Nada är en gemensam institution mellan
Kungliga Tekniska högskolan och Stockholms universitet.

Clustering in Swedish

The Impact of some Properties of the Swedish Language on
Document Clustering and an Evaluation Method

MAGNUS ROSELL

Avhandling som med tillstånd av Kungliga Tekniska högskolan
framlägges till offentlig granskning för avläggande av teknologie licentiatexamen
tisdagen den 18 oktober 2005 kl 10.15
i sal E3, Osquars Backe 14,
Kungliga Tekniska högskolan, Stockholm.

TRITA-NA-05-31
ISSN 0348-2952
ISRN KTH/NA/R-05/31-SE
ISBN 91-7178-166-8

TRITA-NA-05-31
ISSN 0348-2952
ISRN KTH/NA/R-05/31-SE
ISBN 91-7178-166-8

KTH School of
Computer Science and Communication
SE-100 44 Stockholm
SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framläggges till offentlig granskning för avläggande av teknologie licentiatexamen tisdagen den 18 oktober 2005 klockan 10.15 i sal E3, Osquars Backe 14, Kungl Tekniska högskolan, Stockholm.

Tryck: Universitetsservice US AB

Abstract

Text clustering divides a set of texts into groups, so that texts within each group are similar in content. It may be used to uncover the structure and content of unknown text sets as well as to give new perspectives on known ones. The contributions of this thesis are an investigation of text representation for Swedish and an evaluation method that uses two or more manual categorizations.

Text clustering, at least such as it is treated here, is performed using the vector space model, which is commonly used in information retrieval. This model represents texts by the words that appear in them and considers texts similar in content if they share many words. Languages differ in what is considered a word. We have investigated the impact of some of the characteristics of Swedish on text clustering. Since Swedish has more morphological variation than for instance English we have used a stemmer to strip suffixes. This gives moderate improvements and reduces the number of words in the representation.

Swedish has a rich production of solid compounds. Most of the constituents of these are used on their own as words and in several different compounds. In fact, Swedish solid compounds often correspond to phrases or open compounds in other languages. In the ordinary vector space model the constituents of compounds are not accounted for when calculating the similarity between texts. To use them we have employed a spell checking program to split compounds. The results clearly show that this is beneficial.

The vector space model does not regard word order. We have tried to extend it with nominal phrases in different ways. None of our experiments have shown any improvement over using the ordinary model.

Evaluation of text clustering results is very hard. What is a good partition of a text set is inherently subjective. Automatic evaluation methods are either intrinsic or extrinsic. Internal quality measures use the representation in some manner. Therefore they are not suitable for comparisons of different representations.

External quality measures compare a clustering with a (manual) categorization of the same text set. The theoretical best possible value for a measure is known, but it is not obvious what a good value is – text sets differ in difficulty to cluster and categorizations are more or less adapted to a particular text set. We describe an evaluation method for cases where a text set has more than one categorization. In such cases the result of a clustering can be compared with the result for one of the categorizations, which we assume is a good partition. We also describe the kappa coefficient as a clustering quality measure in the same setting.

Sammanfattning

Textklustring delar upp en mängd texter i grupper, så att texterna inom dessa liknar varandra till innehåll. Man kan använda textklustring för att uppdaga strukturer och innehåll i okända textmängder och för att få nya perspektiv på redan kända. Bidragen i denna avhandling är en undersökning av textrepresentationer för svenska texter och en utvärderingsmetod som använder sig av två eller fler manuella kategoriseringar.

Textklustring, åtminstonde som det beskrivs här, utnyttjar sig av den vektorrumsmodell, som används allmänt inom området. I denna modell representeras texter med orden som förekommer i dem och texter som har många gemensamma ord betraktas som lika till innehåll. Vad som betraktas som ett ord skiljer sig mellan språk. Vi har undersökt inverkan av några av svenskans egenskaper på textklustring. Eftersom svenska har större morfologisk variation än till exempel engelska har vi tagit bort suffix med hjälp av en stemmer. Detta ger lite bättre resultat och minskar antalet ord i representationen.

I svenska används och skapas hela tiden fasta sammansättningar. De flesta delar av sammansättningar används som ord på egen hand och i många olika sammansättningar. Fasta sammansättningar i svenska språket motsvarar ofta fraser och öppna sammansättningar i andra språk. Delarna i sammansättningar används inte vid likhetsberäkningen i vektorrumsmodellen. För att utnyttja dem har vi använt ett rättstavningsprogram för att dela upp sammansättningar. Resultaten visar tydligt att detta är fördelaktigt.

I vektorrumsmodellen tas ingen hänsyn till ordens inbördes ordning. Vi har försökt utvidga modellen med nominalfraser på olika sätt. Inga av våra experiment visar på någon förbättring jämfört med den vanliga enkla modellen.

Det är mycket svårt att utvärdera textklustringsresultat. Det ligger i sakens natur att vad som är en bra uppdelning av en mängd texter är subjektivt. Automatiska utvärderingsmetoder är antingen interna eller externa. Interna kvalitetsmått utnyttjar representationen på något sätt. Därför är de inte lämpliga att använda vid jämförelser av olika representationer.

Externa kvalitetsmått jämför en klustring med en (manuell) kategorisering av samma mängd texter. Det teoretiska bästa värdet för måtten är kända, men vad som är ett bra värde är inte uppenbart – mängder av texter skiljer sig åt i svårighet att klustra och kategoriseringar är mer eller mindre lämpliga för en speciell mängd texter. Vi beskriver en utvärderingsmetod som kan användas då en mängd texter har mer än en kategorisering. I sådana fall kan resultatet för en klustring jämföras med resultatet för en av kategoriseringarna, som vi antar är en bra uppdelning. Vi beskriver också kappakoefficienten som ett kvalitetsmått för klustring under samma förutsättningar.

Acknowledgements

I would like to thank the following people (and many more) for their contributions, support and company.

- My supervisor professor Viggo Kann.
- All participants in the Infomat project and The Swedish Research Council (VR) for funding.
- Johnny Bigert, Johan Carlberger, Hercules Dalianis, Martin Hassel, Ola Knutsson, Jonas Sjöbergh, all in the Language Technology Group at Nada.
- Students and supervisors in GSLT (The National Graduate School of Language Technology), among others: Joakim Nivre, Magnus Sahlgren.
- The Theory Group at Nada.
- My room mates Gustav Hast, Rafael Pass, Anna Redz.
- Fellow teachers: Linda Kann, Alexander Baltatzis, Olle Bälter, Ann Bengtsson, Henrik Eriksson, Vahid Mosavat, Isaac Elias, and several others.
- Students: Oscar Täckström, Sumithra Velupillai, and all attending my classes.
- All teachers I have had through the years at all levels in the fantastic Swedish school system.
- All friends, among others: Jakob von Döbeln, Charlotte Eriksson, Kimmo Eriksson, Maja Fjæstad, Tove Gustavi, Maja Karasalo, Arun Kaul, Olof Kjellström, Pär Lindahl, Ylva Leandersson, Martin Mossberg, Hedvig Sidenbladh, Fredrik Zetterling, Robert Zettinger, Jens Waltin, Maria Ögren.
- In retrospect, all people involved in the student comedy group (spexet) Fysikalen.
- My family: Kerstin and Örjan Rosell, my sister Maria and her husband David Löfgren, and my grandfather (morfar) Per Henry Lindroth.

Contents

Chapter 1

Introduction

The Swedish Twin Registry[1] contains much information on Swedish twins. By studying and combining this information researchers may discover what causes a disease. In one of the questionnaires, that are used to gather the information, a portion of the twins were asked to describe their main occupation in a few words or sentences. The result is about 44 000 short texts. Obviously, this is too many for a human to get a general view of. Questions with multiple choices and numerical data, as for instance age, are easy to treat statistically, but text is much more complex. An automatic or semi-automatic tool that could find the main trends in text sets would be of great help.

To cluster a set of objects means to automatically partition them into clusters (parts), so that objects in the same cluster are as similar to each other as possible, and at the same time as dissimilar to objects from other clusters as possible. In text clustering the objective is to group texts with similar content together. Such clusters (or groups) give a view of a text set and could be used to find structure in for instance the occupation answers of the previous paragraph.

Text clustering is an application within the large and growing field of information retrieval, a sub-area of natural language processing. The search engine[2] is the most well known information retrieval tool. Text clustering can also be applied to the documents retrieved by a search engine, so that they can be presented in groups according to content[3].

There are many different clustering algorithms. Most are based on a definition on similarity between the objects. How the objects are represented and the definition of similarity differ between applications. If, for example, the objects are represented in a n-dimensional Cartesian space, dissimilarity may be defined as Cartesian distance. In information retrieval applications texts are represented by the words that appear in them and similarity between two texts is calculated by

[1]See Section 1.4 and http://www.meb.ki.se/twinreg/index_en.html.
[2]For instance Google, http://www.google.com/.
[3]Try the search engine Vivisimo, http://vivisimo.com/.

considering the words that appear in both of them. This model is employed with great success in search engines and it is commonly used for text clustering as well.

This thesis is about text clustering. We have investigated the impact on clustering of some aspects of Swedish and introduced a new way of evaluating clustering results.

1.1 Motivation

To categorize is part of human nature. Few things are more important to our survival. The history of mankind is also the history of our accelerating knowledge and implementation of ways to divide things into comprehensible categories. From the early humans who consciously separated eatable from in-eatable to the coolers of todays supermarkets is a long line of development. Humans force structure on their environment in every aspect of their lives.

To improve our ability to categorize we have developed many ingenious tools. Two of the most profound are the written language and the computer. Using both we have explored structures and made new classifications. But both have as a consequence even more material to consider. To find structure in these ever increasing streams of information we turn our hopes to automatic and semi-automatic tools. We now use computers to explore and structuralize information, a lot of which is in text form.

We, humans, categorize texts in many different ways. Libraries have systems of genres, newspapers use sections for different kinds of news etc. Most of the time, however, one could come up with many different, but valid and valuable, partitions of the same set of texts. Furthermore, in most cases partitions of the same set of texts made by different people are not similar. This is not necessarily something bad. Any new partition of a set of texts may give new insights if one can understand and accept the reasoning used to accomplish it.

Partitions of texts may become obsolete or irrelevant for a certain investigation. New texts may not fit into an old structure or might force a change in a structure. To make a new partition manually is very expensive and time consuming. Automatic tools that partition texts or extract reasonable groups of texts could be very valuable. Even if the partitions made by automatic tools get worse than those accomplished by a human they still would be valuable, since in most cases no-one would ever make a partition manually.

From one point of view, what is a *good* partition of a set of texts depends on the reasoning that is used in the creation of the partition, whether it is sound and if it is used in a consistent manner. The computer is superior when it comes to consistence, but the reasoning must in some manner be supplied by a human.

1.2 Clustering vs. Categorization

By automatic categorization we mean to let a machine decide to which of a set of predefined categories a text belongs. In clustering the machine decides how a given text set should be partitioned. Categorization is suitable when one wants to categorize new texts according to a known categorization, clustering when one wants to discover new structures not previously known. Both methods may give interesting results on an unknown text set; categorization sorts them according to a well known structure, clustering displays the structure of the particular set. This thesis deals with clustering of texts.

1.3 Some Terminology

Some terms that are used throughout this thesis may require explanations. The objects we cluster are sequences of *words* (sometimes we use the word *term* instead of word) we refer to as *texts*, *documents*, *articles* or *papers* depending on the context. We refer to a set of such objects as a *text set*, a *document collection* or a *corpus.* A set may be grouped in several manners: either by a *clustering algorithm* in which case we talk about a *clustering* consisting of *clusters*, or by humans according to some agreement in which case we talk about a *classification* or a *categorization* that consists of *classes* or *categories.*

1.4 The Infomat Project

This thesis is written within the Infomat project[4] at KTH Nada[5] in the informal Human Language Technology Group. Infomat is an abbreviation for *Swedish information retrieval with language technology and matrix computations*, which well summarizes what we are interested in.

The project is a cooperation with the department of Medical Epidemiology and Biostatistics (MEB) at Karolinska Institutet (Swedish Medical University, KI), Stockholm, Sweden. KI has many texts in Swedish dealing with medical issues and free text answers in questionnaires answered by many people. Among other things they administrate The Swedish Twin Registry[6], the largest twin registry in the world with more than 140 000 twins. The registry contains much information, most of which is collected using questionnaires, with both closed questions (multiple choice questions) and open questions that require a free text answer.

As mentioned in the beginning of this chapter it is hard to analyze big text sets manually. We are interested in the use of language technology tools to aid such

[4]See http://www.nada.kth.se/theory/projects/infomat/

[5]The Department of Numerical Analysis and Computer Science (Nada) at The Royal Institute of Technology (KTH), Stockholm, Sweden. As from spring 2005 Nada has been merged into the newly formed School of Computer Science and Communication.

[6]See http://www.meb.ki.se/twinreg/index_en.html.

analyzation. In particular, we want to investigate the use of text clustering as a tool for exploration of this kind of texts. To this end we have worked with a free text answer in which a portion of the twins, in the registry, describe their main occupation in a few words or sentences. It was answered by about 44 000 twins. In paper II (see Section 1.5) we report on this first investigation.

1.5 Outline

This thesis is what in Swedish is known as a *sammanläggningsavhandling*[7], which means that it consists of a few papers with an introductory part. It is divided into three parts, excluding this introduction. The first gives a brief introduction to Information Retrieval in general and Document Clustering in particular. Part two summarizes the contributions of the papers. The last and third part contains the following three conference papers:

Paper I. Magnus Rosell: "Improving Clustering of Swedish Newspaper Articles using Stemming and Compound Splitting", NoDaLiDa 2003, Reykjavik, Iceland, 2003. (Rosell, 2003)

Paper II. Magnus Rosell, Viggo Kann, Jan-Eric Litton: "Comparing Comparisons: Document Clustering Evaluation Using Two Manual Classifications", ICON 2004, Hyderabad, India, 2004. (Rosell *et al.*, 2004)

Paper III. Magnus Rosell, Sumithra Velupillai: "The Impact of Phrases in Document Clustering for Swedish", NoDaLiDa 2005, Joensuu, Finland, 2005. (Rosell and Velupillai, 2005)

[7]This word is a good example of a Swedish solid compound, something that is discussed a great deal in this thesis. It could be split at several levels. At the top level it is constructed from two words. The first is *sammanläggning* which means something like "a put-together" and is constructed from the words *samman*, which means *together*, and *läggning*, a noun constructed from the verb *lägga*, which means *put*. The second word is *avhandling*, the Swedish word for thesis and originally a loanword from German. It is a lexicalized word but could be considered created from *handling* (here, something like *document*) and *av*, which is a preposition that in this case is closest to *of* in English, indicating (in my understanding) that this document treats a subject rather thoroughly.

Part I

Background

Chapter 2

Information Retrieval

Information Retrieval (IR) is a large field within Natural Language Processing (NLP). The search engine is the most well-known (and perhaps still the only really useful) application. Search engines like Google[1] and AltaVista[2] are used by many people on a daily basis.

Many document clustering methods use the same theoretical foundation as search engines, the vector space model. It is a model for representing (the content of) texts. The following sections give a brief introduction to it. There are many texts that describe the vector space model, see for instance (Baeza-Yates and Ribeiro-Neto, 1999; Frakes and Baeza-Yates, 1992; Jurafsky and Martin, 2000; Manning and Schütze, 1999; Van Rijsbergen, 1979).

In the vector space model each text in a set of texts is represented by a vector in a high-dimensional space, with as many dimensions as the number of different words in the set. Each text gets weights (values) in the indices (dimensions) based on what words appear in them. These weights model how important the corresponding word is deemed to be to explain the content of the text. They are dependent on whether (and how often) the word appears in the document and in the entire set. Texts whose vectors are close to each other in this space are considered being similar in content.

2.1 Representation

Consider a text set with n texts that uses a set of ω different words. Each text is represented by a vector:

$$d_j = (w_{1,j}, w_{2,j}, \dots w_{\omega,j}) \tag{2.1}$$

where $j \in \{1 \dots n\}$ and $w_{i,j}$ is the weight given to word i in text j. By joining these vectors we get the *word-by-document matrix*, with elements $w_{i,j}$.

[1]http://www.google.com/
[2]http://www.altavista.com/

In the most common weighting schemes (there are many variants) the weights are the product of two factors, the *term frequency* (tf) and the *inverse document frequency* (idf):

$$w_{i,j} = tf_{i,j} \cdot idf_i. \tag{2.2}$$

The term frequency is a function of the number of occurrences of the particular word in the document divided by the number of words in the entire document. A word appearing frequently in the text is thus deemed more important to describe the content than a word appearing less often. The inverse document frequency models the distinguishing power of the word in the text set; the fewer documents that contain the word the more information about the text in the text set it gives. There are many variants of the idf-measure. A simple example is: $idf_i = \log(n/n_{\mathrm{word}(i)})$[3], where $n_{\mathrm{word}(i)}$ is the number of documents that word i appears in.

In all works presented here we use the following weighting scheme:

$$\begin{aligned} \mathrm{tf}_{i,j} &= c_1 + (1 - c_1)\frac{n_{i,j}}{\max_i n_{i,j}}, & (2.3) \\ \mathrm{idf}_i &= c_2 + \log\frac{n - n_{\mathrm{word}(i)}}{n_{\mathrm{word}(i)}}, & (2.4) \end{aligned}$$

where $n_{i,j}$ is the number of times word i appears in document j ($\max_i n_{i,j}$ is the number of times the most frequent word in text j appears), and we have used $c_1 = 0.5$ and $c_2 = 0.5$.

2.2 Similarity

When using a search engine the user wants to retrieve relevant documents. He or she gives some keywords, a query, as input. This query, gets represented in the same (or a similar) way as the texts, i.e. we get a vector q in the vector space representing the query. The idea is that the relevant texts are those that are closest to the query in the vector space. The most common measure of closeness or similarity is the *cosine measure*, the cosine of the angle between the query and a text:

$$\mathrm{sim}(q, d_j) = \frac{q \cdot d_j}{|q||d_j|} = \frac{1}{|q||d_j|}\sum_i q_i w_{i,j} \tag{2.5}$$

In the basic search engine model, the texts are returned to the user in order of similarity to the query. This means that they are *ranked*[4].

The similarity measure may also be used to measure similarity between texts. The cosine measure is not affected by the size of the documents. It merely considers

[3]The logarithm is used because otherwise the function would grow too fast with decreasing $n_{\mathrm{word}(i)}$. This is related to Zip's law. See (Manning and Schütze, 1999) for an introduction to Zip's law.

[4]See Section 2.4, for more on ranking, using *meta-data*.

the proportions of the words in the document (the normalized vectors). This is intuitively appealing: two texts of different sizes covering the same topics are similar in content.

2.3 Evaluation

It is generally hard to evaluate an Information Retrieval system. They are all in some manner working with a concept of relevance, which is an inherently subjective matter. For web search engines the problem is worse as there is no way to assess the relevance of all web pages. Most evaluation of search engines and the like is thus made on small controlled text sets like those provided by TREC[5], CLEF[6] and others. Hence the results are at least partially questionable. To do better evaluation time and money consuming interviews or questionnaires would need to be carried out. But then again these would be answered by humans with different subjective notions of relevance.

Each text in a set may be retrieved, which means that it is considered relevant by the search enginge. In a controlled set each text is also deemed relevant or not by human(s) with respect to the particular query. By considering the outcome from these perspectives one may define performance measures for the search engine. The two most common are the precision, p, and the recall, r:

$$p = \frac{|rel \cap ret|}{|ret|}, \tag{2.6}$$

$$r = \frac{|rel \cap ret|}{|rel|}, \tag{2.7}$$

where rel is the set of relevant texts in the entire collection and ret is the set of texts retrieved by the search engine. There is a (perhaps obvious?) connection between the two measures: a higher precision usually causes a lower recall and vice versa. To give more information about the performance of a search engine the precision at different levels of recall is often given as a graph.

A search engine is used by many people for very varied reasons. While a researcher may bear with many non relevant texts to find as many relevant texts as possible (high recall) a layman interested in a brief description of a subject only wants relevant answers (high precision).

There exists some measures that try to combine precision and recall when one has an opinion on the relative importance of the two. The most common is the F-measure:

$$F_\beta = \frac{(\beta + 1)pr}{\beta^2 p + r}. \tag{2.8}$$

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

[6]Cross Language Evaluation Forum, http://www.clef-campaign.org/

When β is set to one, precision and recall are considered equally important, when it is set lower than one recall is believed to be more important, and precision is decided more important when it is set higher than one.

Most evaluation measures are questionable, at least since they are dealing with *relevance*. What we do know is that people find search engines useful – they are the primary tool for finding things on the Internet, and most of the time an experienced user succeeds in finding relevant information. Still, most people would agree that search engines could be better. They could be more *intelligent*, being able to understand the need of the user and to present different alternatives in a way the user can understand.

2.4 Modifications

The model described so far (Sections 2.1 and 2.2) is a statistical model based solely on the idea that the words in a text are a reasonable representation of the content. There is no information of the order of the words. Therefore some refer to a representation of a text in this model as a *bag of words*.

The actual content of a text is of course something else than the words in the vector space model. The model merely represents the content in the same manner as the text itself does. In fact what we primarily want to model is the similarity in content between texts, for instance a query and the documents in a text set.

It is easy to object to the vector space model. Still the efficiency and usefulness of this simple model is proven by the highly useful search engines most of us use regularly. The user interfaces of search engines often highlight the search words in the retrieved texts and thus make us aware that the method for finding them is essentially simple word matching.

This section describes some modifications of the vector space model that have been tried, some of which have proven very useful.

Stoplist and Word Classes

One very common modification is to use a stoplist during the indexing, i.e. when creating the representation. The words in the stoplist are simply excluded. A stoplist may be constructed by considering the most frequent words in a large text set and/or function words, such as *and, or, to, the*, etc. These words do not contribute to the content of the texts. Their appearance in one document does not separate it from other documents.

A stoplist mostly consists of functional words from closed word classes. Taking this idea one step further the index (the word-by-document matrix) may be built using only open word classes. Considering the open word classes nouns are the ones that one intuitively connects to content. A representation not containing all words is probably not a good idea for a search engine, as it restricts the usage. But for other IR applications it might be beneficial reducing the processing time and possibly improving quality.

Phrases

Most search engines provide "phrase" search, the possibility to search for the occurrence of a sequence of words (word-n-grams) rather than for a set of words. There are no linguistic considerations taken; all sequences of words that appear in the texts are regarded. To make this work the stoplist has to be abandoned (it is still used for sets of words). In (Williams *et al.*, 2004) a few different representations of sequences of words are compared.

Lemmatizing and Stemming

Different forms of words can be a problem; if you are searching for *cars* you probably also want to get all texts with the singular form *car* in them as well. For languages with richer morphology than English this problem can be severe.

The solution is to use the lemma form of words when indexing. There are automatic *lemmatizers.* Another possibility is a *stemmer*, which strips affixes from the word, following manually written rules, and forms a *stem.* The stem is not necessarily a linguistic unit. The objective when constructing the stemmer is rather an improved performance of the application it will be used in (here: a search engine). A stemmer may give morphological variants (inflections) of the same term the same stem. In addition it may also give derived terms the same stem as the term they are derived from. For instance *cycle* and *cycling* could be given the stem *cycl.* This might both improve results and cause confusion, so care has to be taken when constructing the rules.

Related Words

Words are not unrelated as the vector space model suggests. There are many kinds of relations between words (for example homonymy, polysemy, synonymy, and hyponymy) that are potential problems for this model. It is very hard to know exactly which relations are at work for a certain query. Many different attempts to deal with the different relations have been reported.

In word sense disambiguation (WSD) one tries to decide which meaning is used for polysemous terms. Sanderson (Sanderson, 2000) summarizes the work in WSD for IR. It seems that the accuracy of a disambiguator has to be very good to improve information retrieval. How much a system would improve will depend on several factors, among others the length of the queries and the documents. This has probably a strong connection to the *collocation effect* (Krovetz and Croft, 1992); a query with many words, at least partly defines the meaning of a homograph in it, since the documents that are retrieved contain most of the words and these tend to come from the same domain.

Most queries put to search engines are short. Many relevant documents not containing the few query words are not retrieved. This can, at least theoretically, be remedied by *query expansion*, in which the query is expanded with words that are related to those already in it. This may be accomplished in many ways. In *relevance feedback* the user marks some of the retrieved documents as relevant. The system then uses these to reformulate the query, by for instance expanding it with frequent words in the relevant documents.

Most methods for query expansion not involving a user utilize some sort of thesaurus, which may be either manually constructed or built using statistics of word cooccurrences. Manually constructed theasuri with word relations, such as WordNet[7], are often elaborate and provide many kinds of relations. They are normally constructed for general purposes and the many relations may be hard to adapt to a specific task.

"Thesauri" that are constructed using statistical methods are easily adapted to a specific task by choosing the appropriate text set to extract the relations from. For query expansion one may use for instance the documents retrieved by the original query, the entire text set or any other (perhaps similar) text set. Most statistical methods do not, however, distinguish between relations, but rather deem words related or not.

Statistically Related Words

Most methods for automatically finding statistically related words build some kind of representation for each word from the contexts in which they appear throughout some (large) text data. The size of the contexts may be anything from just a few words to whole documents. The representations are then compared using some similarity measure and the word pairs with high similarity are considered related.

The word-by-document matrix (see Section 2.1) contains just this kind of information. If one compares the words (i.e. their corresponding rows in the matrix) with for instance the cosine measure, words that appear together in many documents get a high value.

Latent Semantic Analysis (LSA) is Singular Value Decomposition (SVD) performed on the word-by-document matrix, see for instance (Berry *et al.*, 1999). By projecting the word representations onto the subspace defined by the eigenvectors with the highest eigenvalues a more compact representation is constructed and noise is removed from the data. This dimension reduction brings statistically related words and documents closer to each other and is sometimes described as uncovering latent semantic relations. LSA has been shown to give improvements in results for search engines and is called LSI (Latent Semantic Indexing) in this context (Deerwester *et al.*, 1990).

LSA is computationally heavy and starts with the full word-by-document matrix. Random Indexing (RI) is a much faster alternative that uses less memory as it does not utilize the full word-by-document matrix (Kanerva *et al.*, 2000). RI gives each word a random label, a vector of a predefined length (typically a few thousand) with all but very few (typically 10) randomly selected non-zero elements

[7]http://wordnet.princeton.edu/

that are set to either one or minus one. For each word a context vector is created by adding the random labels of words that appear in its context. The context may for instance be a window of two or three words on either side, and the addition may be weighted by the distance to the center word. Words that appear in similar contexts get similar context vectors. Two words are considered related if their context vectors are deemed similar by a similarity measure, like the cosine measure. (Sahlgren, 2005)

Meta-data

Meta-data (found in web pages) give additional information that may be used when indexing. Words appearing in headings and boldface are probably more important than other words. Other text in meta-data tags that are or are not visible through a browser may also be used. Big search engines use this information a lot, see for instance (Brin and Page, 1998).

Presumably, the most important use of meta-data for web search engines is the exploitation of the link structure. Google[8] uses PageRank (Page *et al.*, 1998) to rank the search result. Each web site has a PageRank that depends on the number of links from other sites and the PageRank of these sites.

Swedish

The results using different linguistic features depend on the language. Swedish is rather rich in morphology. Stemming improves precision and recall by 15 and 18 %, respectively, in information retrieval for Swedish (Carlberger *et al.*, 2001).

According to (Hedlund *et al.*, 2001) there is much to be gained from proper linguistic treatment of the Swedish language for information retrieval. In particular they point to the rich production of solid compounds and the high frequency of homographic words. A later study (Hedlund, 2002) found that 10 % of the content words (i.e. words remaining after the use of a stoplist) of running text are compounds, meaning that more than 20 % of the morphemes are found in compounds. This suggests that splitting solid compounds should be important in any information processing of Swedish. Improvements in search results using compound splitting for Swedish have been reported (Chen and Gey, 2003; Dalianis, 2005).

[8]http://www.google.com/

Chapter 3

Document Clustering

The objective of clustering is to partition an unstructured set of objects into clusters (groups). One often wants the objects to be as similar to objects in the same cluster and as dissimilar to objects from other clusters as possible. Clustering has been used in many different areas and there exist a multitude of different clustering algorithms for different settings. For a review, see for instance (Jain *et al.*, 1999).

To use most clustering algorithms two things are necessary:

- an object representation,
- a similarity (or distance) measure between objects.

A clustering algorithm finds a partition of a set of objects that fulfills some criterion based on these conditions.

Clustering is an unsupervised learning method. The result (the clustering, the partition) is based solely on the object representation, the similarity measure and the clustering algorithm. If these correspond to the users understanding the result might well be an intuitive and useful clustering. One must keep in mind, though, that clustering algorithms always produce clusterings, even when this is not justified, and that there in most cases exist many relevant clusterings of a set of complex objects.

In document clustering the objects are texts or documents. To represent these the vector space model of the previous chapter is commonly used, see section 3.2.

3.1 Clustering Algorithms

Many different clustering algorithms have been proposed and tried for document clustering. In this section a few basic algorithms are presented.

Clustering algorithms may be divided into groups on several grounds, see (Jain *et al.*, 1999). *Hierarchical* algorithms produce a hierarchy of clusters, while *partitioning* algorithms gives a flat partition of the set. In a *hard* clustering each object belongs to only one cluster. When objects belong to more than one cluster (usually with a degree of membership) one talks about a *fuzzy* clustering.

Partitioning Algorithms

The perhaps most common clustering algorithm is K-Means, which is described in most texts on clustering (see for instance (Jain *et al.*, 1999)). Figure 3.1 gives the basic algorithm. Each step may be elaborated on with different outcomes, and there exist many variants, some given other names.

1. Pick k objects at random and let them define k clusters.
2. Calculate cluster representatives.
3. Make new clusters, one per cluster representative. Let each text belong to the cluster with the most similar cluster representative.
4. Repeat from 2 until a stopping criterion is reached.

Figure 3.1: K-Means Algorithm

The first step defines a random initial partition. There are many other ways of constructing it and the result depends on which one is used.

As cluster representative the mean (the centroid) of the objects in the cluster is usually used. Other variants are to let the median or a few specific objects represent the cluster. When the clustering is fuzzy objects may belong to several clusters and thus the cluster representative may be calculated taking this into consideration.

The stopping criterion is normally when no objects change clusters, or when very few change clusters between iterations. It may also be to stop after a predefined number of iterations, since most quality improvement usually is gained during the first iterations. The stopping criterion may also be defined using some internal quality measure, see Section 3.4.

The time complexity of the K-Means algorithm is $O(knI)$, where k is the number of clusters, n the number of objects and I the number of iterations (which is dependent on the stopping criterion). In each iteration the cluster representatives and the kn similarities between all objects and all clusters must be computed. (Hand *et al.*, 2001)

The K-Means algorithm requires a number of clusters as input. That is, one has to guess the appropriate number. Of course it is possible to run the algorithm with several different numbers of clusters and report only the clustering with the best result (as measured by, for instance, the criterion function, see below) In a general partitioning algorithm both splitting and division of clusters are allowed and theoretically the result has the optimal number of clusters.

Partitioning clustering may be viewed as an optimization problem. An instance is a particular clustering setting: a set of objects, a representation with a similarity measure, and a number of clusters. An assignment is a clustering in this setting. The objective, or criterion, function returns a value for all clusterings and the goal is to find a clustering with an optimal value. In most cases, to find such a clustering would require an exhaustive search, and most partitioning clustering algorithms are local search strategies that are only guaranteed to find a local optimum. The criterion function for the K-Means algorithm is discussed under *Internal Measures* in Section 3.4.

Hierarchical Algorithms

Hierarchical algorithms build a cluster hierarchy; clusters are composed of clusters that are composed of clusters... This may be either all the way from single documents up to the whole text set or any part of this complete structure. There are two natural ways of constructing such a hierarchy: bottom-up and top-down. The first principle is used in *agglomerative* algorithms, see Figure 3.2, and the second in *divisive* algorithms, see Figure 3.3. The stopping criterion for both algorithms may be that the desired number of cluster is reached or some limit on a criterion function or any internal evaluation measure, see Section 3.4.

The result of agglomerative clustering is strongly dependent on the similarity measure. The *single-link* method defines the similarity between two clusters as the similarity between the two most similar objects, one from each cluster. This may result in elongated, locally similar clusters. For equally sized clusters (in volume), the *complete-link* method is a better choice. Here, similarity between two clusters is defined as the similarity between the two most dissimilar objects, one from each cluster. Between these opposites there are several other measures: the centroid measure (similarity between cluster centroids), the group average measure and Ward's measure (Hand *et al.*, 2001).

The agglomerative algorithms are deterministic, generating the same cluster hierarchy every time. The similarity defintion can be viewed as the criterion function, although it is used locally for each merging and not as a global score.

The time complexity of the agglomerative algorithms are $O(n^2)$ as they all need to compute the similarity between all objects to find the pair of objects that are most similar. (Hand *et al.*, 2001)

1. Construct one cluster for each document.
2. Join the t most similar clusters.
3. Repeat 2 until a stopping criterion is reached.

Figure 3.2: Agglomerative Clustering, usually $t = 2$

In the divisive algorithms any partitioning algorithm can be applied to split clusters (step 2). The Bisecting K-Means algorithm (Steinbach *et al.*, 2000) is a divisive algorithm for document clustering that uses the K-Means algorithm to split the worst cluster in two. The worst cluster is defined as the largest, which gives equally good results as choosing the cluster with lowest intra similarity, see Section 3.4. The time complexity for the Bisecting K-Means algorithm ($O(log(k)nI)$) is lower than for the K-means algorithm as it does not compare all objects to all cluster representatives.

1. Put all documents into one cluster.
2. Split one cluster (the worst) in t new.
3. Repeat 2 until a stopping criterion is reached.

Figure 3.3: Divisive Clustering, usually $t = 2$

3.2 Text Representation

In most document clustering implementations the vector space model of the previous chapter provides the document representation and the similarity measure. The objections to the vector space model become even more severe here as we are actually hoping to reflect similarity in content between documents. Still results show that clustering may be useful.

The collocation effect (described in section 2.4) probably plays an important part in clustering as the objects compared (documents and/or clusters) contain many words. This also indicates that the use of other methods for finding related words based on cooccurrences does not improve clustering that much; clustering (especially partitioning clustering) uses the cooccurrence information.

Groups of Documents

In the vector space model one text is represented by a vector. To represent a group of texts, a cluster for instance, the centroid is often used. The centroid for a group D is:

$$c = \frac{1}{|D|} \sum_{d \in D} d, \tag{3.1}$$

where the sum is component wise and $|D|$ is the number of texts in the group. When calculating the similarity of a text and a group of texts, $\text{sim}(d, D)$, the centroid is used. If normalization is not employed for the centroid the similarity becomes the average of the similarities between the text and all texts in the group.

Projection and Feature Selection

In most clustering algorithms similarity is repeatedly calculated between objects (documents and/or clusters). Thus if the similarity calculation can be made faster the total execution time can decrease significantly. The time for the similarity calculation is typically proportional to the smallest number of terms in the two objects being compared. Thus to shorten it one may try to reduce the number of terms in the representation.

In (Dhillon *et al.*, 2003) two term (feature) selection techniques are investigated. Using these the authors get similar results in quality using significantly fewer terms than with a full representation. The first is based on the variance of frequency of the terms in the texts and the second on term cooccurrence.

In (Schütze and Silverstein, 1997) the process of reducing the number of terms is called *projection*, the vector space is projected down onto a new space with fewer dimensions. The authors distinguish between local and global projection. Local projection is carried out on each document on its own, while global projection considers all documents at the same time.

The simplest projection is truncation, i.e. removing terms with low weight from the representation. This could be done on the text set as a whole, on documents or on clusters. Good results are presented for clustering using truncation of cluster centroids, which is a kind of local projection. Global projection using LSA (see Section 2.4) is also investigated. The method is to project the text set to the space consisting of the eigenvectors with the biggest eigenvalues of the original word-by-document matrix.

The similarity measure is not as important for clustering as for search engines; the order in similarity of the documents within a cluster is not as important as to which cluster they belong. Small changes in similarity definitions may change what cluster documents at the boundary of clusters belong to, but that is often not very clear anyhow. (Schütze and Silverstein, 1997)

Apart from the time aspect, projection and term selection reduces the amount of memory needed. However, the few techniques discussed here all start with the full representation. Using RI (see Section 2.4) one could perhaps circumvent this.

3.3 Some Applications of Document Clustering

The *cluster hypothesis* proposes that "closely associated documents tend to be relevant to the same request" (Van Rijsbergen, 1979), i.e. similar documents are believed to be relevant to the same queries put to a search engine. This has made many researchers believe that a credible clustering could make search time shorter as clusters could be retrieved instead of documents.

These beliefs are partially argued against in (Hearst and Pedersen, 1996). Similar documents are probably relevant to the same requests, but that does not mean that a clustering of the entire text set in advance can take all future queries into consideration. Therefore the authors argue for clustering after the ordinary search engine retrieval, and they show through some experiments with their Scatter/Gather system that this indeed can improve the search result quality.

Zamir et. al. (Zamir *et al.*, 1997; Zamir and Etzioni, 1998) has shown an efficient way to cluster web search engine results. The search engine Vivisimo[1] uses clustering on retrieved documents.

The Scatter/Gather system (or any clustering method) has also been proposed for browsing document collections (Cutting *et al.*, 1992) (which certainly may be search results as well (Hearst and Pedersen, 1996)). A document collection is presented to a user as a set of clusters. The user may mark one or several clusters for further investigation and request that these are reclustered giving a more fine tuned grouping. In this way the user may iteratively and interactively explore the collection and get an overview of its content as well as find particular themes that appear in it. This kind of tool is also valuable in text data mining (Hearst, 1999).

Many of the statistical methods that are applied on the word-by-document matrix are closely connected. Text clustering may be used for dimension reduction in the same way as LSA (see Section 2.4); the cluster centroids may serve as a basis onto which the texts can be projected. This method gives similar results as LSA, but is more computationally efficient (Dhillon and Modha, 2001).

3.4 Evaluation

It is very hard to make a reliable evaluation of clustering results, partially since what is a good partition of a text set is very subjective. It depends on the text set, the purpose of the partition and not least, the person that wants to utilize the partition. Still, we need some way to automatically evaluate clustering. It would be too time and money consuming to do manual evaluation and even if such an evaluation would be made it would only reflect the opinions of one or a few persons. In the long run a text clustering tool probably need to be very flexible in order to meet demands from different users.

It is common to distinguish between intrinsic and extrinsic evaluation. External quality measures use external knowledge. Many of these compare the clustering to an other partition. Internal quality measures use no external knowledge, but are based on what was available for the clustering algorithm.

For the measure definitions in the following two subsections consider a text set with n texts. Let C be a clustering with γ clusters, c_1 through c_γ. By n_i we mean the number of texts in cluster c_i ($\sum_{i=1}^{\gamma} n_i = n$). Similarly, let K be a categorization with κ categories, $k^{(1)}$ through $k^{(\kappa)}$ and let $n^{(j)}$ denote the number of texts in category $k^{(j)}$. Also, let the γ by κ matrix M describe the distribution of the texts over both C and K; that is $m_i^{(j)}$ is the number of texts that belong to c_i and $k^{(j)}$.

[1]http://vivisimo.com/

Internal Measures

When clustering a set of objects using a representation it is assumed that this representation expresses those aspects of the objects that are of interest. Hence, it is reasonable to evaluate the result by looking at how cohesive the clusters are and how well separated they are using the representation and the similarity measure. This has a very close connection to the criterion functions (see Section 3.1) as these are defined to drive the algorithms to clusterings with cohesive and/or well separated clusters. The criterion function of the K-Means algorithm (as presented in Section 3.1) is (Zhao and Karypis, 2004):

$$\Phi(C) = \sum_{c_i \in C} \sum_{d \in c_i} \mathrm{sim}(d, c_i), \tag{3.2}$$

where d is a document. If the K-Means algorithm iterates until no objects change clusters it reaches a local optimum, where there is no gain in $\Phi(C)$ by moving any text to another cluster.

The criterion function $\Phi(C)$ measures the cohesiveness, or the *intra similarity*, of the clusters in the clustering and can be used as an evaluation measure. Similarly, it is possible to define the *inter similarity* of the clusters:

$$\Phi_{\mathrm{inter}}(C) = \sum_{1 \le i < j \le \gamma} \mathrm{sim}(c_i, c_j). \tag{3.3}$$

This can also be used as a criterion function (which one probably would want to minimize).

Internal measures are suitable for comparisons of different clusterings of the same text set, if these are produced using the same representation. A comparison of different clustering algorithms may be unfair if one of them uses the measure as its criterion function and the other is not.

In (Zhao and Karypis, 2004) several criterion functions for partitional algorithms are evaluated. Strehl (Strehl, 2002) discusses three different internal quality measures.

External Measures

That a clustering is deemed good evaluated with internal measures shows that the algorithm succeeded with respect to the text representation. Whether it is actually useful is a much harder question. To get a little bit closer to an answer to that question one can compare the clustering with a trusted manual categorization. This is what external measures do. Thus they are dependent on the manual categorization – if it is odd in any sense the evaluation becomes effected. Several categorizations may help in making the evaluation more trustworthy (see Section 4.3 and Paper II (Rosell *et al.*, 2004)).

Evaluation using external measures is a reasonable way to decide on which representation to use. The representation that, used in a clustering algorithm, produces the best clustering compared with a manual categorization is probably the one to use.

For clustering precision, p, and recall, r, (see Section 2.3) compare each cluster c_i to each class $k^{(j)}$:

$$p_i^{(j)} = \frac{m_i^{(j)}}{n_i}, \quad r_i^{(j)} = \frac{m_i^{(j)}}{n^{(j)}}, \tag{3.4}$$

To present these measures for all clusters and classes would be too much information. More reasonable is the maximum precision of each cluster, the *purity* (Strehl, 2002):

$$\rho_i = \max_j \{p_i^{(j)}\} \tag{3.5}$$

This may be motivated in the context of using clustering as an aid for classifiers, classifying clusters (based on cluster descriptions) rather than single texts. The weighted average purity over all clusters can be used as a measure of quality of the whole clustering:

$$\rho = \sum_i \frac{n_i}{n} \rho_i = \frac{n_{\max}}{n}, \tag{3.6}$$

where $n_{\max}$ is the number of texts in the entire set that are part of a cluster, where the number of texts from their classes is greater than the number of texts from the other classes.

For each cluster class pair we can also give the F-measure, $F_i^{(j)}$. In (Larsen and Aone, 1999) the F-measure for a hierarchical clustering was defined as follows. The F-measure for each class over the entire hierarchy is:

$$F^{(j)} = \max_i F_i^{(j)}, \tag{3.7}$$

where the maximum is over all clusters at all levels. The F-measure of the whole clustering hierarchy is:

$$F = \sum_j \frac{n^{(j)}}{n} F^{(j)}. \tag{3.8}$$

The average is made over the classes rather than the clusters as for the purity. The F-measure tries to capture how well the clusters at the best match the categories. Purity tries to capture how well the clusters on average match the categories.

The precision $p_i^{(j)}$ is the probability that a text drawn at random from cluster c_i belongs to category $k^{(j)}$. The *entropy* (Steinbach *et al.*, 2000) for a cluster c_i is defined:

$$H(c_i) \quad = \quad -\sum_j p_i^{(j)} \log(p_i^{(j)}). \tag{3.9}$$

Unlike precision, recall and the F-measure, entropy take all categories into account. The entropy is maximized when the number of texts from all categories are equal: $H_{max} = \log(\kappa)$, so a normalized entropy (see (Strehl, 2002)) (taking values in $[0, 1]$) for a cluster c_i is:

$$\tilde{H}(c_i) = H(c_i)/\log(\kappa). \tag{3.10}$$

To get a measure on the entire clustering the weighted sum^2 can be used:

$$H(C) = -\sum_i \frac{c_i}{n} H(c_i). \tag{3.11}$$

The *information gain* (Bradley and Fayyad, 1998), IG, compares the weighted average entropy of the clustering to the entropy of the entire text set over the categories, H_{tot}:

$$IG(C) = H_{\text{tot}} - H(C), \tag{3.12}$$

$$H_{\text{tot}} = -\sum_j \frac{k^{(j)}}{n} \log(\frac{k^{(j)}}{n}). \tag{3.13}$$

Rather than averaging over the clusters one may calculate the *mutual information* of the clustering and the categorization (Strehl *et al.*, 2000):

$$MI(C, K) = \sum_i \sum_j p_i^{(i)} \log(\frac{p_i^{(j)}}{p_i p^{(j)}}) \tag{3.14}$$

$$= \sum_i \sum_j \frac{m_i^{(j)}}{n} \log(\frac{m_i^{(j)} n}{n_i n^{(j)}}), \tag{3.15}$$

where $p_i = n_i/n$ and $p^{(j)} = n^{(j)}/n$ is the probability of a text drawn at random from the entire set belongs to cluster i and category j respectively.

A theoretical tight upper bound for the mutual information is $MI_{max}(C, K) = \log(\kappa\gamma)/2$, the mean of the theoretical maximal entropy of the clustering and the categorization compared to each other. By dividing the mutual information by this a normalized measure is obtained. (Strehl, 2002)

^{2}Similarly for the normalized entropy

Part II

Contributions

Chapter 4

Contributions

This is a short overview of the papers in this thesis, summarizing the content by topic. In the experiments presented in this thesis we have used the simple clustering algorithms K-Means and Bisecting K-Means with the vector space model for representation. The focus has not been the clustering algorithms. Our main interest is text clustering as a tool for exploration of open answers in medical questionnaires (in Swedish), as discussed in Section 1.4. The influence of different clustering algorithms remains to be investigated.

Our work can be divided into two areas: representation in the vector space model using Swedish language technology tools (Section 4.2) and evaluation of clustering results (Section 4.3). First we present the text sets we have used in our experiments.

4.1 Text Sets

The results of clustering are influenced by many things. One of the main contributors is of course the text set. We have used the following text sets in our investigations:

- A few sets of newspaper articles (DN, DN150, A1, A2, A3, A150) from the KTH News Corpus (Hassel, 2001). The sections of the newspapers provide a categorization with five categories for each set. (See paper I)
- A set of open answers to one question about occupation in a questionnaire in the Swedish Twin Registry[1] (Occ). Each answer was manually classified following two established hierarchical occupation classification systems: one with 11, 28, 114, 361, 969 categories at the different levels and one with 12, 59, 288 categories. (See paper II)
- A set of medical papers from Läkartidningen[2] (Med). Each paper have one

[1]http://www.meb.ki.se/twinreg/index_en.html
[2]http://www.lakartidningen.se/

or more MeSH-terms (The Medical Subject Headings[3]) assigned to it. Using these four different unambiguous categorizations were constructed, three with 15 categories and one with 814 categories. (See paper III)

Table 4.1 gives some statistics for these sets. All counts are after removing stop-words and splitting compounds into their components, not keeping the original word. We have also applied stemming before counting the average number of words per document and how many of the documents the words appear in on average.

The text set AB was used in paper III and is in fact the same set as A1. The differences in the statistics are due to different preprocessing in the two papers (paper I and III).

Text set	docs	words	stems	stems/ doc	docs/ stem
DN	6 954	652 324	15 877	59.53	26.07
DN150	3 485	466 151	13 506	82.78	21.36
A1	2 500	140 920	7 096	34.42	12.13
A2	2 500	140 886	7 192	34.66	12.05
A3	2 500	142 930	7 261	35.27	12.14
A150	5 325	795 290	16 307	85.73	27.99
Occ	43 341	453 105	6 179	9.27	65.01
AB	2 500	119 401	5 896	27.04	11.47
Med	2 422	4 383 169	26 102	316.85	29.40

Table 4.1: Text set statistics

4.2 Clustering in Swedish

Papers I and III are mainly concerned with the impact of using Swedish language specific tools when building the vector space representation. Stemming is shown to improve clustering results moderately in paper I. The main advantage of stemming is probably the reduction of the number of different terms (although stemming naturally becomes more and more important with decreasing number of terms in each text).

Both papers show the significance of splitting Swedish solid compounds when building the representation. It is important not to split compounds that have a different meaning than what is suggested by the components on their own. A stoplist for such compounds was manually constructed for newspaper articles in paper I. That no similar effort was made for medical text in paper III might partly explain the lack of improvement for the medical papers.

[3]http://www.nlm.nih.gov/mesh/meshhome.html

In paper III we investigated the use of phrases in the representation in two domains: newspaper articles and medical papers. We hoped to find that phrases would improve clustering results and that this improvement would be greater for the medical papers as they have many significant phrases. The results were worse when using phrases than when using only the ordinary word representation. Clustering using phrase representation performed better on the medical papers than on the newspaper articles. This at least shows the importance of adapting the representation to each particular domain.

4.3 Clustering Evaluation

Paper II presents two ways of using two different classifications of the same text set for clustering evaluation. Most extrinsic evaluation methods compare a partition (normally a clustering) with an other partition (the categorization). By comparing two trusted classifications we get values on all measures that indicate what is a good result, something that is impossible to say having only one classification. The result for a clustering with respect to one of the classification may later be compared to the comparison of the two classifications (comparing comparisons).

The other way to exploit the situation with two classifications is to use the *kappa statistics* (Eugenio and Glass, 2004). Here we consider the clustering and one of the classifications as classification attempts following the second classification. The kappa coefficient measures the agreement of two classification attempts.

The three papers show increasingly elaborate evaluation methods. A result as measured by any quality measure does not give any information in isolation. A reference is needed. In paper I and III we compared results when changing representations. In the later we also present worst and best case results, as well as random clustering results, which gives further perspective. Further we are able to compare results on the two different text sets by using the simplest representation as a reference on both.

In paper II we present the results in comparison with both classifications. We also present the result for a "random clustering". We must confess that this is an unfortunate term, as what we mean here is simply the "clustering" consisting solely of one cluster.

4.4 My Part

Here is a short description of what I have done of the work on the three papers (see Section 1.5):

Paper I. The idea to investigate the impact of Swedish language technology tools on text clustering was Viggo Kann's. I built a clustering tool, made the experiments and realized the need for a stoplist for some solid compounds. I wrote the paper.

Paper II. This paper is the beginning of the cooperation with KI within the Infomat project. We want to use clustering for exploration of questionnaires. In this paper we try to prove that this is useful. Jan-Eric Litton provided the open answers with classifications. The idea of comparing comparisons is mine. I implemented and made the experiments. I wrote the paper with assistance from Viggo Kann and Jan-Eric Litton. Some formulations are Henrik Eriksson's.

Paper III. This paper is a cooperation with master student Sumithra Velupillai. The interest in phrases and in particular for medical texts is collective within the Infomat project. Sumithra found the text set, categorized it and extracted phrases. I built the different representations and made the experiments. I wrote the paper with assistance from Sumithra.

4.5 Future Work

We are already working on the natural continuation of paper II; to use clustering as a tool for exploration in general and for open answers in epidemiological questionnaires in particular. In this respect we will look at automatic assessment of clusters to be able to present only potentially interesting clusters to a user. We are also interested in automatic cluster description extraction and visualization to make such a tool usable.

Bibliography

R. Baeza-Yates and B. Ribeiro-Neto. 1999. *Modern Information Retrieval.* Addison-Wesley. ISBN 0-201-39829-X.

Michael W. Berry, Zlatko Drmac, and Elizabeth R. Jessup. 1999. Matrices, vector spaces, and information retrieval. *SIAM Rev.*, 41(2):335–362. ISSN 0036-1445.

P. S. Bradley and U. M. Fayyad. 1998. Refining initial points for K-Means clustering. In *Proc. 15th Int. Conf. on Machine Learning*, pages 91–99. Morgan Kaufmann, San Francisco, CA. URL `http://citeseer.ist.psu.edu/bradley98refining.html`.

S. Brin and L. Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117. URL `http://citeseer.nj.nec.com/brin98anatomy.html`.

J. Carlberger, H. Dalianis, M. Hassel, and O. Knutsson. 2001. Improving precision in information retrieval for Swedish using stemming. In *Proc. 13th Nordic Conf. on Comp. Ling. – NODALIDA '01.*

A. Chen and F. Gey. 2003. Combining query translation and document translation in cross language retrieval. In *CLEF 2003.* URL `http://www.clef-campaign.org/2003/WN\_web/05.pdf`.

D. R. Cutting, J. O. Pedersen, D. Karger, and J. W. Tukey. 1992. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proc. 15th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval.* URL `http://citeseer.nj.nec.com/cutting92scattergather.html`.

H. Dalianis. 2005. Improving search engine retrieval using a compound splitter for Swedish. In *Proc. 15th Nordic Conf. on Comp. Ling. – NODALIDA '05.*

S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407.

I. Dhillon, J. Kogan, and C. Nicholas. 2003. *Survey of Text Mining*, chapter Feature Selection and Document Clustering. Springer-Verlag New York, Inc., Secaucus, NJ, USA. ISBN 0387955631.

I. S. Dhillon and D. S. Modha. 2001. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1-2):143–175. ISSN 0885-6125.

B. Di Eugenio and M. Glass. 2004. The kappa statistic: A second look. *Comp. Ling.*, 30(1):95–101.

W. B. Frakes and R. Baeza-Yates. 1992. *Information Retrieval Data Structures & Algorithms.* Prentice Hall. ISBN 0-13-463837-9.

D. J. Hand, H. Mannila, and P. Smyth. 2001. *Principles of data mining.* MIT Press, Cambridge, MA, USA. ISBN 0-262-08290-X.

M. Hassel. 2001. Automatic construction of a Swedish news corpus. In *Proc. 13th Nordic Conf. on Comp. Ling. – NODALIDA '01.*

M. A. Hearst and J. O. Pedersen. 1996. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *Proc. of SIGIR-96, 19th ACM Int. Conf. on Research and Development in Information Retrieval*, pages 76–84, Zürich, CH. URL `http://citeseer.ist.psu.edu/hearst96reexamining.html`.

Marti A. Hearst. 1999. Untangling text data mining. In *Proc. 37th Annual Meeting of the Association for Computational Linguistics*, pages 3–10, Morristown, NJ, USA. Association for Computational Linguistics. ISBN 1-55860-609-3.

T. Hedlund. 2002. Compounds in dictionary-based cross-language information retrieval. *Information Research*, 7(2). URL `http://InformationR.net/ir/7-2/paper128.html`.

T. Hedlund, A. Pirkola, and K. Järvelin. 2001. Aspects of Swedish morphology and semantics from the perspective of mono- and cross-language information retrieval. *Information Processing & Management*, 37(1):147–161.

A. K. Jain, M. N. Murty, and P. J. Flynn. 1999. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323. URL `http://citeseer.ist.psu.edu/jain99data.html`.

D. Jurafsky and J. H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice-Hall. ISBN 0-13-095069-6.

P. Kanerva, J. Kristofersson, and A. Holst. 2000. Random indexing of text samples for latent semantic analysis. In *Proc. of the 22nd annual conference of the cognitive science society.*

R. Krovetz and W. B. Croft. 1992. Lexical ambiguity and information retrieval. *ACM Trans. Inf. Syst.*, 10(2):115–141. ISSN 1046-8188.

B. Larsen and C. Aone. 1999. Fast and effective text mining using linear-time document clustering. In *Proc. 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*. ISBN 1-58113-143-7.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA. ISBN 0-262-13360-1.

L. Page, S. Brin, R. Motwani, and T. Winograd. 1998. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project. URL `http://citeseer.nj.nec.com/page98pagerank.html`.

M. Rosell. 2003. Improving clustering of Swedish newspaper articles using stemming and compound splitting. In *Proc. 14th Nordic Conf. on Comp. Ling. – NODALIDA '03*.

M. Rosell, V. Kann, and J. Litton. 2004. Comparing comparisons: Document clustering evaluation using two manual classifications. In *Proc. Int. Conf. on Natural Language Processing (ICON – 2004)*, pages 207–216. Allied Publishers Pvt. Ltd. ISBN 81-7764-724-5.

M. Rosell and S. Velupillai. 2005. The impact of phrases in document clustering for Swedish. In *Proc. 15th Nordic Conf. on Comp. Ling. – NODALIDA '05*.

M. Sahlgren. 2005. An introduction to random indexing. In *Methods and Applications of Semantic Indexing Workshop at the 7th Int. Conf. on Terminology and Knowledge Engineering, TKE 2005*.

M. Sanderson. 2000. Retrieving with good sense. *Inf. Retr.*, 2(1):49–69. ISSN 1386-4564.

H. Schütze and C. Silverstein. 1997. Projections for efficient document clustering. In *Proc. 20th annual int. ACM SIGIR conf. on Research and development in information retrieval*, pages 74–81, New York, NY, USA. ACM Press. ISBN 0-89791-836-3.

M. Steinbach, G. Karypis, and V. Kumar. 2000. A comparison of document clustering techniques. In *Proc. Workshop on Text Mining, 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*. URL `http://citeseer.nj.nec.com/steinbach00comparison.html`.

A. Strehl. 2002. *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*. PhD thesis, The University of Texas at Austin. URL `http://strehl.com/download/strehl-phd.pdf`.

A. Strehl, J. Ghosh, and R. Mooney. 2000. Impact of similarity measures on web-page clustering. In *Proc. Workshop on AI for Web Search, 17th National Conf. on Artificial Intelligence*. URL `http://citeseer.ist.psu.edu/strehl00impact.html`.

C. J. Van Rijsbergen. 1979. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow. URL `http://citeseer.ist.psu.edu/vanrijsbergen79information.html`.

H. E. Williams, J. Zobel, and D. Bahle. 2004. Fast phrase querying with combined indexes. *ACM Trans. Inf. Syst.*, 22(4):573–594. ISSN 1046-8188.

O. Zamir and O. Etzioni. 1998. Web document clustering: A feasibility demonstration. In *Research and Development in Information Retrieval*, pages 46–54. URL `http://citeseer.ist.psu.edu/zamir98web.html`.

O. Zamir, O. Etzioni, O. Madani, and R. M. Karp. 1997. Fast and intuitive clustering of web documents. In *Knowledge Discovery and Data Mining*, pages 287–290. URL `http://citeseer.ist.psu.edu/zamir97fast.html`.

Ying Zhao and George Karypis. 2004. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Mach. Learn.*, 55(3):311–331. ISSN 0885-6125.

Part III

Papers