and (3) paired t-test of statistical significance, also to see whether the difference is statistically significant. These initial experiments showed that disambiguation does not make a significant difference in case of balanced data sets, however shows improvement more often in case of mildly imbalanced data sets. Therefore, a new set of experiments was performed for strongly imbalanced data sets. In this case precision and recall were compared.

The results of all experiments do not offer a strong answer to the stated question. In some cases the disambiguation helped to classify documents better, in some other there was no improvement, or the results were worse. Only in isolated cases the results obtained on disambiguated texts were significantly better or significantly worse than those on ambiguous texts. Some general statements can be made about performance of each system and difficulty of classifying each of the corpora.

1.1 Text Classification

Due to the increased availability of online documents and the growing need to access and handle them, content-based document management tasks, known as information retrieval (IR), received a lot of interest and gained an important status in the information systems field. One of those tasks is text classification (TC) – labeling natural language documents with thematic categories.

Modern approaches to text classification use machine learning (ML) techniques to induce a classifier from a training set – a set of documents with near-identical statistical properties to the documents which are to be later classified. Later thus obtained classifier is used to categorize previously unseen documents. In ML terminology the classification problem is a task of *supervised learning*, since the learning is "supervised" by the knowledge of the training set examples along with the categories they belong to.

Text categorization can be formally defined as the task of assigning a boolean value to each pair $<d_j, c_i> \in D \times C$, where D is a domain of documents and $C = \{c_1, ..., c_{|C|}\}$ is a set of predefined categories. A value of T assigned to $<d_j, c_i>$ indicates a decision to file d_j under c_i while the value of F indicates a decision not to file d_j under c_i. More formally, the task is to approximate the unknown *target function* $\Phi' : D \times C \rightarrow \{T, F\}$ (that describes how documents ought to be classified) by means of a function $\Phi : D \times C \rightarrow \{T, F\}$called the *classifier* (aka *rule* or *hypothesis* or *model*) such that Φ' and Φ "coincide as much as possible". This coincidence is called *effectiveness*. [SEB02]

Text classification is now being applied in many contexts, such as document indexing, document filtering, and in general any application requiring document organization. Spam control, organizing documents into folders, mailboxes or newsgroups and recommender systems are practical problems where automatic text classification can offer significant help.

The key resource in the machine learning approach to text classification are preclassified documents, since it is this initial corpus that a system uses to build a learner. Once a

learner is built, its effectiveness must be evaluated. In order to do that the initial corpus is split into two sets:

- a *training set* (pairs of instances and the categories they belong to) used to built a classifier
- a *test set* used to measure the effectiveness of the classifier. The test set is fed to the classifier and the classifier decides on the categories the instances belong to. Then the classifier's decisions are compared against the original categories. The measure of classifier's effectiveness is based on how often both values match.

The documents in test set cannot participate in the construction of the classifier.

This method is called the *train-and-test* approach. An extension to the train-and-test method is the *k-fold cross validation* technique [MIT97]. The initial corpus is divided into k parts. One of the parts is used as a test set while the remaining k-1 parts are used as a training set. The process repeats for each of the k parts. The overall effectiveness is an average of the k individual results.

There are several approaches to inducing text classifiers. The most popular ones are

- probabilistic, for example based on the Bayes theorem
- decision tree classifiers
- decision rule classifiers
- support vector machines
- example-based classifiers (NN method)

and other.

In order to check or eliminate the sensitivity of a particular inductive method to word sense ambiguity, four different systems were used:

- Rainbow (NaiveBayes classifier)
- Ripper (decision rules)
- C5.0 (decision trees)
- LibSVM (support vector machines)

Each of the systems is described in the Section 2.2 Machine Learning Resources.

1.2 Word Sense Disambiguation

Many English words carry more than one meaning or sense. These meanings can be fairly close or entirely different. The actual sense of a particular ambiguous word can only be determined by examining the context it appears in.

One of the most common examples presented when discussing word sense ambiguity is the word "bank". It has several different meanings, two of which are perhaps the most common and distinct: *an edge of a river* sense, and a *financial institution* sense.

"He leapt from a bank into the river."
"She deposited her cheques at the bank."

We can, without difficulty, infer the *edge of the river* sense in the first sentence and the *financial institution* sense in the second sentence. However, despite the fact that humans can usually deal with this problem effortlessly, it has been quite difficult to define it formally and discover a reliable automatic method to disambiguate natural language texts.

Below we have an example of three different text documents found on the Internet. All three of them mention the word "bank" several times, and in each article this word is used in a different sense. If these documents were to be classified, they could be filed into the same category based on the frequency of this word. However, if we look at the three documents closely, we will immediately notice that their topics are unrelated. The first one advertises a financial institution called River Bank (despite the word "river" this *bank* does not refer to an edge of a river). The second one discusses problems related to erosion of actual river banks. The third article contains advice on how to fix computers. The word bank appears in it in the sense of a computer memory bank.

River **Bank** is regarded as a high quality, well staffed financial institution in Western Wisconsin. Its friendly tellers, customer relations staff and loan officers have given the **Bank** a growth pattern that is greater than any other of the area's **banks**. They have a combined lending experience of well over one hundred years.
The River **Bank** was established to service the financial needs of the community, its businesses and its citizens; no line of financial services is beyond its charter as long as the **bank** is serving the financial needs of businesses and families in our community.
(http://www.riverbank.biz/)

.....

This project aims to investigate **bank** erosion and deposition processes at sites where **bank** sediments are very dissimilar, non cohesive and cohesive, but flows, sediment loads and bed material calibre are comparable. This will identify and explain the effect of **bank** sediment type on channel morphology. Sites will be chosen on alluvial rivers which are in dynamic equilibrium and which flow through former glacial lake beds (e.g. River Severn through former Lake Vrynwy). In the approach to the former lake, the river is wide, relatively shallow and moderately sinuous as it freely migrates through its own slightly cohesive alluvial deposits. Immediately downstream the river is very sinuous, narrow and deep, even though there is no change in discharge, load and sediment calibre, due to the cohesiveness of the former lake deposits which now make up the river **banks**. This illustrates that there are an infinite number of possible channel morphologies that can transmit a given discharge, calibre and volume of sediment, which is also borne out by mathematical models, but that the actual morphology is controlled by the nature of the **bank** sediment.
http://www.uea.ac.uk/env/studentships/21.htm

.....

The EPoX 8RDA+ motherboard has 3 DIMM memory slots, so I'll just be using two of them. The 3 DIMM slots are divided into two **banks**, which you can actually see by the physical separation of the memory slots on the motherboard. The single memory slot nearest the bottom of the picture is **bank** 0. The two memory slots above it are **bank** 1. With two memory modules, the best performance is realized by inserting one memory module in **bank** 0 and one memory module in **bank** 1. There's only one memory slot in **bank** 0 so it's pretty clear that's the one to use. There are two memory slots in **bank** 1. Most people say it doesn't matter which memory slot is used, but some people insist that the lower of the two, which is DIMM3, gives better performance. So that's the one I used. And just so you know, best performance is with either two memory modules in separate **banks** or three memory modules. Different motherboards will group and order the memory slots differently, so consult the owner's manual for your own motherboard for details
http://www.mysuperpc.com/computer_assembly/pc_install_ram_system_memory.shtml

Fig. 1 Example of three different online texts, all using the word „bank" in three different senses: *financial institution* sense, an *edge of a river* sense, and a *computer memory bank* sense. All these documents may be classified into the same category, unless the different meanings of the word „bank" are taken into consideration.

While the sense of an *edge of a river* is quite different from the other two senses, the *financial institution* and a *slot for a computer memory* senses do have something in common – they both refer to a place where something is held available, a depot, collection or a storage. Another related sense could be a *blood bank*, a place where human blood is kept for medical reasons. (http://www.m-w.com/cgi-bin/dictionary Merriam-Webster)

We can see (Fig. 1) that the appearance of the word *bank* in all three articles may affect the classification of these documents. However, if each sense of this word was marked as being a different word by attaching a numerical label to it, such that, for example *bank1* – financial institution, *bank2* – edge of a river, *bank3* – computer memory bank, the potential confusion is avoided. The three documents not only do not contain the possibly confusing word *bank* any more, but are described by three distinct words *bank1*, *bank2* and *bank3* which helps differentiate between them.

River **Bank1** is regarded as a high quality, well staffed financial institution in Western Wisconsin. Its friendly tellers, customer relations staff and loan officers have given the **Bank1** a growth pattern that is greater than any other of the area's **banks1**.

.....

This project aims to investigate **bank2** erosion and deposition processes at sites where **bank2** sediments are very dissimilar, non cohesive and cohesive, but flows, sediment loads and bed material calibre are comparable. This will identify and explain the effect of **bank2** sediment type on channel morphology.

.....

The EPoX 8RDA+ motherboard has 3 DIMM memory slots, so I'll just be using two of them. The 3 DIMM slots are divided into two **banks3**, which you can actually see by the physical separation of the memory slots on the motherboard. The single memory slot nearest the bottom of the picture is **bank3** 0. The two memory slots above it are **bank3** 1.

Fig. 2 Parts of documents in Fig.1 with the different senses of the word bank clearly marked as bank1, bank2 and bank3.

Thus disambiguated documents (Fig. 2), where ambiguous words are clearly marked to make them distinct words, should intuitively be easier to classify by automatic categorization systems. The purpose of this thesis is to measure how much, if at all, words sense disambiguation influences text categorization.

1.3 Methodology

This section gives an overview of the methodology applied to carry out the experiments described in this thesis. The sections that follow describe the process in more details.

The purpose of this thesis was to compare the effectiveness of a classifier built on a non-disambiguated corpus of text documents versus the effectiveness of a classifier built on a disambiguated corpus, therefore the need to have same text documents in both forms, as shown in Fig1 and Fig2.

For these experiments three different non-disambiguated (from now on referred to as *original*) corpora were used, O-Corpus$_1$, O-Corpus$_2$ and O-Corpus$_3$ (Section 2.1.1 Corpora). Each document in all three corpora was disambiguated manually, with help of WordNet (Section 2.1.2), Brill's Parts of Speech Tagger (Section 2.1.3) and some scripts aiding the manual labor (detailed description of the process in Section Disambiguation). The process of disambiguation resulted in documents of the same content, but where ambiguous words were tagged with appropriate numerical tags indicating different senses of those words. That produced corresponding disambiguated corpora D-Corpus$_1$, D-Corpus$_2$ and D-Corpus$_3$.

Once the process was completed, we used the O-Corpus$_i$ and D-Corpus$_i$, i=1,2,3, to build classifiers. We used four different systems: Learner$_1$, Learner$_2$, Learner$_3$ and Learner$_4$, using four different algorithms, as described in Section 2.2 Machine Learning Resources.

We used each O-Corpus$_i$ and Learner$_j$, i=1,2,3 and j=1,2,3,4 to obtain 12 O-Classifier$_{ij}$ and O-Results$_{ij}$ measuring classification error of each O-Classifier$_{ij}$. The process was repeated for D-Corpus$_i$ and each of the Learner$_j$ resulting in 12 D-Classifier$_{ij}$ and D-Results$_{ij}$. Then the corresponding O-Results$_{ij}$ and D-Results$_{ij}$ were compared to measure how, if at all, they differ.

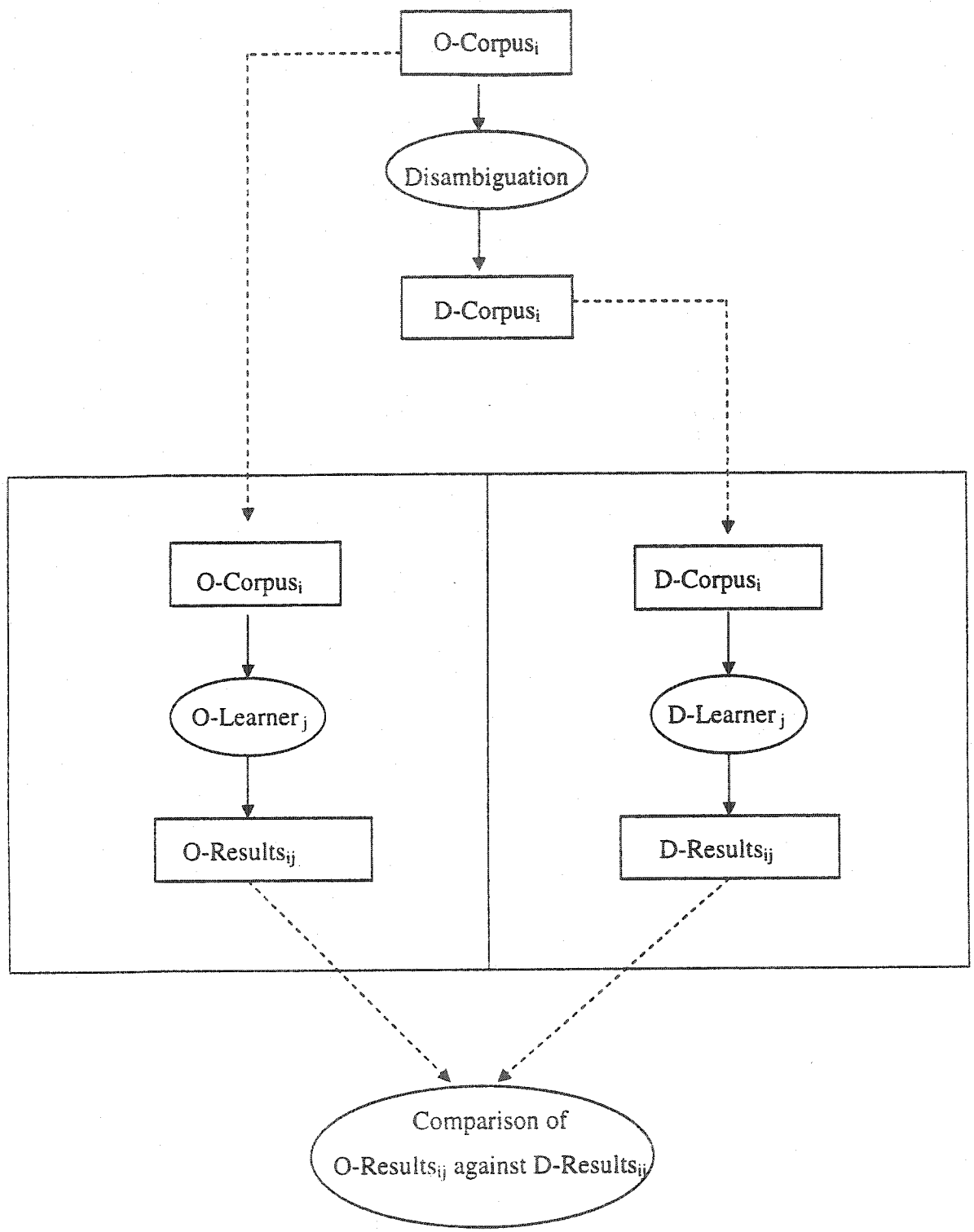

Fig. 3 Flowchart of the methodology applied in the experiments described in this thesis.

2. Resources

Several resources have been used in the experiments described in this thesis. We can categorize them into two groups:

(1) Lexical Resources, which include:

- Three corpora – 20 Newsgroups, Yahoo Discussion Groups and Digitrad, along with the description of necessary formatting needed for the experiments
- WordNet, a Machine Readable Dictionary
- Brill's Parts of Speech Tagger

(2) Machine Learning Resources – Learning Algorithms:

- C5.0
- Ripper
- Rainbow
- SVM.

2.1 Lexical Resources

The focus of this thesis is to compare performance of various classification systems on disambiguated documents versus original documents in order to see whether disambiguation aids the task of the automatic text categorization.

The main resource of any text classification experiment is a corpus of text documents. In the experiments described in this thesis three different corpora were used in order to thoroughly compare results of the experiments.

The corpora are in the original version, that is, they contain ambiguous words. They had to be disambiguated, and the task was performed with help of WordNet and Brill's Parts

of Speech Tagger, which description can be found in Sections 2.1.2 WordNet and 2.1.3 Brill's Parts of Speech Tagger.

2.1.1 Corpora

The difficulty of classifying natural language text is influenced by the type of documents to be classified. Documents written in a formal language, focusing on one topic and containing no ambiguities may be easier to classify than documents using colloquial language, not respecting spelling rules and not concentrating one theme. News stories written by professional writers are commonly considered to fall into the first category of documents that may be less problematic to classify. Internet however encourages use of informal language and is not strict about spelling and grammatical rules or keeping a particular theme, so not all stories are written as properly as news stories or research papers. E-mails, as a more informal means of written communication, might present a more difficult classification task.

Two of the most commonly used by text classification researchers corpora are *Reuters-21578*, and *20 Newsgroups*. The documents in the *Reuters-21578* collection are news stories that appeared on the Reuters newswire in 1987. *20 Newsgroups* data set consists of about 20000 messages taken from 20 Usenet newsgroups

Since the focus of this thesis is to measure how word sense ambiguity influences text classification, we decided to choose texts that are likely to use more ambiguous words. Since *Reuters-21578* is a collection of news stories written in a clear and concise formal language, they probably do not contain a lot of ambiguities. Intuition suggests that e-mails, given their usually informal nature, would present themselves to be an appropriate subject to investigate, and therefore the *20 Newsgroups* set was used in the experiments.

The *20 Newsgroups* corpus is one of the three corpora used in the experiments. The second one is also a collection of e-mails, originating from *Yahoo Discussion Groups* [http://www.groups.yahoo.com]. The third corpus is vastly different from the first two –

it is a collection of folk songs lyrics, *Digital Tradition*, collected and maintained by MudCat Cafe [GRE96], and first used for classification tasks by Scott [SCM98]. Given a usually unstructured and vague nature of song lyrics, we considered this corpus to be an interesting challenge for our experiments.

2.1.1.1 20 Newsgroups

Usenet is a world-wide distributed discussion system. The system is broken into newsgroups with names that are classified hierarchically by subject. It is a public forum, where anyone with an e-mail account can post an article or a message. Because if its huge popularity is has become an enormous, constantly growing corpus of online documents.

Since the newsgroups are divided by theme or topic, they lend themselves easily for a text classification task using newsgroup names as class labels. Classification of newsgroup posts is a realistic, but, as mentioned before, also a difficult task. Apart from the fact that these texts are not written in formal, well structured language, the users of newsgroups do not always feel restricted to discussing the official topic, which may result in documents that should not belong to a particular group, or, in our case, to a particular class. Another problem is caused by cross-posting where one document may be sent to several newsgroups and therefore belongs to more than one class.

The 20 Newsgroups corpus [http://www.ai.mit.edu/~jrennie/20Newsgroups/] is a popular data set used for machine learning experiments applied to text, such as text classification. The original owner and donor is Ken Lang.

It is a collection of about 20000 Usenet articles taken from 20 newsgroups:

alt.atheism
comp.graphics
comp.os.ms-windows.misc
comp.sys.ibm.pc.hardware
comp.sys.mac.hardware
comp.windows.x
misc.forsale
rec.autos

rec.motorcycles
rec.sport.baseball
rec.sport.hockey
sci.crypt
sci.electronics
sci.med
sci.space
soc.religion.christian
talk.politics.guns
talk.politics.mideast
talk.politics.misc
talk.religion.misc

Each newsgroup has its own directory (called by its name), and about 1000 documents in it. Each document is a typical e-mail message (posting) including subject lines, signatures and quoted portions of other postings. Each document is stored in a separate file.

For our experiments only a subset of the entire collection was used. We chose the following newsgroups:

GROUP	TOPIC
alt.atheism	atheism
comp.graphics	computer graphics
comp.sys.ibm.pc.hardware	IBM PC hardware
comp.sys.mac.hardware	MAC hardware
sci.med	medicine

The choice of the above newsgroups was dictated by a need of having documents that would be very close and very different in topic. We are assuming that, for example, e-mails in *alt.atheism* and *sci.med* would cover vastly different subjects and use very different language. On the other hand, we expected *comp.sys.ibm.pc.hardware* and *comp.sys.mac.hardware* to be fairly close when it comes to vocabulary and theme.

Since the data was to be disambiguated by hand - a tedious and a very time consuming job - only 100 articles from each group were chosen. That was also the case with the other corpora, where we always worked on 100 examples of each category.

2.1.1.2 Yahoo Discussion Groups

Usenet was one of the first internet services. Since it turned out to be an enormous success, other web portals decided to offer its users a possibility of forming similar forums, called discussion mailing lists or discussion groups. Yahoo [http://www.yahoo.com], aside from providing a search engine, free e-mail services and news, also holds several hundreds of discussion groups. Anyone can start a mailing list, but the difference between Yahoo groups and Usenet groups is that its members often must join (sign up to) a particular group in order to post, and often in order to read, messages on this group.

Yahoo Discussion Groups data set is a collection of e-mails of 8 Yahoo Discussion Groups [http://groups.yahoo.com/], whose archives are open to public. The nature and format of this corpus is very similar to the previous one, with the exception of a header and subject line removed. They also include signature files and quoted portions of previous postings.
We chose the following newsgroups:

GROUP	TOPIC
machine-learning	machine learning
perl-beginner	PERL for beginners
distantsuns	astronomy
digital-photography	digital photography
The_Unhandled_Horse	horsemanship
avconnection	african violet connection
ChildrenfreeUK	welfare of childeren in UK
CelticLore	Celtic music

This set consists of groups whose topics are mostly unrelated.

2.1.1.3 DigiTrad

The Digital Tradition (DigiTrad) was first introduced for classification tasks by Scott and Matwin [SCM98]. It is a collection of about 9000 folk songs lyrics, freely available at the "MudCat Cafe" [GRE96].

Most songs have been assigned a set of keywords taken from a fixed list. The keywords usually indicate a topic of a song (i.e. marriage, death), but can also point at the geographic origin or the genre of the song (i.e. Irish, gospel).

The DigiTrad collection has been preprocessed and made easily usable for machine learning experiments by Scott [SCM98]. The text is marked with SGML tags. Title and end notes of each song are delimited by the <TEXT> tags. The keywords are put at the top, before <TEXT>, and are delimited by <TOPIC> and <D> tags.

</TEXT>
<TOPICS><D>**English**</D><D>**sailor**</D><D>**battle**</D></TOPICS>
<TEXT>THE 23RD OF FEBRUARY

On the 23rd of February,
The weather being clear.
We spied to be sure seven Turkish men-of-war,
Come a sailing from Algier.

Ch.
To me ri fol leather ol,
Rol a diddle i,
rol a diddle i,ol lay.
to me ri fol leather ol lay.

Well the very first ship to come alongside,
Was called the Green Pea.
We fired to her a warning shot,
And quickly she did flee.

filename[FEB23
AG
apr97

</TEXT>
<TOPICS><D>**parody**</D><D>**seasonal**</D></TOPICS>
<TEXT>THE 8 DAYS OF CHANNUKAH
(Tom Paley?)

On the first day of Channukah, my bube gave to me
A bagel mit a schtick lox.

Two kreplach
Three gefilte fishes
Four knadlech
Seven pickled herrings
Six kosher dills
bubkis* (omit others on this day)

*: nothing
Note: remembered imperfectly, but should be OK.
filename[CHAN8DAY
play.exe XMAS12DY
RG

</TEXT>

Fig. 4 Example of two song lyrics from DigiTrad, as received from Sam Scott [SCO98]. Keywords are marked in bold.

The keywords listed at the top of each song are a good indication of a category the text belongs to. However, some of the songs have no keywords, while other may have multiple keywords – which may indicate a song belonging to zero or to more than one class. Some keywords appear only a few times across the entire collection, while other are attributed to many songs. There are 672 keywords altogether.

For the purpose of our experiments we decided to have 6 data sets, each consisting of 100 examples (one song being one example). Class labels were represented by 1 keyword and each example was to belong to only one category.

Out of all the keywords top 6 returning biggest number of songs were chosen. Those were:
Irish
Scots
Political
Death
Marriage
Love

These sets were not mutually exclusive – some songs had more than one of the 6 keywords in their keyword set, which violated the decision of having songs which belonged to only one category. Therefore, only those songs which set of keywords contained only one of the chosen 6 keywords were extracted, and out of these sets 100 examples were chosen (first 100). This way the final data set was mutually exclusive and contained no examples belonging to more than one category.

2.1.1.4 Data Formatting

One of the most common ways of representing text for purpose of text categorization is through a "bag of words" technique. All words found in the entire data set form a vector of attributes. Each vector corresponds to one document. If a particular word appears in the given document, it is represented by a 1, and its absence is represented by 0. Therefore, the attribute vectors are strings of binary 0's and 1's.

This method produces vectors of a very high dimensionality, which may be problematic for many systems to handle. It may also cause overfitting, which is a phenomenon where a classifier is tuned to the contingent characteristics of the training data rather than to the constitutive characteristics of categorization. Classifiers that overfit the training data are very good at reclassifying the data they had been trained on, but do significantly worse on previously unseen data. Overfitting may be avoided if the dimensionality of data is reduced.

The first step in reducing dimensionality is removal of stop words – words that are very often used and therefore carry no crucial informative value, such as articles, pronouns, prepositions, conjunctions and other. This step was performed on the three corpora used in the experiments. The list of stopwords was taken from the Internet (https://securesite.chireader.com/Archive/stopwords.txt).

Aside from removing stopwords in all the corpora, the data was cleaned of standalone non-numerical characters, and words that were less than 3 letters long.

More sophisticated terms removal was not applied to the documents. The focus of this work was to see whether a change made to some words would result in better classification. An advanced terms removal might have resulted in discarding the words in question, therefore the process was omitted in order to avoid the possibility of loosing terms crucial to this work.

2.1.2 WordNet

WordNet is a Machine Readable Dictionary developed at Princeton University by a group led by George Miller [MIL90]. It currently contains about 120,000 sets of noun, verb, adjective and adverbs organized in synonym sets (synsets). Each synset represents a concept, and contains all word forms that can refer to a given concept, a definitional gloss and, in most cases, an example sentence. The basic semantic relation between words in WordNet is the synonymy relation.

```
Synonyms/Hypernyms (Ordered by Frequency) of noun bank
9 senses of bank

Sense 1
bank, side
     => slope, incline, side
Sense 2
depository financial institution, bank, banking concern, banking company
     => financial institution, financial organization
Sense 3
bank
     => ridge
Sense 4
bank
     => array
Sense 5
bank
     => reserve, backlog, stockpile
Sense 6
bank
     => funds, finances, monetary resource, cash in hand, pecuniary resource
Sense 7
bank, cant, camber
     => slope, incline, side
Sense 8
savings bank, coin bank, money box, bank
     => container
Sense 9
bank, bank building
     => depository, deposit, repository
```

Fig. 5 WordNet's synset of the noun "bank".

WordNet was used during the process of disambiguation, as described in Section 3.1.2 Manual Disambiguation.

2.1.3 Brill's Parts of Speech Tagger

The role of a part of speech tagger is to assign appropriate parts of speech to words, and as such it offered significant help in the process of disambiguation. Brill's Pars of Speech tagger offers reliable accuracy of 92.98% [BRI92], and therefore was chosen for the experiments.

2.2 Machine Learning Resources: Learning Algorithms

Text classification research to date has used many various methods to induce classifiers for text documents. The most successful approaches include probabilistic (Naïve Bayes), decision trees, decision rule induction, and support vector machines (SVM).

For the experiments described in this thesis, four different learning algorithms were used:

- Naïve Bayes (Rainbow)
- Decision Tree (C5.0)
- Rule Induction (Ripper)
- Support Vector Machine (LibSVM)

One of the commonly used systems for machine learning and data mining experiments is a WEKA package (http://www.cs.waikato.ac.nz/~ml/weka/), which is a collection of machine learning algorithms, including Naive Bayes, C4.5, SVM, and many other. Unfortunately WEKA was not able to process data of dimensionality as high as the one used in our experiments.

What follows is a short description of each learning algorithm and the classification system used in the experiments.

2.2.1 Naive Bayes

The Bayes learning algorithm is considered by researchers to be among the most practical approaches to certain types of learning problems. Michie at al [MST94] provide a detailed study comparing the Naive Bayes classifier to several other learning algorithms, such as decision tree and the neural network. Naive Bayes classifier is competitive with those other learning algorithms and in some cases outperforms them. The Naive Bayes classifiers has been successfully used for text classification and therefore it was one of the classifiers considered for the experiments described in this thesis.

2.2.1.1 Naïve Bayes and Text Classification

The main idea in the Naïve Bayes approach to text classification is to use the joint probabilities of words and categories to estimate the probabilities of categories given a new document.

Bayes Theorem in a general form:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

In terms of text classification, what we want to determine is the probability of assigning a class label given the training data P(Class|Data).

The Bayes Theorem is used for text classification in the following way. Given the set of categories $C=\{c_1, c_2, \ldots, c_n\}$ and a document consisting of d words $(w_1, w_2, \ldots, w_d)$ a document can be assigned to a single class c^* which a posteriori probability is the highest:

$$\begin{aligned} c^* &= \arg\max_{c_j \in C} P(c_j \mid w_1, w_2, \ldots, w_d) \\ &= \arg\max_{c_j \in C} \frac{P(w_1, w_2, \ldots, w_d \mid c_j)P(c_j)}{P(w_1, w_2, \ldots, w_d)} \\ &= \arg\max_{c_j \in C} P(w_1, w_2, \ldots, w_d \mid c_j)P(c_j) \end{aligned}$$

The denominator was dropped because it is a constant within all classes.

The Naïve Bayes classifier is based on the assumption of conditional independence, that is that the occurrence of a word in a document given the class c_j is independent of the

occurrence of any other word in this document given the same class. Given this assumption $P(w_1, w_2, ..., w_d)$ becomes:

$$P(w_1, w_2, ..., w_d \mid c_j) = \prod_{i=1}^{d} P(w_i \mid c_j)$$

Therefore, given the independence assumption, the classifier becomes:

$$c^* = \arg\max_{c_j \in C} P(c_j) \prod_{i=1}^{d} P(w_i \mid c_j)$$

The assumption of conditional independence is incorrect, since certain words often form common phrases or are otherwise used together. For example the probability of seeing the word "am" in some position may be greater if the preceding word is "I". Despite the inaccuracy of this assumption the Naive Bayes method performs remarkably well in document classification.

The conditional probability $P(w_j|c_j)$ is estimated by using Laplace smoothing that adds one to all the word counts to avoid probabilities of zero. Therefore, the estimate of a probability of a word w_I in class c_j is:

$$P(w_i \mid c_j) = \frac{1 + N(w_i, c_j)}{T + \sum_{w \in T} N(w, c_j)}$$

Where N(w, c) is the number of times the word w_I occurs in the training documents for class c_j, T is the total number of unique words within the set of the training documents. Given a set of training documents $D = \{d_1, d_2, \ldots, d_M\}$ the prior probability $P(c_j)$ is

$$P(c_j) = \frac{N(d_i, c_j)}{M}$$

Where M is the total number of training documents and $N(d_I, c_j)$ is the number of training documents d_i assigned to class c_j.

2.2.1.2 Rainbow

Rainbow is a program that performs statistical document classification. It is based on the Bow [http://www-2.cs.cmu.edu/~mccallum/bow/] C-code library developed by Andrew McCallum at Carnegie Mellon University. It is mostly designed to employ the Naïve Bayes algorithm, but it also provides TFIDF/Rocchio, Probabilistic Indexing and K-nearest neighbor. In our experiments only the Naïve Bayes algorithm was used.

Rainbow accepts data in plain text files, one file per document. No special tags are needed at the beginning or end of documents. Files are organized in directories such that each directory contains documents with the same class label.

Since Rainbow accepts plain text as data, no special formatting of the corpora was required. The files (e-mails in case of 20Newsgroups and Yahoo Newsgroups, as well as songs of DigiTrad) were already organized in directories signifying appropriate class.

Sample output given by Rainbow:

```
Correct: 151 out of 160 (94.38 percent accuracy) -

Confusion details, row is actual, column is predicted
classname    0      1      : total
0 ml         81     6      : 87   93.10%
1 pb         3      70     : 73   95.89%
```

There are two classnames: 0 – ml (machine learning) and 1 – pb (perl beginner).

81 out of 87 examples of ml class have been correctly classified, giving a 93.1% accuracy. 70 out of 73 examples of pb class have been correctly classified, yielding 95.89% accuracy. The overall accuracy is 94.38% (correctly classified 151 examples out of 160).

2.2.2 Decision Rules

A classifier for category c_i built by an inductive rule learning method consists of a rule in a disjunctive normal form (DNF). The literals in the premise denote the presence or absence (negated keyword) of the keyword in document d_j, while the head of the rule denotes a decision to classify the document d_j to the class c_i. DNF rules are similar to decision trees in that they can encode any Boolean function. However, they tend to produce more compact classifiers than decision trees with rules that are easily readable for humans.

DNF rule learners vary in terms of the methods, heuristics and criteria employed in generalization, therefore what follows is the more detailed description of the algorithm used in Ripper, the system that was used in the experiments.

2.2.2.1 Decision Rules in Ripper

RIPPER, a rule-based machine learning algorithm, was developed by William Cohen [COH95] and is based on repeated application of Furnkanz and Widmer's [FUW94] IREP algorithm. RIPPER stands for Repeated Incremental Pruning to Produce Error Reduction, while IREP stands for Incremental Reduced Error Pruning. The Ripper algorithm offers a significant performance over previous rule induction algorithms. Compared to C4.5 rules [COH96] Ripper produced rules of comparable or better accuracy on 22 out of 37 benchmark data.

Ripper has been applied to several problems in text classification with good results [COH96] and has made its mark in a field dominated by almost purely statistical methods.

In Ripper a decision rule is defined as a sequence of Boolean clauses linked by logical AND operators indicating belonging to a particular class. The clauses can either be in the

form $A \geq x$ or $A \leq x$ for continuous (numerical) attributes, or $A=x$ or $A \neq x$ for nominal (descriptive values taken from a predefined set) attributes. A classification hypothesis is a sequence of rules ending in a default rule with an empty set of clauses. During classification the left-hand sides of the rule are applied until one of them evaluates to true and the implied classification label from the right hand side is the predicted class for the examined instance.

In order to explain the algorithm used by Ripper, a case in which examples fall into one of the two classes: positive or negative is considered.

The original IREP algorithm forms rules by repeated growing and pruning. IREP first splits the training set into the growing and pruning set. Rules to predict a positive class are grown one at a time by starting with an empty rule and then adding clauses to the left-hand side according to a grow heuristic. This way rules are made more and more restrictive in order to fit the data as closely as possible. Growing of a rule stops when the rule covers no negative examples from the growing set. The pruning stage is required in order to avoid overfitting, which may cause a poor performance on unseen data. Each rules is pruned immediately after it is grown by deleting clauses covering too many negative clauses in the pruning set, according to the pruning heuristic.

After a new rule is grown and pruned, the used examples are removed from both sets. Then the remaining data is repartitioned and the process repeated until all the examples in the training set are covered or some other stopping condition reached.

The grow heuristic used in Ripper is the information gain function proposed by J.R.Quinlan [QUI90].

After the rule is grown, it overfits data if it covers negative examples from the pruning set. The ideal is for the rule to cover many positive examples and no negative examples. The pruning heuristic measures this coverage by maximizing the function:

$$v = \frac{p-n}{p+n}$$

where p is the number of positive examples, n number of negative examples in the pruning set covered by the rule. v is maximized if n=0.

The stopping condition in Ripper is based on the Minimum Description Length (MDL) principle. The formula balances accuracy against complexity based on the number of bits required to communicate the complete and correct classification for a set of examples. It is obtained by adding a number of bits required to describe the classification hypothesis to the number of bits required to enumerate the exception to the hypothesis. Ripper stops adding rules if the new description length is over 64 bits larger than the best description length so far.

The final step in the algorithm is the rule optimization. Each rule is considered in turn and two new potential rules are grown. The first one starts with an empty rule, the second one is grown starting with the original rule instead of an empty one. Both rules are grown and pruned in order to optimize the error rate of the entire rule set on the entire pruning set. The decision of which of the three candidate rules to choose is based on the MDL heuristic.

Ripper repeats the entire algorithm. The number of iterations can be changed by a user. The default is set to 2 and this is the value always used in all the experiments.

2.2.2.2 Ripper and Text Classification

Ripper has the advantage of allowing set-valued features in which a set of atomic values can be defined as a single feature. In case of text classification a single value named WORDS is defined and the set of all words that appear in each example is used as feature values. Ripper considers the entire training set and assembles a binary feature vector where each word from the entire training corpus is represented by a single value. The value is 0 if the word appears in the given document and 0 otherwise.

The input data for Ripper is one text file, where each document is a single instance. Each attribute (word) in an instance has to be in single quotes, and separated from other attributes by a single space. The class tag is the last attribute in an instance, separated from attributes by a comma, with a delimiting period. The class tag is not surrounded by single quotes.

```
'Program' '1' 'is' 'wrong' 'You' 'do' 'not' 'open' 'a' 'file' 'in' 'one' 'sub' 'then' 'close' 'the' 'file'
'handle' 'in' 'another', pbtag.
```

Fig. 6 Example of input data in the format acceptable by Ripper. The last attribute, *pbtag*, is he class label. *Pbtag* in this case refers to *perl_beginner*, so this particular instance comes from the *perl_beginner* group in Yahoo Groups.

The output is ready to read and comprehend:

```
Hypothesis:
mltag :- WORDS ~ like, WORDS ~ wrote (21/1).
mltag :- WORDS ~ perl (15/0).
mltag :- WORDS ~ write (5/0).
default othertag (60/17).
Error rate on holdout data is 16.9492%
Running average of error rate is 16.9492%
```

This data set consisted of examples belonging to two classes, labeled mltag and othertag.

In 21 out of 22 cases in the training set words *like AND wrote* imply mltag.

In 15 cases in the training set the word *perl* implies mltag.

In 5 cases in the training set the word *write* implies mltag.

The default is othertag, which held in 60 out of 77 cases in the training set.

2.2.3 Decision Trees

Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree. Learned trees can also be re-represented as a set of if-then rules which are easier for a human to comprehend. These learning methods are among the most popular inductive inference algorithms and have been successfully applied to various classification tasks.

Each node in the decision tree corresponds to a test of some attribute and each branch descending from that node represents one of the possible values for this attribute. An instance is classified by starting at the root node and testing the attribute specified by this node, then moving down the branch corresponding to the value of the attribute in the given instance. The process is then repeated for each subtree rooted at the reached node.

The most important question when constructing a decision tree is which attribute should be placed at the root of the tree. In order to answer this question each attribute is tested to determine how well it alone would classify the training examples. The best attribute is selected and placed at the root of the tree. Then it is removed from the set of all attributes and the remaining attributes are tested in the same manner, always choosing the best one to place at the root of subsequent subtrees.

C5.0 – the system used in the experiments described in this thesis, just like its predecessor C4.5 [QUI93] uses the statistical property called *information gain* to determine the choice of attributes for root nodes. This property measures how well a given attribute separates the training examples according to their target classifications.

In order to measure information gain, first entropy characterizing the impurity of an arbitrary collection of examples must be calculated.

$$\text{Entropy}(S) \equiv -p_+ \log_2 p_+ - p_- \log_2 p_-$$

where S is a set containing positive and negative examples, p_+ is the proportion of positive examples in S and p_- is the proportion of negative examples in S. In all calculations involving entropy 0 log 0 is defined to be 0.

The entropy is 0 if all members of S belong to the same class. The entropy is 1 if the collection contains an equal number of positive and negative examples. If the number of positive and negative is unequal, the entropy is between 0 and 1.

The above formula is restricted to the special case when the target classification is boolean. The general form of the formula is

$$Entropy(S) \equiv \sum_{i=1}^{c} - p_i \log_2 p_i$$

where p_i is the proportion of S belonging to class i.

Information gain is the expected reduction in entropy caused by partitioning the examples according to a particular attribute. The information gain Gain(S, A) of an attribute A relative to the set S is defined as

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

where Values(A) is the set of all possible values for the attribute A, and S_v is the subset of S for which the attribute A has value v.

Gain(S, A) gives the information about the target function value, given the value of some attribute A. Attribute with the highest information gain is selected to be the root of a tree. The process is then repeated for the remaining attributes which are consecutively selected as roots of subsequent subtrees. The process continues until either of the two conditions is met: 1) there are no more attributes or 2) the training examples associated with a particular leaf node all have the same class value.

One of the issues in decision trees learning is that of how deeply to grow the tree. Smaller trees consistent with the examples are preferred to larger trees consistent with the examples. Given noise in the data or data where the number of training examples is too small to produce a representative sample of the true target function, growing trees too deeply may lead to producing trees that overfit training examples.

There are several approaches to avoid overfitting data, which can be grouped into two categories:

- stop growing the tree earlier before it reaches the point where it perfectly classifies the training data
- allow the tree to overfit the data and then post-prune the tree.

Since it is difficult to estimate exactly when the tree should stop being grown, the second approach of post-pruning is considered to be more successful.

C5.0 uses a variation of the second approach, which is rule post-pruning. In this approach the tree is grown until it fits the training data well and overfitting is allowed. Then the tree is converted to an equivalent set of rules by creating one rule for each path in the tree. Each rule is then pruned (generalized) by removing any preconditions which removal does not worsen its estimated accuracy. The pruned rules are sorted by their estimated accuracy and considered in sequence when classifying new instances.

2.2.3.1 C5.0

C5.0 (www.rulequest.com) is an improved version of the classification decision trees induction algorithm C4.5 [QUI93]. It has been widely used for categorization problems, including those of text categorization, and as such, has also been chosen for our experiments.

C5.0 requires at least two files as its input. The first one is the **names** file which describes attributes and classes. The name of each explicitly-defined attribute is followed by a colon and a description of the values taken by the attribute. There are three different acceptable types of values: continuous (numerical value), date and time. The first line in the names file lists the classes. The second essential file, the application's **data** file provides information on the training cases from which C5.0 will extract patterns. The entry for each case consists of one or more lines that give the values for all explicitly-defined attributes. The attribute values are followed by the case's class value. Values are separated by commas and the entry is terminated by a period.

A sample output of C5.0

```
Read 178 cases (2933 attributes) from ml_pb/ml_c5.data
Decision tree:

perl > 0:
:...craft <= 0: othertag (39)
:   craft > 0:
:   :...learn <= 0: othertag (5)
:       learn > 0: mltag (2)
perl <= 0:
:...north > 0: othertag (7)
    north <= 0:
    :...file > 0:
        :...craft <= 0: othertag (8)
        :   craft > 0: mltag (4)
        file <= 0:
        :...kei > 0: othertag (6/1)
kei <= 0:
          :...talk <= 0: mltag (84/7)
              talk > 0:
              :...data <= 2: othertag (3)
                  data > 2: mltag (2)

Evaluation on hold-out data (18 cases):

        Decision Tree
      ----------------
      Size      Errors

        10    0( 0.0%)   <<
```

Output is represented as a decision tree. Values of the nodes are words appearing in the text – perl, craft, learn, etc. and the test measures how many times they appear in the document. Leaves decide on the class label the given document assumes.

In the given example perl is at the top of the decision tree. The first leaf is craft

craft <=0 – if the word craft does is not found in the document, the document belongs the "othertag" class. That was the case on the 39 of the tested examples

craft<=0: othertag(39)

2.2.4 SVM

Let us remind the formal definition of classification: it is to estimate an unknown function $f: X \rightarrow \{\pm 1\}$ given training data – N-dimensional vectors x_i and class labels y_i:

$(x_1, y_1),\ldots,(x_l, y_l) \in X \times \{\pm 1\}$

such that f will predict a class value $y \in \{\pm 1\}$ for a previously unseen example $x \in X$. It will be possible if the new example x is in some way similar to the training examples. A type of similarity of mathematical interest is a dot product. Given two vectors, $x, x' \in R^N$, the canonical dot product is defined

$$(x \circ x') := \sum_{i=1}^{N} x_i x'_i$$

The geometrical interpretation of the dot product is that it computes a cosine of the angle between x and x', provided their length is normalized to 1. It also allows computation of the length and the distance between the two vectors.

SVM classifiers are based on the class of hyperplanes:

$$(w \circ x) + b = 0, w \in R^N, b \in R$$

corresponding to decision functions:

$$y = sign((w \circ x) + b)$$

where w is the *weight vector*, b is the *threshold*.

There exists a unique hyperplane yielding the maximum margin of separation between two planes. To construct this optimal hyperplane a constrained dual quadratic optimization problem must be solved. The solution w has an expansion

$$w = \sum_i v_i x_i$$

in terms of a subset of training patterns that lie on the margin. These patterns are called support vectors and carry all information relevant to the classification.

The final decision function (details of calculations omitted) is:

$$y = sign(\sum_i v_i (x \circ x_i) + b)$$

and it depends only on the dot product between the vectors.

This is a simple linear case, but helps generalize to non-linear cases.

The basic idea of SVM is to map the data into some other dot product space F (feature space) via a non-linear map:

$$\Phi : X \rightarrow F$$
$$x \rightarrow \Phi(x)$$

and perform the linear algorithm in F. Computing a dot product in F, given it is usually of very high dimension, may be very expensive. Therefore, a kernel function is used, such that it can be evaluated efficiently.

$$k(x, x') = (x \circ x') = (\Phi(x) \circ \Phi(x'))$$

Since not all problems are linearly separable, more complex kernels can be used:

polynomial
$$k(x, x') = ((x \circ x') + \theta)^d$$

radial basis function (RBF)
$$k(x, x') = \exp(\frac{-\| x - x' \|^2}{c})$$

sigmoid
$$k(x, x') = \tanh(\kappa(x \circ x') + \theta)$$

Using different kernels produces different classifiers in the input space: polynomial, gaussian (RBF), 3-layer neural nets, and other.

SVM can construct complex models (neural nets, radial basis function nets, polynomial and other) yet it is simple to analyze mathematically. It corresponds to a linear method in a high dimensional feature space nonlinearly related to the input space. Even though it can be thought of as a linear algorithm in high-dimensional space, by the use of kernels all necessary computations are performed directly in the input space.

2.2.4.1 LibSVM

LIBSVM - A Library for Support Vector Machines [CHL01] is an integrated software for support vector classification, (C-SVC, nu-SVC), regression (epsilon-SVR, nu-SVR) and distribution estimation (one-class SVM).

A C- or ν-Support Vector Classifiers take into consideration the fact that a separating hyperplane may not exist, that is, a high noise level may cause a large overlap between the classes. $C>0$ is the penalty parameter of the error term – it sets an upper bound on the number of training errors. $\nu \in (0,1]$ is an upper bound on the fraction of training errors and lower bound on the fraction of support vectors.

In the experiments the C-SVC was used. Classification was performed using all four available kernels- linear, polynomial of degree 3, RBF and sigmoid. In all cases linear kernel produced the best results of very high accuracy (over 90%). In the Section 3.3 Results and Discussion, we report only the results of LibSVM with a linear kernel.

Sample output of LibSVM:

```
optimization finished, #iter = 354
nu = 0.011736
obj = -0.944739, rho = 0.192434
nSV = 74, nBSV = 0
Total nSV = 74
Accuracy = 100% (17/17)
```

Accuracy is of the main interest to us. The remaining portion of the result reports values of various parameters. Total nSV is the total number of support vectors which is the summation of nSV (number of support vectors) and nBSV (number of bound support vectors). Obj is the objective value of the dual SVM problem. Rho is the bias term in the decision function. Nu is the value of either C or ν parameter, depending on which classifier was chosen.

3. Disambiguation

The objective of this work was to measure the influence of word sense disambiguation (WSD) on text classification, therefore all three corpora had to be first disambiguated.

Word sense disambiguation (WSD) has received a lot of interest from natural language processing researchers in the recent years. There are now quite a few working WSD systems employing various approaches to solving the problem. It was difficult, however, to choose the best system at the time when the work on this thesis began, given that the process of their evaluation had not been yet standardized [YAR95], [KIL97]. In 1998 [KIL98] WSD researchers launched a SENSEVAL [http://www.senseval.org/] project meant to evaluate various WSD systems. The most recent results (Senseval 2002) show systems which precision and recall reaches about 70%.

The accuracy of word sense disambiguation is of a detrimental value for this work. If disambiguation is not done correctly, it will influence the results of text categorization. Therefore, given the lack of reliable automatic or semi-automatic WSD tools at the time when this work began, the task was carried out by hand. The undertaking of manual word sense disambiguation is a tedious and time-consuming job, therefore, only partial disambiguation focusing exclusively on **nouns** was performed.

3.1 Dictionary

Choosing an appropriate meaning of a word is not always an obvious decision. One of the approaches to automatic word sense disambiguation is to use an online dictionary, or thesaurus.

WordNet was used as an aid during the process of disambiguation, but not all word senses available in this database were taken into consideration. Many of WordNet senses are too fine-grained to be easily distinguished. Some of the senses are missing, some other are very uncommon. Therefore, in process of disambiguation, a new dictionary was created, which consisted only of the nouns that appeared in the tested corpora and their most distinct senses. This dictionary consisted of 388 words and 1177 distinct senses, but even those meanings were not all used in the corpora.

Each of the different meanings of a particular noun in the dictionary has a numerical tag attached to it to make it a new distinct word. For example the word *account* appearing in the original data set would be converted to either *account1* - record, history, *account2* – monetary found, *account3* – reason, ground, and so on.

The word "account" in WordNet, returns 12 distinct meanings:

```
Synonyms/Hypernyms (Ordered by Frequency) of noun account
12 senses of account

Sense 1
history, account, chronicle, story
    => record
Sense 2
report, news report, story, account, write up
    => news
Sense 3
explanation, account
    => statement
Sense 4
account, business relationship
    => relationship
Sense 5
account, accounting, account statement
    => statement, financial statement
Sense 6
```

score, account
    => reason, ground
Sense 7
report, account
    => informing, making known
Sense 8
account
    => commercial enterprise, business enterprise, business
Sense 9
account, bank account
    => fund, monetary fund
Sense 10
account, brokerage account
    => fund, monetary fund
Sense 11
bill, account, invoice
    => statement, financial statement
Sense 12
account
    => profit, gain

Fig. 7 Synset of the word "account" given by WordNet.

In the above example, Fig.7 we can see that some of the meanings returned by WordNet repeat (Sense 9 and Sense 10) or seem fairly close (Sense 1 and Sense 2). Therefore in the new dictionary, *account* has only 6 entries. The choice of the senses is purely subjective. In this case, a new meaning, not present in WordNet, was also added: *an e-mail account*. *Account* appeared in this particular sense most often in the investigated corpora (Fig. 8).

account1 - record, history
account2 - monetary fund
account3 - reason, ground
account4 - report, informing, making known
account5 - email account
account6 - importance, significance

Fig. 8 Entry of the word "account" in the dictionary used for the manual disambigutaion.

3.2 Manual Disambiguation

Process of a manual word senses disambiguation could be divided into 4 subtasks:

(1) finding a word in text
(2) finding out its possible meanings
(3) choosing a proper meaning
(4) replacing the noun with the same noun accompanied by an appropriate tag

One of the ways to manually disambiguate text documents would be to open and read each document, and mark each found ambiguous word with an appropriate tag. This way, however, would be very tiresome and time-consuming, even if only nouns, as in this case, were taken into account. One can take advantage of the fact that often a popular ambiguous noun appears many times across the entire corpus used in one particular sense. Therefore, a faster way of dealing with the problem would be to find out all positions of this particular noun and mark all occurrences of the same sense automatically.

For the subtask (1), Brill's Part of Speech (POS) Tagger was used to find out where all nouns are in the text.

The/DT first/JJ **premise/NN** and/CC the/DT **conclusion/NN** are/VBP not/RB properly/RB translated/VBN as/IN **identity/NN** statements/NN since/IN the/DT is/NN in/IN those/DT statements/NNS "is"/VBZ the/DT "is"/NN of/IN predication/NN rather/RB than/IN of/IN identity/NNP

Fig. 9 Example of a tagged text. Source: 20Newsgroups, alt.atheism, file 51222.

Nouns are marked with tags NN, NNS and NNP.

Subtask (2) was to find out their possible meanings. A script took each word marked by Brill's POS Tagger as a noun, and extracted its WordNet synset. If there were more than one possible meaning, the word along with its WordNet meanings was recorded in a file.

If the word had only one meaning according to WordNet, it was considered as non-ambiguous and ignored.

In the above example Fig.9 the word *premise* returned only one meaning, therefore it was not considered to be an ambiguous word. Words *conclusion* and *identity* had more than one meanings according to WordNet, therefore their meanings were recorded in a file:

conclusion: decision, determination, conclusion
judgment, judgement, mind
conclusion: conclusion
deduction, entailment, implication
conclusion: finale, finis, finish, last, terminu
point, point in time
conclusion: ending, conclusion
happening, occurrence, natural event
conclusion: conclusion
proposition
conclusion: termination, ending, conclusion
change of state
conclusion: conclusion, close, closing, ending
section, subdivision
conclusion: decision, determination, conclusion
choice, selection

identity: identity, individuality
personality
identity: identity
recognition, identification
identity: identity, identity element, identity operator
operator
identity: identity, identicalness, indistinguishability
sameness

Fig. 10 List of possible meanings of words "conclusion" and "identity" extracted from Wordnet.

Subtask (3) - choosing a proper meaning for an ambiguous word - was the task that required human intervention. In each data set parts of sentences containing ambiguous nouns were extracted, and then grouped by nouns in question. This way it was possible to see each ambiguous noun in all sentences within the data set. If one sentence had more than one ambiguous noun, say *ambig_noun1* and a different *ambig_noun*2, it was

extracted twice and appeared along with other sentences containing *ambig_noun1* and then with other sentences containing *ambig_noun2*.

The example below (Fig.11) shows all sentences containing nouns *conclusion* and *identity* that appeared in the data set *alt.atheism* belonging to the *20 Newsgroups* corpus.

-- **conclusion**
51153: To actually draw a **conclusion** about the " existence "
51211: you have evidence against this **conclusion** not to present one's own **conclusions** prior to this
51222: The first premise and the **conclusion** are not properly translated as identity statements

-- **identity**
51222: The first premise and the conclusion are not properly translated as **identity** statements
51222: " is " the " is " of predication rather than of **identity**
51223: " is " 's of **identity** both syllogisms are valid

Fig. 11 All sentences from the data set alt.atheism belonging to the 20Newsgroups corpus containing words "conclusion" and "identity". Numbers on the left indicate the filename the sentence was found in.

The majority of times it was possible to determine the sense of the word just by inspecting the sentence it appeared in. In cases where one sentence was not enough to decide on the meaning of the noun, the entire file was consulted (and even then, sometimes the decision was not obvious).

Subtask (4) – attaching an appropriate numerical tag to the noun, was automated by a script which needed the name (or names) of the file (or files), noun in question and the appropriate tag as an input. The script changed all occurrences of the specified noun in the specified file (files) to the new noun with the given tag. However, if one ambiguous noun appeared in different meanings in one text file, that task was completed manually.

Spelling mistakes were ignored, so only the properly typed words were changed where appropriate.

During the process of manual disambiguation it was possible to see that some words were used in different meanings across all the data sets within one corpus. The example below shows the word *bit* used in three different senses in classes *The_Unhandled_Horse, Distantsuns* and *ChildFreeUK*, belonging to *the Yahoo Discussion Groups* corpus.

The_Unhandled_Horse

265 :: try a rubber **bit** in her mouth
273 :: maybe you should change his **bit**
275 :: always mount from something a **bit** higher off the ground to take

Distantsuns

278 :: 24 **bit** rendering on the Jupiter
282 :: card is really a 24 **bit** card

ChildfreeUK

209 :: I was a **bit** of a loon for thinking it
212 :: to it although the copyrighted **bit** might prove a problem
217 :: largely selfish (especially the **bit** " well nobody forced you to
227 :: I've done quite a **bit** of that before

Fig. 12 An example of an ambiguous word "bit" appearing in three of the Yahoo groups – The_Unhandled_Horse, Distantsuns and ChildfreeUK. The word "bit" was recognized to have three distinct meanings: bit1 - small quantity, piece, bit2 - computer memory unit, bit3 - stable gear, tack.

The above example Fig.12 shows that disambiguation of the documents in chosen corpora would help differentiate between various senses and therefore might aid the classification task

4. Classification Experiments

Two different types of classification experiments were executed. The initial experiments were performed on perfectly balanced or mildly imbalanced data sets. The second set of experiments was performed on strongly imbalanced data sets in order to see whether disambiguation has a stronger or lesser effect on such data. Results of both types of experiments were verified using different techniques, description of which follows in Sections 4.1 Initial Classification Experiments and 4.2 Imbalanced Data Sets Experiments.

4.1 Initial Classification Experiments

All initial classification experiments were run using the 10 fold cross validation technique: the training set was divided randomly into 10 parts, out of which one part was used for testing and remaining 9 parts for training.

Each of my corpora had more than two data sets, however, most of the classifiers used in the experiments can handle only 2 classes classification. Therefore two types of experiments were performed.

(1) One against One

Only two data sets from a given corpus were classified at a time. Each data set was classified against all the other ones. 20Newsgroups with 5 data sets produced 10 possible pairings, Yahoo Groups with 8 data sets 28, and the Digitrad corpus with 6 different data sets 15 pairings.

(2) One against Rest

One class from a corpus was classified against all remaining classes treated as one. This resulted in an imbalanced data set. Worst results were expected, however that was not always the case.

What follows is a presentation of classification results and the discussion.

4.1.1 Results and Discussion

Note. In all tables of results short names of data sets were used. See Appendix I.

Each classifier outputs results either in the form of average accuracy or average error. In case an answer was an average accuracy it was converted to an average error, so all results are represented in the form of average errors.

Average Error on Original Data is referred to as **AEOD**.
Average Error on Disambiguated Data is called **AEDD**.

This section is divided into two parts – **One against One** and **One against Rest.** The results are reported in various forms. In the first section only AEOD and AEDD considered. The second approach takes into consideration standard deviation calculated from intermediate results of each of the folds (the method of classification is 10 fold cross validation). The third section we present results obtained from performing a statistical significance test T-Test, also based on intermediate results of 10 fold cross validation.

4.1.1.1 One against One

In each corpus each data set was classified against each of the other data sets in the "one against one" manner.

4.1.1.1.1 AEOD vs AEDD

First, we look at average errors on original data (AEOD) and disambiguated data (AEDD). For simplicity we report only the difference AEOD – AEDD.

If AEOD – AEDD > 0 we infer disambiguation helped in case of the data set
If AEOD – AEDD < 0 we infer disambiguation made classification worse.

Results are sorted by the difference, from greatest (best improvement on disambiguated data) to smallest, for each classifier separately. This format is chosen in order to see whether any particular data set benefited from disambiguation more often than other.

20 Newsgroups:

Classifier	Data Set	AEOD-AEDD
C5	ibm_med/	4
	ath_comp/	3.5
	ibm_mac/	1.9
	comp_med/	0
	comp_ibm/	0
	comp_mac/	-0.8
	ath_mac/	-0.8
	ath_med/	-1.5
	mac_med/	-2.9
	ath_ibm/	-5.6
Ripper	ibm_med/	14
	comp_med/	7
	comp_mac/	5.6
	ibm_mac/	5.6
	ath_comp/	3.7
	comp_ibm/	3
	ath_mac/	2.4
	ath_ibm/	1.3
	ath_med/	0
	mac_med/	-1

Classifier	Data Set	AEOD-AEDD
SVM	comp_ibm/	14
	ath_med/	3.08
	ibm_mac/	2.97
	ath_mac/	2.55
	mac_med/	2.5
	comp_mac/	0.84
	ath_comp/	0.52
	ath_ibm/	0
	comp_med/	0
	ibm_med/	-2
Rainbow	ibm_mac/	3.6
	ibm_med/	1.71
	mac_med/	1.43
	comp_mac/	1.3
	comp_ibm/	1.2
	ath_ibm/	0.11
	ath_comp/	-0.4
	ath_mac/	-0.4
	comp_med/	-0.7
	ath_med/	-0.9

Table 1 One-One. 20Newsgroups, AEOD-AEDD

In the 20Newsgroups corpus we assumed that *comp, mac* and *ibm* are similar data sets, while *ath* and med are different from all other data sets. Intuitively, classification of similar data sets should benefit from disambiguation more than classification of data sets that are noticeably different in their original version.

The 3 combinations of *comp, mac* and *ibm* (*ibm_mac*, *comp_mac* and *comp_ibm*) are at the top of the list for the Rainbow classifier. The improvement is very small (later showed to be insignificant if standard deviation is taken into account or a t-test performed), but it is the biggest improvement out of all data sets in this classifier. These

data sets also show improvement when classified in Ripper and SVM. C5.0 shows no improvement or slight loss on these data sets.

Ripper and SVM benefited from disambiguation of the 20Newsgroups corpus 8/10 and 7/10 times respectively. Performance of C5.0 suffered from disambiguation 7/10 times, while Rainbow benefited 6/10 times.

Yahoo Groups:

Classifier	Data Set	AEOD-AEDD
C5	avc_ds/	8
	cl_pb/	4.2
	avc_cfuk/	3.1
	dp_ml/	2.5
	avc_dp/	2
	cl_dp/	1.6
	ds_pb/	1.4
	cfuk_dp/	1.2
	cfuk_ds/	1.2
	cfuk_pb/	1.1
	dp_tuh/	1.1
	ml_tuh/	1
	cl_ml/	0.9
	dp_ds/	0.9
	ds_tuh/	0.4
	dp_pb/	0.1
	avc_cl/	0
	ds_ml/	0
	ml_pb/	0
	cl_tuh/	-0.1
	cfuk_tuh/	-0.5
	avc_tuh/	-1
	avc_pb/	-1.1
	avc_ml/	-1.5
	cfuk_cl/	-2.5
	pb_tuh/	-2.8
	cfuk_ml/	-3.7
	cl_ds/	-3.8

Classifier	Data Set	AEOD-AEDD
SVM	ds_tuh:	3.98
	cfuk_ds:	2.08
	ds_pb:	1.61
	cl_tuh:	1
	cfuk_dp:	0.98
	cl_dp:	0.92
	pb_tuh:	0.5
	avc_ml:	0.44
	dp_ds:	0.43
	cfuk_pb:	0.03
	avc_cl:	0
	avc_dp:	0
	avc_tuh:	0
	cfuk_ml:	0
	cl_pb:	-0.03
	dp_ml:	-0.03
	cl_ml:	-0.05
	dp_tuh:	-0.42
	cfuk_cl:	-0.48
	cl_ds:	-0.48
	avc_cfuk:	-0.5
	dp_pb:	-0.53
	avc_pb:	-0.59
	cfuk_tuh:	-1.5
	ml_tuh:	-1.53
	ds_ml:	-1.58
	ml_pb:	-2.23
	avc_ds:	-6

Ripper	avc_cl/	5
	avc_ds/	5
	cfuk_ml/	4.74
	cfuk_tuh/	4.71
	ds_pb/	4.53
	avc_cfuk/	4.21
	avc_pb/	3.3
	dp_tuh/	3.14
	cl_ds/	3.05
	cl_tuh/	2.95
	avc_tuh/	2.5
	ml_pb/	2.36
	ds_tuh/	2
	dp_ml/	1.97
	ml_tuh/	1.96
	dp_pb/	1.42
	cl_dp/	1.32
	avc_ml/	0.92
	cfuk_pb/	0.59
	cfuk_dp/	0.33
	ds_ml/	0.28
	cl_ml/	0
	dp_ds/	-0.05
	avc_dp/	-0.5
	cfuk_cl/	-0.83
	cfuk_ds/	-1.77
	cl_pb/	-3.33
	pb_tuh/	-4.61

Rainbow	dp_ds/	1.79
	pb_tuh/	1.23
	Cfuk_cl/	1.2
	avc_ml/	0.75
	cl_ml/	0.75
	ml_tuh/	0.6
	dp_tuh/	0.47
	avc_cfuk/	0.38
	Cfuk_tuh/	0.36
	cl_tuh/	0.28
	avc_pb/	0.11
	ds_ml/	0.11
	cl_pb/	0.01
	cl_dp/	0
	dp_pb/	0
	avc_tuh/	-0.1
	cl_ds/	-0.1
	avc_cl/	-0.2
	Cfuk_dp/	-0.3
	ml_pb/	-0.3
	ds_pb/	-0.34
	avc_dp/	-0.4
	dp_ml/	-0.5
	ds_tuh/	-0.5
	Cfuk_ds/	-0.67
	Cfuk_ml/	-1
	avc_ds/	-2.72
	Cfuk_pb/	-2.9

Table 2 One-One. Yahoo Groups, AEOD-AEDD

Ripper again benefited from disambiguation most often. In this data set C5.0 also shows improvement more often than not (it suffered from disambiguation of previous two corpora). Rainbow and SVM show improvement about half the time. There are several data sets that showed improvement ¾ times. Their topics are usually unrelated, so nothing about similarity or the lack of it affecting classification can be inferred.

Digitrad:

Classifier	Data Set	AEOD-AEDD
C5	Scots_marriage/	16
	Scots_political/	11
	death_political/	9
	Irish_death/	7.5
	Irish_political/	3.1
	death_marriage/	2
	love_political/	1.5
	Irish_Scots/	1.1
	Scots_death/	0
	political_marriage/	0
	Irish_love/	-0.4
	love_marriage/	-0.4
	Scots_love/	-0.6
	death_love/	-2.4
	Irish_marriage/	-5.9
Ripper	Irish_political/	15.1
	Irish_death/	8.3
	Scots_love/	6.7
	Irish_marriage/	5.6
	Scots_marriage/	5.5
	love_political/	5.1
	death_political/	4.5
	political_marriage/	2.5
	Irish_Scots/	2.1
	Irish_love/	2
	Scots_death/	1.5
	Scots_political/	0.5
	death_marriage/	0.5
	death_love/	-0.6
	love_marriage/	-2.2

Classifier	Data Set	AEOD-AEDD
SVM	Irish_marriage/	13.66
	death_political/	10
	Scots_political/	7.46
	Irish_love/	3.03
	love_political/	-0.05
	love_marriage/	-0.98
	Irish_Scots/	-1
	Irish_death/	-1
	Scots_love/	-2.16
	Scots_death/	-4.21
	political_marriage/	-8
	death_marriage/	-12
	death_love/	-20.13
	Scots_marriage/	-20.33
	Irish_political/	
Rainbow	Irish_marriage/	3.7
	love_political/	3.4
	death_marriage/	1.4
	Scots_political/	1.29
	Scots_love/	0.6
	Irish_death/	0.3
	love_marriage/	0.2
	Scots_death/	-0.2
	death_political/	-0.4
	Irish_political/	-0.6
	Irish_love/	-0.7
	Irish_Scots/	-1
	Scots_marriage/	-1
	death_love/	-2
	political_marriage/	-2.4

Table 3 One-One. Digitrad, AEOD-AEDD

It is a bit more difficult to notice any consistency in this set. Again Ripper benefited from disambiguation more than other classifiers – 13/15 times. SVM however suffered from disambiguation of this data set more often than benefited. Rainbow and C5.0 show a similar tendency of results split more or less in half.

Data sets that benefited from disambiguation most often are *Scots_political*, *Irish_marriage* and *death_political*. It is difficult to discuss the similarity between data sets in each pair.

Data sets that suffered from disambiguation are *death_love*, *love_marriage* and *political_marriage*. The other 9 data sets show improvement and decline 2/4 times across all classifiers. It is difficult to infer any general tendency from those data sets.

General observations:

(a) Which classifier did better on each corpus.

Average AEOD and average AEDD is an average of all AEOD results and al AEDD results respectively given by each classifier for each corpus. They are sorted by the average AEOD, from smallest to highest.

classifier	corpus	average AEOD	average AEDD
svm	20 Newsgroups	6.849	4.403
rainbow	20 Newsgroups	11.797	11.102
c5	20 Newsgroups	14.89	15.11
ripper	20 Newsgroups	24.05	19.89
svm	Digitrad	10.51	13.06
c5	Digitrad	19.3	16.53
rainbow	Digitrad	20.54	20.38
ripper	Digitrad	28.28	24.47
svm	Yahoo	2.665	2.807
rainbow	Yahoo	3.888	3.959
c5	Yahoo	9.65	9.161
ripper	Yahoo	18.232	16.618

Table 4 Average errors of classifiers in the One - One approach

For each corpus the SVM produced smallest classification error. The order is the same for each corpus – SVM, Rainbow, C5.0 and Ripper. It agrees with results of Joachims [JOA97] stating that SVM does significantly better on text classification than other methods. The reason for it may be that SVM deals very well with noisy data of high dimensionality.

Rainbow made it second on 20Newsgroups and Yahoo corpora, third in case of Digitrad.

C5.0 did better than Rainbow on the Digitrad corpus. In other cases it placed third.

Despite the fact that Ripper produced numerically worst results for all corpora, it is the only classifier that overall shows improvement on disambiguated data most often, even though the improvement is later shown not to be significant.

The fact that C5.0 and Ripper do worse than the other classifiers may be contributed to the fact that decision trees and decision rules do not do well with data of very high dimensionality.

(b) Which corpus was easiest to classify:

classifier	corpus	average AEOD	average AEDD
svm	Yahoo	2.665	2.807
svm	20 Newsgroups	6.849	4.403
svm	Digitrad	10.51	13.06
ripper	Yahoo	18.232	16.618
ripper	20 Newsgroups	24.05	19.89
ripper	Digitrad	28.28	24.47
rainbow	Yahoo	3.888	3.959
rainbow	20 Newsgroups	11.797	11.102
rainbow	Digitrad	20.54	20.38
c5	Yahoo	9.65	9.161
c5	20 Newsgroups	14.89	15.11
c5	Digitrad	19.3	16.53

Table 5 Average errors on each corpus in the One -One approach

All classifiers except Ripper considered Yahoo Groups to be easiest to classify, followed by 20 Newsgroups and Digitrad. The results are not surprising. Digitrad, being a collection of folk songs, contains very wide and unusual vocabulary, and thus presents a significant classification challenge.

The fact that Yahoo Groups 3 out of 4 times did better than 20 Newsgroups may come from the fact, that all data sets in the Yahoo collection were unrelated, whereas some data sets in 20 Newsgroups were close in topic. Even though it is difficult to notice the general

tendency in the 20 Newsgroups set that related data sets are more difficult to classify, it may have been a factor in the overall results.

4.1.1.1.2 Average Error and Standard Deviation

In all experiments a 10-fold cross validation technique was used. Each learner system outputs results (in the form of either error or accuracy percentage) for each of the folds, and then the total result which is the average of the intermediate results. Given 10 intermediate results, a standard deviation (SD) of the result was also calculated, which gives a better understanding of the range of the possible results.
All results are in the following format:

Average Error on Original Data (AEOD) +/- SD vs Average Error on Disambiguated (AEDD) Data +/- SD

Having SD we can infer whether the difference between the results is significant or not. We assume that if

I. AEOD < AEDD and AEOD + SD < AEDD – SD, then the results obtained on disambiguated data are *significantly worse*
II. AEOD > AEDD and AEOD – SD > AEDD + SD, then the results obtained on disambiguated data were *significantly better.*

In other cases the difference in results is insignificant.

In this section we present results in the form of the significance in the difference between results obtained on original versus disambiguated data, as described in the previous section. Four possibilities are taken into account:

- If AEOD > AEDD
  - If AEOD – SD > AEDD + SD — results are significantly better on disambiguated data (SB)
  - Else — results are better, but not significantly better on disambiguated data (+)
- If AEOD < AEDD
  - If AEOD + SD < AEDD – SD — results are significantly worse (SW) on disambiguated data
  - Else — results are worse, but not significantly worse on disambiguated data (-)

Yahoo Groups:

DATA SET	C5	RIPPER	SVM	RAINBOW
avc_cfuk:	+	+	-	+
avc_cl:	-	+	-	-
avc_dp:	+	-	-	-
avc_ds:	+	+	-	-
avc_ml:	-	+	+	+
avc_pb:	-	+	-	+
avc_tuh:	-	+	-	-
cfuk_cl:	-	-	-	+
cfuk_dp:	+	+	+	-
cfuk_ds:	+	-	+	-
cfuk_ml:	-	+	-	-
cfuk_pb:	+	+	+	-
cfuk_tuh:	-	+	-	+
cl_dp:	+	+	+	-
cl_ds:	-	+	-	-
cl_ml:	+	-	-	+
cl_pb:	+	-	-	+
cl_tuh:	-	+	+	+
dp_ds:	+	-	+	+
dp_ml:	+	+	-	-
dp_pb:	+	+	-	-
dp_tuh:	+	+	-	+
ds_ml:	-	+	-	+
ds_pb:	+	+	+	-
ds_tuh:	+	+	+	-
ml_pb:	-	+	-	-
ml_tuh:	+	+	-	+
pb_tuh:	-	-	+	+

Table 6 One - One. Yahoo Groups: Overview of the results. + corresponds to improvement on disambiguated data, - indicates the results were worse on disambiguated data.

20 Newsgroups:

DATA SET	C5	RIPPER	SVM	RAINBOW
ath_comp:	+	+	+	-
ath_ibm:	-	+	-	+
ath_mac:	-	+	+	-
ath_med:	-	-	+	-
comp_ibm:	-	+	+	+
comp_mac:	-	+	+	+
comp_med:	-	+	-	-
ibm_mac:	+	+	+	+
ibm_med:	+	+	-	+
mac_med:	-	-	+	+

Table 7 One – One. 20 Newsgroups: Overview of the results. + corresponds to improvement on disambiguated data, - indicates the results were worse on disambiguated data.

Digitrad:

DATA SET	C5	RIPPER	SVM	RAINBOW
Irish_Scots:	+	+	-	-
Irish_death:	+	+	-	+
Irish_love:	-	+	+	-
Irish_marriage:	-	+	+	+
Irish_political:	+	+	-	-
Scots_death:	-	+	-	-
Scots_love:	-	+	-	+
Scots_marriage:	+	+	-	-
Scots_political:	+	+	+	+
death_love:	-	-	-	-
death_marriage:	+	+	-	+
death_political:	+	+	+	-
love_marriage:	-	-	-	+
love_political:	+	+	-	+
political_marriage	-	+	-	-

Table 8 One – One. Digitrad: Overview of the results. + corresponds to improvement on disambiguated data, - indicates the results were worse on disambiguated data.

In all cases the difference in results obtained on original and disambiguated data is insignificant in terms of standard deviation. The above table however gives an overall picture of the distribution of +'s and –'s, that is, how often there was improvement on disambiguated data versus the lack of it. As discussed before, with the exception of Ripper which shows improvement on disambiguated data most often, the distribution of

+'s and –'s does not show much consistency. Data sets that did better with some classifiers, did worse with the others. Performance of classifiers also does not hold the general tendency across the three corpora.

4.1.1.1.3 Paired T-Test

A paired t-test is a statistical test used to determine whether there is a reliable (statistically significant) difference between two means in the two observations. If D represents a difference between two observations, the hypotheses are:

H_0: D = 0 (null hypothesis)

H_1: D ≠ 0

The test statistics is t with n-1 degrees of freedom. The output can be given in the form of a P-value, which is a probability of finding a difference between two conditions. If P-value associated with t is low, say < 0.05, there is evidence to reject the null hypothesis. In other words, there is evidence that there is a difference in means across the paired observations.

The t-test was performed on results obtained from classification of original and disambiguated data, and output in the form of a P-value.

The two observations are intermediate results of the 10 fold cross validation on original and corresponding disambiguated data sets.

Example:

The following are intermediate results of each of the folds in the C5.0 10 folds cross validation of the data set alt.atheism - sci.med of the 20 Newsgroups corpus.

original	disambiguated
21.1	26.3
15.8	10.5
10	25
10	10
5	5
10	30
10	15
25	15
25	10
0	0

The P-value = 0.66664 > 0.05, which is an evidence in favour to keep the null hypothesis – there is no difference in means between these two observations. In other words, there is no significant difference in classifying original and disambiguated data.

Next example shows the results for data sets alt.atheism vs comp.sys.ibm.pc.hardware of the 20 Newsgroups corpus, classifier C5.

original	disambiguated
10.5	15.8
5.3	21.1
5	10
20	20
15	20
5	5
15	10
10	15
5	10
5	25

The P-value = 0.039301 < 0.05, which gives evidence to reject the null hypothesis – there is a difference between the means of the two observations. In other words, there is a significant difference in classifying original and disambiguated data.

The P-value < 0.05 indicates the difference, but does not give us information which of the observations is "better". What follows is a list of data sets for which the P-value is less than 0.05. Next to the P-value are the actual average errors. If the average error for

disambiguated data is smaller than the average error of the corresponding original data, we say results are significantly better (SB) for disambiguated data. In the opposite case, we say results are significantly worse (SW) for disambiguated data.

Classifier	Corpus	Data Set	P-Value	AEOD	AEDD	SW/SB
C5	20news	ath_ibm/	0.039301	9.5	15.1	SW
C5	Digitrad	death_political/	0.047766	25	16	SB
Rainbow	Yahoo	avc_ds/	0.020712	4.1	6.82	SW
Rainbow	Yahoo	ml_tuh/	0.017306	3.3	2.7	SB
Ripper	20news	ibm_med/	0.003439	31	17	SB
Ripper	Digitrad	Irish_political/	0.012599	34.2	19.1	SB

Table 9 One – One. P-Values < 0.05 of the paired t-test.

The paired t-test is more sensitive than comparing standard deviations only, and returned a bigger number of data sets with a significant difference. Neither of the listed data sets resulted in significant difference in more than one classifier. Both similar data sets (*ibm comp*) and not similar (*ibm med*) appear in the list as those where classification of disambiguated data is much better than classification of original data.

4.1.1.2 One against Rest

In each corpus each data set was classified against the remaining data sets treated as one set. All tests performed for the One against Rest approach are same as in the One against One approach, therefore the full description of tests is omitted in this section. For details refer to section 3.3.1 One against One.

4.1.1.2.1 AEOD vs AEDD

20Newsgroups:

Classifier	Data Set	AEOD-AEDD
C5	med	4.38
	ibm	2.56
	comp	1.18
	ath	1.17
	mac	0.6
Ripper	ath	0.853176
	med	0.821049
	mac	0.349585
	comp	-0.79934
	ibm	-2.724863
SVM	ath	1.1796
	mac	0.17552
	med	-0.02041
	ibm	-0.24081
	comp	-1.43264
Rainbow	comp	1.482
	mac	0.816
	ibm	0.576
	med	-0.14
	ath	-0.245

Table 10. One – Rest. 20Newsgroups, AEOD – AEDD.

The difference between AEOD and AEDD here is very small; significantly smaller than in the case of "One against One".

C5.0 is the only classifier that consistently shows improvement on disambiguated data (that is the opposite result from the "One against One" case). Ripper is second best, which is consistent with its behaviour in the "One against One" approach.

Only one data set (*mac*) shows consistent improvement when classified by the four systems. The other data sets do not show consistent tendencies.

Yahoo:

Classifier	Data Set	AEOD-AEDD
C5	cfuk	0.65
	dp	0.52
	ml	0.39
	pb	0.25
	tuh	0.12
	cl	-0.12
	ds	-0.38
	avc	-0.4
Rainbow	dp	0.838
	cl	0.448
	pb	0.432
	tuh	0.335
	ds	0.058
	ml	-0.573
	avc	-0.677
	cfuk	-2.889
Ripper	ml	1.152995
	cfuk	0.641432
	ds	0.5274
	tuh	0.381691
	cl	0.373773
	pb	0.251822
	avc	-0.657268
	dp	-1.420654
SVM	ds	0.77755
	ml	0.3996
	cfuk	0.1282
	pb	0.00164
	tuh	-0.00999
	dp	-0.12987
	cl	-0.25807
	avc	-0.25808

Table 11 One – Rest. Yahoo Groups, AEOD – AEDD.

C5.0 and Ripper show improvement most often in case of the Yahoo corpus.

One data set (*avc*) suffered from disambiguation in all cases. *Pb* benefited from disambiguation on all classifiers.

Digitrad:

Classifier	Data Set	AEOD-AEDD
C5	political	5.47
	Irish	1.16
	Scots	0
	marriage	-2.05
	death	-2.88
	love	-4.57
Ripper	death	2.426316
	political	2.417039
	love	2.00197
	Irish	0.747945
	Scots	0.176559
	marriage	-1.34889
SVM	Irish	1.65535
	marriage	0.64127
	Scots	-0.02541
	political	-0.03389
	love	-1.88135
	death	-2.95299

Table 12 One – Rest. Digitrad, AEOD – AEDD.

Rainbow is the only system that was unable to classify this corpus, therefore no results are presented.

Ripper, consistently, shows most improvement on disambiguated data.

Irish is the only data set that benefited from disambiguation when classified by the three classification systems.

General observations:

(a) Which classifier did best on each corpus:

Classifier	Corpus	average AEOD	average AEDD
SVM	20Newsgroups	6.51	6.58
Ripper	20Newsgroups	9.89	10.19
Rainbow	20Newsgroups	12.24	11.74
C5	20Newsgroups	13.65	11.67
SVM	Digitrad	11.13	11.56
Ripper	Digitrad	14.11	13.04
C5	Digitrad	15.82	16.03
Rainbow	Digitrad		
SVM	Yahoo	2.64	2.56
Ripper	Yahoo	4.2	4.04
Rainbow	Yahoo	4.77	5.03
C5	Yahoo	4.9	4.77

Table 13 Average errors of classifiers in the One - Rest approach

SVM again is the best classifier. Ripper, which was the worst classifier in the "One against One" case, here places second.

(b) Which corpus was easiest to classify:

Classifier	Corpus	average AEOD	average AEDD
C5	Yahoo	4.9	4.77
C5	20Newsgroups	13.65	11.67
C5	Digitrad	15.82	16.03
Rainbow	Yahoo	4.77	5.03
Rainbow	20Newsgroups	12.24	11.74
Rainbow	Digitrad		
Ripper	Yahoo	4.2	4.04
Ripper	20Newsgroups	9.89	10.19
Ripper	Digitrad	14.11	13.04
SVM	Yahoo	2.64	2.56
SVM	20Newsgroups	6.51	6.58
SVM	Digitrad	11.13	11.56

Table 14 Average errors on each corpus in the One -Rest approach

The tendency from "One against One" case is kept in this approach as well – Yahoo Groups show the lowest average classification error, followed by 20Newsgroups and Digitrad.

4.1.1.2.2 Average Error and Standard Deviation

Corpus	Data Set	C5	Ripper	SVM	Rainbow
20Newsgroups	med	+	+	-	-
	ibm	+	-	-	+
	comp	+	-	-	+
	ath	+	+	+	-
	mac	+	+	+	+
Yahoo	cfuk	+	+	+	-
	dp	+	-	-	+
	ml	+	+	+	-
	pb	+	+	+	+
	tuh	+	+	-	+
	cl	-	+	-	+
	ds	-	+	+	+
	avc	-	-	-	-
Digitrad	political	+	+	-	x
	Irish	+	+	+	x
	Scots	+	+	-	x
	marriage	-	-	+	x
	death	-	+	-	x
	love	-	+	-	x

Table 15 One - Rest. Overview of the results. + corresponds to improvement on disambiguated data, - indicates the results were worse on disambiguated data.

Since the results of Rainbow for Digitrad are missing, we cannot infer anything about how it did compared to other classifiers. Worst results are shown by SVM. C5.0 shows improvement on disambiguated data most often.

However, no significant improvement when taking standard deviation into account was shown.

4.1.1.2.3 Paired T-Test

The only data set that shows statistically significant improvement is political data set from the Digitrad corpus when classified by C5.0:

Political c5 P-Value: 0.0421

In all other cases the difference in results was statistically insignificant.

4.1.2 Influence of Disambiguated Words

C5.0 and Ripper are two classifiers that report their results in the form easy to comprehend by humans, showing those attributes that played the biggest role in classification. We looked inside those reports, in attempt to see whether disambiguated words are in fact found to make a difference.

Disambiguated words were chosen rarely. It may come from the fact that statistically they do not appear in texts as often as we suspected. If an ambiguous word does appear often, one of two cases are common:

(1) it appears often in various data sets used in the same meaning

(2) it appears often in one data set and therefore it is a discriminating factor by itself, no matter whether a numerical tag signifying its sense is attached to it or not.

If the word does not appear in the data set often enough, indicating its sense does not make a big difference to the classification system.

Only in isolated cases disambiguated words that appeared in "disambiguated" decision trees or decision rules were not chosen in the corresponding "original" decision trees or decision rules. It is more often the case in Ripper (decision rules) than C5.0 (decision trees).

4.1.4 Performance of Classifiers

SVM overall produced smallest classification errors on all corpora both in the "One against One" and "One against Rest" approach. This is consistent with results of Joachims [JOA97] who contributes this fact to SVM being able to deal very well with noisy and high dimensional data.

SVM did not notice much improvement when classifying disambiguated data, which may be contributed to the reasons mentioned in the previous section.

Ripper and C5.0 on average showed highest classification errors. The classifiers employ decision rules induction (Ripper) and decision trees (C5.0) algorithms, which in some ways are similar in nature. The algorithms have already been shown [COH95] to produce similar classification results, and this tendency is also shown in experiments performed here. The fact that both classifiers on average do the worst job of classifying may be attributed to the high dimensionality of data.

Rainbow, the Naïve Bayes classifier built to classify text on places second in most cases when it comes to overall classification errors. It is not as good as SVM, but still does very well on the difficult corpora.

Ripper is the system that is most sensitive to disambiguated data and considers disambiguation an advantage, although not significant. It may be contributed to the way decision rules in Ripper are constructed and then pruned. The pruning strategy in Ripper considers "replacing" or "revising" individual rules instead of just removing certain rules as is the case with C5.0. It must be in this step that more disambiguated words are retained and certain advantage taken of their presence in the disambiguated data sets.

4.2 Imbalanced Data Sets Experiments

Results of classification in the one-rest approach showed improvement on disambiguated texts more often than in the one-one approach. Data sets in the one-one approach were perfectly balanced (100 examples of each set), while data sets in the one-rest approach were not (1-7 in Yahoo Groups, 1-4 in 20 Newsgroups and 1-5 in Digitrad). That suggested that word sense disambiguation may be beneficial for classification of imbalanced data sets and a set of experiments for imbalanced data was performed.

The experimental setup and evaluation used for these experiments differed from the one used for the one-one and one-rest approach.

Same pairings of the data sets were used as in the one-one approach. Two types of experiments were performed:

- 5 examples of the first class against 100 examples of the second class
- 10 examples of the first class against 100 examples of the second class

4.2.1 Accuracy, Precision and Recall

In case of one-one and one-rest only accuracy (or error) was taken into account. However, full results of classifications are counted in four categories: correct positives, correct negatives, false positives and false negatives. These four statistics can be represented in a simple contingency table or confusion matrix, as shown in Fig below.

Correct Class	Hypothesis		
		Positive	Negative
	Positive	a	b
	Negative	c	d

Fig. 13. An example of a confusion matrix . Numbers a and d correspond to correct positive and negative classifications respectively, b and c false positive and negative classifications respectively.

The accuracy (A) of a classifier is the number of correct classifications divided by the total number of classifications:

$$A = \frac{a+d}{a+b+c+d}$$

While accuracy is an appropriate measure when classifying balanced data sets, it proves to be less informative in case of imbalanced data sets. When there are many more negative examples than positive ones, even the default classifier will yield a very high accuracy simply by classifying everything to be negative.

In order to avoid this problem, instead of looking at accuracy, the IR community considers two other statistics known as precision and recall. Roughly stated, precision (P) measures the proportion of correct positives over all examples classified as positive:

$$P = \frac{a}{a+c}$$

while recall (R) measures the proportion of correct positives over all correctly classified examples:

$$R = \frac{a}{a+b}$$

The higher these measures are, the better performance of the classifier on the positive class.

10 fold cross validation used in case of one-one and one-rest approach does not prove useful in case of imbalanced sets used for these experiments given the very small number of examples. Since the positive class is represented by only 5 or 10 examples against 100 negative examples, positive examples may not appear in some of the 10 partitions used for training or for testing.

To avoid this problem, a *leave-one-out* technique was used for the imbalanced data sets experiments. Only one example at a time was used for testing, while the remaining examples were used for training. The procedure was repeated for all examples, that is each example was used as a testing set. That meant 105 classifications for those data sets where 5 positive and 100 negative examples were used, and 110 trials for the 10-100 pairings.

4.2.2 Results and Discussion

Each table below shows:

- average precision and average recall for original and disambiguated data
- difference in average precision: disambiguated precision – original precision
- difference in average recall: disambiguated recall – original recall
- +/- number of times difference was positive "+" (better for disambiguated data) and negative "-" (worse for disambiguated data)

Corpus	original		disambiguated		difference		+/-	
	prec	rec	prec	rec	prec	rec	prec	rec
20Newsgroups	0.02	0.02	0.02	0.02	0	0	"0/0"	"0/0"
Digitrad	0.153	0.08	0.13	0.06	-0.02	-0.02	"0/1"	"0/1"
Yahoo	0.44	0.25	0.46	0.27	0.025	0.014	"3/0"	"2/0"

Table 16 Precision and recall results for imbalanced data sets 5-100 for C5.0 classifier

Corpus	original		disambiguated		difference		+/-	
	prec	rec	prec	rec	prec	rec	prec	rec
20Newsgroups	0.328	0.17	0.314	0.16	0.013	0.01	"3/1"	"1/1"
Digitrad	0.444	0.21	0.405	0.21	0.038	0	"2/3"	"1/1"
Yahoo	0.681	0.31	0.605	0.34	0.076	-0.03	"10/10"	"7/4"

Table 17 Precision and recall results for imbalanced data sets 10-100 for C5.0 classifier

Corpus	original		disambiguated		difference		+/-	
	prec	rec	prec	rec	prec	rec	prec	rec
20Newsgroups	0.553	0.292	0.53	0.278	-0.022	0.014	"0/3"	"0/2"
Digitrad	0.2	0.04	0.2	0.04	0	0	"0/0"	"0/0"
Yahoo	0.237	0.2	0.447	0.513	0.21	0.313	"5/0"	"6/0"

Table 18 Precision and recall results for imbalanced data sets 5-100 for Ripper classifier

Corpus	original		disambiguated		difference		+/-	
	prec	rec	prec	rec	prec	rec	prec	rec
20Newsgroups	0.724	0.388	0.579	0.25	-0.0145	-0.138	"2/3"	"1/0"
Digitrad	0.473	0.291	0.432	0.232	-0.04	-0.059	"4/3"	"1/3"
Yahoo	0.785	0.485	0.798	0.478	0.013	-0.007	"3/4"	"1/3"

Table 19 Precision and recall results for imbalanced data sets 10-100 for Ripper classifier

Corpus	original		disambiguated		difference		+/-	
	prec	rec	prec	rec	prec	rec	prec	rec
20Newsgroups	0.275	0.12	0.275	0.12	0	0	"0/0"	"0/0"
Digitrad	0.4	0.12	0.4	0.12	0	0	"0/0"	"0/0"
Yahoo	0.857	0.864	0.857	0.864	0	0	"0/0"	"0/0"

Table 20 Precision and recall results for imbalanced data sets 5-100 for Rainbow classifier

Corpus	original		disambiguated		difference		+/-	
	prec	rec	prec	rec	prec	rec	prec	rec
20Newsgroups	0.823	0.62	0.823	0.62	0	0	"0/0"	"0/0"
Digitrad	0.95	0.36	0.95	0.36	0	0	"0/0"	"0/0"
Yahoo	0.907	0.98	0.907	0.98	0	0	"0/0"	"0/0"

Table 21 Precision and recall results for imbalanced data sets 10-100 for Rainbow classifier

The SVM classifier again proved to give best classification results. Accuracy in almost all cases was 100% giving both precision and recall = 1.

The precision and recall results do not indicate improvement on disambiguated data. In case of Rainbow and SVM precision and recall is identical for original and disambiguated data. C5.0 and Ripper report different precision and recall results for some of the original and disambiguated data sets. In case of 5-100 data sets, there are only a few differences. There are more differences in case of 10-100 data sets. However, the results are sometimes better and sometimes worse – there is no regular tendency of improvement or worsening of the performance of these classifiers.

5. Conclusions and Future Work

Word sense ambiguity is considered to be a potential source of error in many of the natural language processing domains. Text classification, a field on intersection of machine learning and information retrieval, is also believed to suffer from this phenomenon. Choosing a proper meaning for words, or word sense disambiguation, is a growing area of research in the natural language technology field.

The purpose of this thesis was to see whether word sense disambiguation affects text categorization. The intuitive hypothesis was that text classification will benefit from word sense disambiguation.

In order to evaluate the value of this hypothesis extensive empirical experiments were performed. Three different corpora of on-line documents, Yahoo Groups, 20 Newsgroups and Digitrad, were disambiguated manually. Both original and corresponding disambiguated corpora was classified using four different classification systems, C5.0, Ripper, Rainbow and LibSVM, in order to check the sensitivity of a particular inductive method to the disambiguation.

Two different sets of experiments were performed. In the first set data was assumed to be balanced and the effectiveness of each classification was measured by looking at the average error, standard deviation and performing a paired t-test in order to compare results of classifying disambiguated and original data. The results showed no significant differences. However, whenever the data was slightly imbalanced, an insignificant improvement for disambiguated data was noticed more often than in case of perfectly balanced data. That suggested that disambiguation may in fact be beneficial for scarce data and another set of experiments was performed, this time for strongly imbalanced data sets. In this case instead of accuracy, the results were evaluated by looking at precision and recall.

In each corpus only a few data sets improved from disambiguation when classified by all classifiers. In most cases the results vary – improvement can be observed for some classification systems, and in the other the results are worse.

In all cases the results show no significant improvement in classification of disambiguated data over classification of original data.

The results do not support the stated hypothesis very strongly, however some general tendencies can be observed. Ripper, despite giving usually highest classification errors, benefited from disambiguation most often. C5.0 reported improvement on disambiguated data more than half the times. Other classifiers, Rainbow and LibSVM give low average errors, however, no significant improvement on disambiguated data can be observed.

The data was of a very high dimensionality, which was not reduced in any way other than removing stop words and standalone non-numerical characters. Any other more advanced terms removal in order to reduce dimensionality of data was not applied in order to avoid removing ambiguous words, since they were the main subject of this work.

Process of disambiguation increases dimensionality of data. Instead of one ambiguous word *w* appearing across an entire corpus, several words *w1, w2, w3*, ... appear instead. That might have made the task of classification significantly more difficult for some systems, for example C5.0 (decision trees) and Ripper (decision rules).

Since disambiguation was performed manually, a task tedious and very time consuming, only nouns were disambiguated. The results might have been more conclusive if full disambiguation of nouns, verbs and adjectives was performed. Also, a human error in disambiguating data must be taken into consideration.

General tendencies are observed in performance of classifiers, where SVM does best, followed by Rainbow (Naive Bayes), C5.0 (decision trees) and Ripper (decision rules). These results are not surprising. Given very high dimensionality of data, SVM is the best method to deal with it, which is consistent with results of Joachims [JOA97]. Rainbow, as

a classification system built specifically to classify text is the second best. C5.0 and Ripper's comparably worse performance may be contributed to the fact that noisy data of very high dimension may be more difficult to deal with when using decision trees or decision rules approach.

Ripper, despite showing highest average classification errors, benefited from disambiguation most often. It must be contributed to the pruning technique used in Ripper.

The other classifiers benefit from having disambiguated data only in some cases.

Another consistent result is that Yahoo Groups corpus was easiest to classify (on average produced smallest classification errors), followed by 20 Newsgroups and then Digitrad, which in most cases produced average classification errors much higher then the other two corpora. The results are not surprising. E-mails (Yahoo Groups and 20 Newsgroups) despite their informal nature are still better structured and use more common language than lyrics of folk songs. The fact that Yahoo Groups did better than 20 Newsgroups may be contributed to the fact, that data sets of Yahoo Groups were unrelated when it comes to discussed topics, while 20 Newsgroups contained a few data sets of a similar nature.

There are a number of possible directions for future research. It would be interesting to see whether the same inconclusive results hold when the full disambiguation, instead of nouns only, is performed. Given the availability of the state of the art disambiguation tools, such an experiment should not be as time consuming to set up.

Another possibility would be to use classifiers where an extra 'weight' can be given to the new arguments created by disambiguation. That would help a classifier take advantage of less common, but very distinctive ambiguous words.

Disambiguation seems not to affect classification of long texts, however, it would be worth verifying whether it affects other text learning tasks, such as sentence classification, and sentence or keyword phrases extraction. While ambiguities may not be

as significant in full documents given the presence of many other non-ambiguous words describing thecontext and character of a document, they may affect isolated sentences much more severely given their scarce representation. Sentence classification and extraction are used for text summarization and other related NLP and IR tasks.

Disambiguation should also help with the problem of finding and grouping synonyms in text. This, in turn, might help with the features extraction needed for text classification.

6. Appendix

Short names of actual data sets, as used in section Results.

CORPUS	Actual name of data sets	Short name of data sets used in Results
20Newsgroups	alt.atheism	ath
	comp.graphics	comp
	comp.sys.ibm.pc.hardware	ibm
	comp.sys.mac.hardware	mac
	sci.med	med
Yahoo Groups	machine-learning	ml
	perl-beginner	pb
	distantsuns	ds
	digital-photography	do
	The_Unhandled_Horse	tuh
	avconnection	avc
	ChildrenfreeUK	cfuk
	CelticLore	cl
DigiTrad	Irish	irish
	Scots	scots
	Political	political
	Death	death
	Marriage	marriage
	Love	love

7. References

[BRI92] Eric Brill. A Simple Rule-based Parts of Speech Tagger. *In Proceedings of the 3rd Conference on Applied Natural Processing* (Trento, Italy 1992).

[CLS03] P.-H. Chen, C.-J. Lin, and B. Schölkopf. A tutorial on nu-support vector machines. May 2003.

[CHL01] Chih-Chung Chang, Chih-Jen Lin. LIBSVM: a Library for Support Vector Machines (Version 2.31). 2001

[COH95] William W. Cohen. Fast Effective Rule Induction. *In Proc. ICML-95. 1995.* 115-123.

[COH96] William W. Cohen. Learning Trees and Rules with Set-valued Features .In *AAAI-96.* 1996.

[DUM98] Susan Dumais. Using SVMs for Text Categorization. In *IEEE Intelligent Systems Magazine,* Trends and Controversies, Marti Hearst, ed., 13(4), July/August 1998.

[FEL98] Christiane Fellbaum (ed). WordNet – an Electronic Lexical Database. The MIT Press. 1998.

[FRW98] Eibe Frank, Ian H. Witten. Generating Accurate Rule Sets Without Global Optimization. In *Proc. 15th International Conf. on Machine Learning.* 1998

[FUW94] Johannes Furnkranz and Gerhard Widmer. Incremental Reduced Error Prunning. *Proc. ICML-94.* 1994. 70-77.

[GRE96] Dick Greenhaus. About the Digital Tradition. www.mudcat.org/DigiTrad-blurb.html. 1996

[HCL03] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin. A Practical Guide to Support Vector Classification.July 2003.

[JOA97] Thorsten Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *Proceedings of ECML-98, 10th European Conference on Machine Learning.*1997

[KIL98] Adam Kilgarriff . SENSEVAL: An Exercise in Evaluating Word Sense Disambiguation Programs. *ITRI-98-09.*1998

[KRC92] Robert Krovetz, W. Bruce Croft. Lexical Ambiguity and Information Retrieval. In *ACM Transactions on Information Systems* (TOIS). April 1992

[LEW96] David Lewis. Natural language processing for information retrieval. In *Communications of the ACM,* 1996.

[LSM94] Xiaobin Li, Stan Szpakowicz and Stan Matwin. A WordNet-based Algorithm for Word Sense Disambiguation. In *Machine Learning*. 1994. 1368-1374.

[MST94] D. Michie, D.J. Spiegelhalter, C.C. Taylor (eds), Machine Learning, Neural and Statistical Classification. New York Ellis Horwood. 1994

[MIH99] Rada Mihalcea, Word Sense Disambiguation and its Application to Internet Search, Master's Thesis, April 13, 1999.

[MIL90] George A. Miller. WordNet: an On-line Lexical Database. *In International Journal of Lexicography 3(4).* 1990. 235-244.

[MIT97] Tom Mitchell. Machine Learning. McGraw Hill. 1997

[MMRT01] K.-R. Muller, S. Mika, G. Riitsch,, K. Tsuda, B. Scholkopf, "An Introduction to kernel-based learning algorithms", *IEEE Transactions on Neural Networks,* Vol. 12, pp.181 201, 2001.

[MOM00] Dan Moldovan and Rada Mihalcea Using WordNet and Lexical Operators to improve Internet Searches in *IEEE Internet Computing*, vol. 4 no. 1, 2000.

[QUI90] J.R.Quinlan. Learning Logical Definitions from Relations. Machine Learning 5:3. August 1990. 236-266.

[QUI93] J.R.Quinlan. C4.5: Program for Machine Learning. San Mateo, CA: Morgan Kaufmann. 1993

[REY97] Philip Resnik, David Yarowsky. A Perspective on Word Sense Disambiguation Methods and Their Evaluation. In *Proceedings of SIGLEX '97*, Washington, DC, pp. 79-86, 1997.

[SEB02] Fabrizio Sebastiani. Machine Learning in Automated Text Categorization. In *ACM Computing Surveys* 34(1), 1--47, 2002

[SCH95] Bernhard Scholkopf. SVMs – A Practical Consequence of Learning Theory. IEEE Intelligent Systems, July/August, 18-21 1995.

[SCO98] Sam Scott. Feature Engineering for a Symbolic Approach to Text Classification . Master's Thesis.1998.

[SCM98] Sam Scott, Stan Matwin. Text Classification Using WordNet Hypernyms. In *Usage of WordNet in Natural Language Processing Systems: Proceedings of the Workshop (COLING-ACL'98)*. August 1998. 45-51.

[VOO93] Ellen M. Voorhes. Using WordNet to Disambiguate Word Senses for Text Retrieval. In *Proc. SIGIR-93*. 1993. 171-180.

[YAR95] David Yarowsky. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc of the 33rd Meeting of the ACL*. June 1995. 189-196.