KTH Computer Science
and Communication

Text Clustering Exploration

Swedish Text Representation and Clustering Results Unraveled

MAGNUS ROSELL

Doctoral Thesis
Stockholm, Sweden 2009

TRITA-CSC-A 2009:04
ISSN-1653-5723
ISRN-KTH/CSC/A--09/04-SE
ISBN 978-91-7415-251-7

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

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen i datalogi måndagen den 6 april 2009 klockan 13.15 i Kollegiesalen, F3, Kungl Tekniska högskolan, Lindstedtsvägen 26, Stockholm.

© Magnus Rosell, April 2009

Tryck: Universitetsservice US AB

Abstract

Text clustering divides a set of texts into clusters (parts), so that texts within each cluster 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 familiar ones. The main contributions of this thesis are an investigation of text representation for Swedish and some extensions of the work on how to use text clustering as an exploration tool. We have also done some work on synonyms and evaluation of clustering results.

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.

Swedish has more morphological variation than for instance English. We show that it is beneficial to use the lemma form of words rather than the word forms. 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. Our experiments show that it is beneficial to split solid compounds into their parts when building the representation.

The vector space model does not regard word order. We have tried to extend it with nominal phrases in different ways. We have also tried to differentiate between homographs, words that look alike but mean different things, by augmenting all words with a tag indicating their part of speech. None of our experiments using phrases or part of speech information 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. 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 how evaluation can be improved 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.

In some related work we have built a dictionary of synonyms. We use it to compare two different principles for automatic word relation extraction through clustering of words.

Text clustering can be used to explore the contents of a text set. We have developed a visualization method that aids such exploration, and implemented it in a tool, called Infomat. It presents the representation matrix directly in two dimensions. When the order of texts and words are changed, by for instance clustering, distributional patterns that indicate similarities between texts and words appear.

We have used Infomat to explore a set of free text answers about occupation from a questionnaire given to over 40 000 Swedish twins. The questionnaire also contained a closed answer regarding smoking. We compared several clusterings of the text answers to the closed answer, regarded as a categorization, by means of clustering evaluation. A recurring text cluster of high quality led us to formulate the hypothesis that “farmers smoke less than the average”, which we later could verify by reading previous studies. This hypothesis generation method could be used on any set of texts that is coupled with data that is restricted to a limited number of possible values.

Sammanfattning

Textklustring delar upp en mängd texter i kluster (delmängder), 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. De huvudsakliga bidragen i denna avhandling är en undersökning av textrepresentationer för svenska texter och fortsättning på arbetet med klustring som ett utforskningsverktyg. Vi har också arbetat med synonymer och utvärdering av klustringsresultat.

Textklustring, åtminstonde som det beskrivs här, utnyttjar sig av den vektorrumsmodell, som används allmänt inom informationssökningsområdet. 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.

Svenska har större morfologisk variation än till exempel engelska. Vi visar att det är fördelaktigt att använda orden i lemmaform istället för i deras böjningsformer. I svenska används och skapas hela tiden sammansatta ord. De flesta leden i dessa används även som enskilda ord och förekommer i flera olika sammansättningar. Svenska sammansatta ord motsvarar ofta fraser och öppna sammansättningar i andra språk. Våra experiment visar att det är fördelaktigt att dela upp sammansättningar när man bygger representationen.

I vektorrumsmodellen tas ingen hänsyn till ordens inbördes ordning. Vi har försökt utvidga modellen med nominalfraser på olika sätt. Vi har också försökt skilja på homografer, ord som ser likadana ut men betyder olika saker, genom att märka alla ord med deras ordklass. Inga av våra experiment visar på någon förbättring jämfört med den vanliga 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. 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 – textmängder 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.

I ett relaterat arbete har vi konstruerat en synonymordlista. Vi använder den för att jämföra två olika principer för automatisk ordrelationsutvinning genom klustring av ord.

Textklustring kan användas för att utforska en mängd texter. Vi har utvecklat en visualiseringsmetod som underlättar det och implementerat ett verktyg vi kallar Infomat. Det visar representationsmatrisen direkt i två dimensioner. När ordningen på texter och ord ändras, genom till exempel klustring, uppstår fördelningsmönster som visar på likheter mellan texter och ord.

Vi har använt Infomat för att utforska en mängd av fritextsvar om yrke från en enkät som gavs till över 40 000 svenska tvillingar. Enkäten innehåll också ett slutet svar om rökning. Vi jämförde många klustringar av fritextsvaren med det slutna svaret, betraktat som en kategorisering, genom klustringsutvärdering. Ett återkommande textkluster med hög kvalitet fick oss att formulera hypotesen att "jordbrukare röker mindre än snittet", vilket vi senare kunde verifiera genom att läsa tidigare studier. Denna hypotesgenereringsmetod kan användas på vilken mängd texter som helst som är kopplad till något annat data med begränsat antal möjliga värden.

Acknowledgements

I would like to thank the following people (and many more) for your contributions, funding, support and company. Foremost, my supervisor professor Viggo Kann, you have been a reassuring safty net, and have provided me with clever, precise and clarifying comments and questions on all my work. Also, my assisting supervisor professor Joakim Nivre, for your seemingly endless interest in all things concerning language technology, and your insightful comments on my work.

I am indebted to my co-authors: Martin Hassel, Jan-Eric Litton, and especially Sumithra Velupillai, for our unanimous and much appreciated co-operation.

The work prestented here has been funded by The Swedish Research Council (VR) through the Infomat project, GSLT (The National Graduate School of Language Technology), and KTH CSC. For this I am grateful. I would also like to express my appreciation to all the participants in the Infomat project and to all students and supervisors in GSLT. The meetings, courses, and reatreats have been very pleasurable and inspiring.

Thanks to all who are connected to our informal human language technology group, especially (including those already mentioned): Johnny Bigert, Hercules Dalianis, Ola Knutsson, Jonas Sjöbergh. Your company and the discussions in our soffgrupp ($\sim$ sofa meetings) have been invaluable. Also, I thank all other language technology researchers in Stockholm, especially: Jussi Karlgren, Magnus Sahlgren, Oscar Täckström, all at SICS, for our discussions and much appreciated company.

I have been fortunate to work at KTH CSC, and I thank you all, especially the theory group (TCS). Gustav Hast, Rafael Pass, Anna Redz are just a few of all the room mates I have had. Thank you all for your company.

I have had the pleasure to teach and thank my master thesis students and all attending my classes for the things I have learnt by teaching you. Among fellow teachers I thank: Linda Kann, Alexander Baltatzis, Ann Bengtsson, Olle Bälter, Lennart Edsberg, Isaac Elias, Gerd Eriksson, Henrik Eriksson, Vahid Mosavat, and several others. Here, at the end (?) of my own formal academic education, I would also like to thank all teachers I have had through the years at all levels in the fantastic Swedish school system.

I also thank all my friends. Most important for this thesis is Jakob von Döbeln. Your interest in and understanding of my work, as well as your personal support, has helped me a lot. Robert Zettinger has proofread my English in a couple of the papers, and helped me with the background picture on this page. Thank you also for otherwise jovially disregarding my work. Many more of my friends deserve a special mention, among them: Helena Dreber, Charlotte Eriksson, Kimmo Eriksson, Maja Fjæstad, Tove Gustavi, Maja Karasalo, Arun Kaul, Hedvig Kjellström, Olof Kjellström, Pär Lindahl, Martin Mossberg, Joakim von Scheele, Jens Waltin, Fredrik Zetterling, Maria Ögren.

Finally, I thank my family. My dear sister Maria Löfgren, her husband David, and their son, my nephew, Anton. My parents Kerstin and Örjan Rosell, for your unwavering love and support in all aspects of my life.

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 was 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 give a overview of a text set and could be used to find structure in for instance the occupation answers of the twins.

Text clustering is an application within the large and growing field of Information Retrieval, a sub-area of Language Technology. 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].

This thesis is about text clustering. My main interest is in how text clustering can be used as an exploration tool. Apart from studying this, I have investigated the impact on clustering of some aspects of Swedish and introduced a new way of evaluating clustering results. Further, I have made some work in the adjacent field of word relation modeling.

[1] See Section 1.3 and http://www.meb.ki.se/twinreg/index_en.html
[2] For instance Google, http://www.google.com/
[3] Try the search engines Clusty, http://clusty.com/, and iBoogie, http://www.iboogie.com/

1.1 General 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 may make it necessary to change the 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 nobody 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 Some Terminology

Some terms that are used throughout this thesis may require explanations. As I discuss them I will also home in on the subject we are dealing with.

This is a thesis in computer science about text clustering. It could also be said to be a thesis in language technology about the unsupervised machine learning method clustering applied to texts. Some would argue that language technology is a subfield of computer science, others that it should be called computer linguistics

and that it is a subfield of linguistics. Information Retrieval (IR)[4] deals with how to retrieve information from a large database, usually texts from a set of texts. Text clustering is definitely an IR method, but also a text (data) mining method that sometimes is used to try to find new information by combining what is stored in a (unstructured) database.

The previous paragraph demonstrates that there is no single categorization of an object that suits all perspectives.

Clustering vs. Categorization

By automatic categorization I 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.

Corpus, Clustering, Cluster, Text, Word

The objects I cluster are sequences of *words* I refer to as *texts*, *documents*, *articles* or *papers* depending on the context. In this work I have not used any of the other information that can be found in certain types of documents, i.e. all kinds of meta-data, such as types of text (titles, boldfaces, etc.) and links that can be found in hypertexts.

The constituents of a text, the words, are in fact not that easily defined. In this work I define a word (I also call it a *token* or a *word form*) to be a sequence of characters in a text that are separated by white spaces or other common delimiters, such as commas, exclamation marks and the like. This is a rather good definition for Swedish and English, but for other language it might be less appropriate. A word comes in many forms. Its basic (or lexical) form is the *lemma*. I also use the word *terms* to denote the words (or forms) that are the result of such processing of a text, that is made to extract the information content (like trying to find lemma forms for the words). In the representation model that will be discussed, the words can be considered the *features*, that are used to describe the *objects*, i.e. the texts.

I refer to a set of texts 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 I talk about a *clustering* consisting of *clusters*, or by humans according to some agreement, in which case I talk about a *classification* or a *categorization* that consists of *classes* or *categories*. I sometimes use the terms a *grouping* consisting of *groups* or a

[4]IR is sometimes regarded as a subfield of computer science, but also could be considered being a subfield of language technology.

partition consisting of *parts* to denote either a clustering, a categorization or a classification.

Clustering Algorithms, Representation, and Similarity

The motive for creating a partition, be it a categorization or a clustering, can be of several different kinds. It could, for instance, be beneficial to have a set of texts divided into groups of different levels of readability, so that you can choose which text to read depending on how good a reader you are. Texts can also be categorized into genres in several ways.

There are many different clustering algorithms. Most of them need to know the (dis)similarity between the objects (texts). Some of them need a representation for each object, and a definition of similarity, so they can calculate it when necessary. How the objects are represented and the definition of similarity differ between applications. It is usually convenient to build a representation and define similarity in terms of it. There are many ways to achieve this. If, for instance, the objects can be represented as points in a n-dimensional vector space, dissimilarity could be defined as the distance between them.

When given a set of objects and the similarity between them, a clustering algorithm outputs a partition that tries to satisfy some criteria. It could be that the objects in each cluster should be as similar as possible. However, in order for the result to be useful at all, we must have reasons to believe that the similarity definition reflects the similarity between the actual objects.

This work focuses on content groups, that is groups of texts that are similar in content. So how do we represent texts and define (content) similarity between them? In many Information Retrieval applications texts are represented by the words that appear in them and similarity between two texts is defined by considering the words that appear in both of them. Basically, two texts that share many words are considered more similar than two that share fewer words. This model is employed with great success in search engines and I use it here as well.

1.3 Research Environment

The work presented here has been performed at KTH CSC[5] (The School of Computer Science and Communication) at KTH (Kungliga Tekniska Högskola, The Royal Institute of Technology), within the informal human language technology group[6] at KTH CSC. It was funded by three sources:

- The Infomat project (first half)
- GSLT (second half)

[5] http://www.kth.se/csc/
[6] http://www.csc.kth.se/tcs/humanlang/

- KTH CSC (a few extra months)

I have also had the opportunity to participate in the KEA[7] project, so far primarily in discussions.

The Infomat Project

The first half of my PhD studies was funded by Vetenskapsrådet (The Swedish Research Council) through the project Infomat[8] (2003–2005). It was a cooperation with the department of Medical Epidemiology and Biostatistics (MEB) at Karolinska Institutet (Swedish Medical University, KI), Stockholm, Sweden. Infomat is an abbreviation for *Swedish information retrieval with language technology and matrix computations*, which summarizes what we were interested in.

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[9], 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 were interested in the use of language technology tools to aid such analyzation. In particular, we wanted to investigate the use of text clustering as a tool for exploration of this kind of texts.

GSLT

The second half of my PhD studies has been funded by GSLT – The Swedish National Graduate School of Language Technology, one of sixteen national graduate schools established by the Swedish government in 2001. It has been rewarding to meet other PhD students and senior researchers within my field at all the different arrangements that have been organized within GSLT.

During this period I continued the work we started in the Infomat project.

Human Language Technology Group and Tools

The human language technology group at KTH CSC consists of researchers from a few different groups at KTH. The research focus of our group is, as formulated by professor Viggo Kann:

- To develop efficient, resource lean and evaluable (preferably automatically) natural language processing (NLP) methods, especially for Swedish, and

[7] http://researchprojects.kth.se/index.php/kb_7795/io_9851/io.html
[8] http://www.nada.kth.se/theory/projects/infomat/
[9] http://www.meb.ki.se/twinreg/index_en.html

- to use these methods to develop useful and freely available NLP resources and tools.

Among the tools[10] that have been developed in the group, the following have been used in this thesis:

- Stava – a spell checking program, and a compound splitter based on it.
- Granska – a grammar checking program, and in particular the tagger it uses, the Granska Tagger – a part-of-speech tagger for Swedish.
- JavaSDM – a Java package for Random Indexing.

1.4 Research Goals

During my master thesis work I read about the Scatter/Gather text clustering system (Cutting *et al.*, 1992) and understood the potential of text clustering used as an exploration tool. This idea is the motive for all my work.

In the present work the research focus of the human language technology group, as described in the previous section, have been applied to research on text clustering with the aim that it should be used as an exploration tool. It is thus application driven research, but the results are also interesting and valid on their own.

In the papers (see the next section) I (and co-workers) have followed the first part of the research focus. The second part is met by the production of the People's Dictionary of Synonyms (see Section 4.4 and paper IV) and the implementation of the freely available program Infomat[11] (see Section 4.5 and paper VIII). Infomat is named after the project, and I consider it an abbreviation of *information matrix*. It is a comparatively easy-to-use visual exploration tool, that also serves as an experimentation system for further related research.

1.5 Outline

This thesis is what in Swedish is known as a *sammanläggningsavhandling*[12], which means that it consists of a few papers with an introductory part. It is based on my licentiate thesis (Rosell, 2005), and some formulations in the papers have been reused in the main text.

[10] http://www.csc.kth.se/tcs/humanlang/tools.html

[11] http://www.csc.kth.se/tcs/projects/infomat/infomat/

[12] This word is a good example of a Swedish solid compound, something that is discussed a great deal in Paper I. 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*, indicating (in my understanding) that this document treats a subject rather thoroughly.

The rest of the thesis is divided into three parts. The first gives a brief introduction to Information Retrieval in general and Text Clustering in particular. Part two summarizes the contributions of the papers. It also includes an appendix with some additional experiments. The last and third part contains the following 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)

Paper IV. Viggo Kann and Magnus Rosell: "Free Construction of a Free Swedish Dictionary of Synonyms", NoDaLiDa 2005, Joensuu, Finland, 2005. (Kann and Rosell, 2005)

Paper V. Magnus Rosell, Sumithra Velupillai: "Revealing Relations between Open and Closed Answers in Questionnaires through Text Clustering Evaluation", LREC'08, Marrakesh, Morocco. (Rosell and Velupillai, 2008)

Paper VI. Magnus Rosell: "Part of Speech Tagging for Text Clustering in Swedish", submitted. (Rosell, 2009b)

Paper VII. Magnus Rosell, Martin Hassel, Viggo Kann: "Global Evaluation of Random Indexing through Swedish Word Clustering Compared to the People's Dictionary of Synonyms", submitted. (Rosell *et al.*, 2009)

Paper VIII. Magnus Rosell: "Infomat – Visualizing and Exploring Vector Space Model Data Matrixes", submitted. (Rosell, 2009a)

Part I

Background

Chapter 2

Information Retrieval

Information Retrieval (IR) is a large field within Natural Language Processing. 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 text 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 (Manning *et al.*, 2008; 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 is assigned weights (values) in the indices (dimensions) based on what words appear in them. These weights can be thought of as modeling how important the corresponding word is deemed to be to explain the content of the text. Each weight is dependent on whether (and how often) the word appears in the text and in the entire set. Texts whose vectors are close to each other in this space are considered to be similar in content.

2.1 Text Representation

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

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

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

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-document matrix*[3], with elements $w_{i,j}$.

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

$$w_{i,j} = \mathrm{tf}_{i,j} \cdot \mathrm{idf}_i. \tag{2.2}$$

There are many versions of tf and idf. These are two simple examples:

$$\mathrm{tf}_{i,j} = \frac{n_{i,j}}{\sum_i n_{i,j}}, \tag{2.3}$$

$$\mathrm{idf}_i = \log(n/n_{\mathrm{word}(i)}), \tag{2.4}$$

where $n_{i,j}$ is the number of times word i appears in document j, and $n_{\mathrm{word}(i)}$ is the number of documents that word i appears in. The logarithm in idf is used because it has been found empirically that a function growing faster would grow too fast with decreasing $n_{\mathrm{word}(i)}$[4].

The reason for calculating these weights and constructing the word-document matrix is to define similarity (see the next section). However, the weight can be thought of as modeling how important each word is to describe the content of each document. The term frequency deems words that appear frequently in a document more important than words that appear infrequently. The inverse document frequency models the distinguishing power of a word in the text set; the fewer documents that contain a word the more information it reveals about those documents among the others in the set.

There are many different weighting schemes. Most of them utilize both *local* and *global* information, which corresponds to tf and idf above.

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 α between the query q and a text d_j:

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

In the basic search engine model, the texts are returned to the user in order of similarity to the query; they are *ranked*[5].

[3] It is often called the *term-document-matrix*, but we prefer *word* over *term*.

[4] This is related to Zip's law. See Manning and Schütze (1999) for an introduction.

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

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 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 therefore made on small controlled text sets like those provided by TREC[6], CLEF[7], 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 engine. 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:

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

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

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:

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

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

[7]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 higher than one recall is considered more important, and precision when it is set lower 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 Extensions and 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 its content (for modeling text content similarity). 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, not the content in itself.

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 extensions and 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-document matrix) may be built using only open word classes. Among the open word classes nouns are the ones that are intuitively connected to content. A representation not containing all words is 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 can still used for sets of words). Williams *et al.* (2004) compare a few different representations of sequences of words.

Term Normalization

Words come in many different forms. In the most naive version of the vector space model all character strings separated by delimiters (such as space, commas, exclamation marks, etc.)[8] are used in the representation. *Term normalization* is the process of removing superficial differences that prevent words in different forms, that convey the same information, to be matched. One part of it is to treat capitalization in a suitable and consistent way. More interesting is how to treat the morphology of the language(s) that are used.

If you are searching for *cars* you probably also want to get all texts with the singular form *car* in them as well. 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 word the same stem. In addition it may also give derived words the same stem as the word 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.

For languages with richer morphology than English the need for stemming, lemmatization, and other morphological analysis is even bigger.

Swedish

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

[8] We will not discuss how to remove meta data at all.

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).

Swedish is rather rich in morphology. Stemming improves search result precision by 15 % and recall by 18 % for Swedish (Carlberger *et al.*, 2001). There are also a lot of studies in CLEF that include Swedish, several of which report improvements using morphological analysis.

Document Structure and Meta-Data

The document can contain additional information that may be used when indexing. One possible source of information is the document structure: words appearing in headings and boldface are probably more important than other words. Texts with meta-data, such as web pages, can contain even more information. Even meta-data tags that 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[9] 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.

2.5 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 (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

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

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 co-occurrences. Manually constructed theasuri with word relations, such as WordNet[10], 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.

Word Relation Models

So called *word space models* (see among others (Deerwester *et al.*, 1990; Schütze, 1993; Landauer and Dumais, 1997; Kanerva *et al.*, 2000; Sahlgren, 2006)) are vector space models that aim at defining similarity or relatedness between words. A broader, and perhaps more appropriate, name for these models could therefore be *word relation models*.

Word space models map words to vectors in a multidimensional space by extracting statistics about the context they appear in from a large sample of text. Words that thus become represented by similar vectors (as measured by for instance a similarity measure such as the cosine measure) are considered related. What this (meaning) relation could be referred to in ordinary (human) semantics is not obvious. It may capture something like synonymy, but may as well regard for instance antonyms, and a hyponym and its hyperonym as highly related.

The word-document-matrix described in the previous sections can be used as a word space model. The row vectors corresponding to two words can be compared by for instance the cosine measure. Words are thus considered related if they occur in the same texts.

It also possible to compare words based on which other words appear close to them. A word-word-matrix (or a word-co-occurrence-matrix) contains counts that indicate how many times each pair of words occur in the same context (the same document, paragraph, or any other text segment that is decided in advance). Row or column vectors can be compared (through for instance the cosine measure) to decide whether the corresponding words are related. Words are thus considered related if they appear together with similar words.

Relations between words based on their contexts can be divided into two categories (Sahlgren, 2006):

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

- Two words have a *syntagmatic relation* if they appear together in the same contexts.
- Two words have a *paradigmatic relation* if they appear in similar contexts, but do not co-occur.

Word space models can be constructed in attempts to capture either of these two relations. Using the word-document-matrix can be considered an attempt to capture syntagmatic relations, and using the word-word-matrix an attempt to capture paradigmatic relations.

Word space models are evaluated using different lexical resources, such as manually created thesauri and synonym tests, see Sahlgren (2006). The following two sections give brief descriptions of two word relation (or word space) models.

Latent Semantic Analysis

The word-document-matrix is very sparse. Latent Semantic Analysis (LSA) (Landauer and Dumais, 1997) reduces the number of dimensions used to represent each word[11] by means of a singular value decomposition (SVD) (Berry *et al.*, 1999).

By projecting the word representations onto the subspace defined by the eigenvectors with the highest eigenvalues from the SVD an optimal dimensionality reduction is achieved (as measured by least square distance). This leads to a more compact representation and noise is removed from the data. It also 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).

It is also possible to apply SVD on the word-word-matrix. This gave excellent results on a synonymy test in (Rapp, 2003).

Random Indexing

LSA is computationally heavy and starts with the full word-document-matrix. Random Indexing (RI) (Kanerva *et al.*, 2000; Sahlgren, 2006) is a much faster alternative that uses less memory as it does not start from the full word-document-matrix (or word-word-matrix).

RI associates each word with a unique extremely sparse *random vector*, all with the same predefined dimensionality n, usually a few thousands. The random vectors only contain $2t$ ($t \ll n$) randomly selected non-zero elements, half of which are assigned one (1), and half minus one (-1). For each word a *context vector* is created by adding the random labels of words that appear in its context. Words that appear in similar contexts get similar context vectors. Two words are considered related

[11] At the same time texts can be represented in the same reduced space.

if their context vectors are deemed similar by a similarity measure, like the cosine measure.

The context of a word can be defined in many ways. If we choose it to be whole texts, the matrix consisting of the context vectors corresponds to a random projection (to a random subspace) of the words represented by the word-document-matrix. When the original data matrix is sparse and the projection is constructed well the distortion in the similarities are small (Kaski, 1998).

We can also let the context be, for instance, a few words surrounding a center word. In this version the method runs through the texts sequentially word by word focusing on the current center word. For each word the random vectors of all the surrounding words are added to the context vector. The addition may be either constant (all random vectors simply added to the context vector) or weighted depending on the distance between the center word and the particular surrounding word, d. Commonly used is exponential dampening: 2^{1-d}.

Using surrounding words as contexts corresponds to a random projection of a word-word-matrix with a similar definition of context. Random Indexing can thus approximate both the word-word- and the word-document-matrix without having to start from the full matrix. The approximation is very efficient and leads to good results, see Sahlgren (2006).

Chapter 3

Text Clustering

Recall from the introduction that the objective of clustering is to partition an unstructured set of objects into clusters (parts). 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; Behrkin, 2006).

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 the similarity measure (and the representation). Some clustering algorithms do not need the object representation, but starts from the similarities between all objects. However, the similarities have to be provided in some way, and that is often most conveniently calculated using a similarity measure defined on a representation.

The input to a clustering algorithm can be viewed as a graph consisting of vertices corresponding to the objects, and edges corresponding to the similarities. Clustering can be defined as a minimization/maximization problem on this graph in several ways. Many of these are NP-hard problems.

Clustering is an unsupervised learning method. The result (the clustering, the partition) is based solely on the similarity between the objects (via the object representation) 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 many cases exist many relevant clusterings of a set of complex objects.

In text clustering the objects are texts or documents. These could of course be grouped in many ways. We are primarily interested in clustering them based

on content. For this purpose the vector space model of the previous chapter can be used. Many different clustering algorithms have been proposed and tried for document clustering, see for instance Manning *et al.* (2008).

3.1 Clustering Algorithms

The number of possible partitions of a set of n objects into k groups is the Stirling number of the second kind[1], which has an upper bound in $k^n/k!$ (Manning *et al.*, 2008). It obviously grows very fast with the number of objects, so it is not feasible to try them all. Clustering algorithms therefore restrict the number of clusterings they consider in different ways.

Clustering algorithms may be divided into groups on several grounds, see Jain *et al.* (1999). In a *hard* clustering each object is assigned to only one cluster. When objects can belong to more than one cluster (usually with a degree of membership) one talks about a *soft* or *fuzzy* clustering. Here, we only consider hard and *exhaustive* clusterings, i.e. clusterings that contain all objects in the set (Manning *et al.*, 2008).

The most common division is into *hierarchical* algorithms that produce a hierarchy of clusters, and *partitioning* algorithms that return a flat partition of the set. In the following two subsections we discuss a few basic algorithms from these two classes.

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.

The initial partition of step one can be constructed in many ways: pick k objects at random and let them define k clusters, construct a random partition of the entire set, use another clustering algorithm on a part of the set, etc. The result depends on which the initial partition is, so whenever the initial partition is constructed in a random manner the K-Means algorithm is non deterministic. To avoid the worst results several techniques can be used, see for instance Manning *et al.* (2008). The simplest is perhaps to run the algorithm several times with different randomly constructed initial partitions, and choose the best as measured by the objective function, see below.

Instead of the centroid (the mean vector, see Section 3.2) we could use some other representation for the cluster. We could let the median or a few specific objects represent the cluster. In a soft version of K-Means objects may belong to several clusters (with a degree of membership) and the cluster representation is calculated taking this into consideration.

[1] $\beta_k^{(n)} = \beta_{k-1}^{(n-1)} + k\beta_k^{(n-1)}$, and $\beta_0^{(n)} = 0$. (Råde and Westergren, 1995)

1. Construct an initial partition consisting of k clusters.
2. Calculate cluster centroids.
3. Make new clusters, one per cluster centroid. Let each text belong to the cluster with the most similar centroid.
4. Repeat from 2 until a stopping criterion is satisfied.

Figure 3.1: K-Means Algorithm

The basic stopping criterion is when no objects change clusters, or when very few change clusters between iterations. Another possibility is to stop after a predefined number of iterations, since most quality improvement usually is gained during the first iterations. The criterion can also be defined using some internal quality measure, see Section 3.3.

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, but usually can be considered bounded by some not to large number, see below). In each iteration the cluster centroids 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. It is, of course, 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 objective function, see below). In a general partitioning algorithm both merging and splitting 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 function* (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 objective function for the K-Means algorithm is discussed under *Internal Measures* in Section 3.3.

The K-Means algorithm can be derived from the EM[2] algorithm (Mitchell, 1997). To calculate the centroids of the clusters corresponds to the expectation step, and to assign texts to clusters to the maximization step. For a soft version of K-Means with the Cartesian distance used as the dissimilarity measure, the

[2]Expectation Maximization.

1. Construct one cluster for each object.
2. Merge the t most similar clusters.
3. Repeat 2 until a stopping criterion is satisfied.

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

centroids are the means of k normal distributions.

The EM algorithm is guaranteed to converge to a local optimum "under fairly broad conditions". It might take a lot of time, but it often comes close in only 5 to 20 iterations. (Hand *et al.*, 2001)

When using the EM algorithm we have to assume what kind of distributions the data is generated from[3]. The more complex distributions, the slower the algorithm becomes. When we do not know what they are it is reasonable to start with simple distributions.

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 an objective function or any internal evaluation measure, see Section 3.3.

Agglomerative clustering methods are named after how they define similarity between two clusters. The *single-link* method defines it 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 definition can be viewed as the objective function, although it is used locally for each merging and not as a global score. This determinism comes at the price of never being able to change local decisions

[3]The EM algorithm and other methods with similar explicit assumptions are often referred to as *model based clustering*.

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

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

when at higher levels in the hierarchy; that two objects are (rather) similar does not necessary mean that they should be in the same cluster.

The time complexity of the agglomerative algorithms are at least $O(n^2)$ 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)

In the divisive algorithms any clustering algorithm can be applied to split clusters (step 2). The Bisecting K-Means algorithm (Steinbach *et al.*, 2000) is a divisive algorithm 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 3.3. The time complexity[4] for the Bisecting K-Means algorithm is $O(nI \log k)$. It is lower than for the K-means algorithm as it does not compare all objects to all cluster centroids.

Partitioning vs. Hierarchical Clustering

All clustering algorithms have strengths and weaknesses. No algorithm can be best for all data sets as none considers all possible clusterings. Which one to choose depends on the particular application.

Many researchers believe that hierarchical algorithms return clusterings of higher quality than partitioning algorithms, but there is no consensus (Manning *et al.*, 2008). Zhao and Karypis (2002) argue that this belief is based on a limited number of experiments performed on low dimensional data sets. Later comparisons point in the other direction; partitioning algorithms perform better.

K-Means and Bisecting K-Means outperform agglomerative clustering in (Steinbach *et al.*, 2000). Zhao and Karypis (2002) compare several different partitioning algorithms (using different objective functions) and several different agglomerative algorithms (with different similarity measures) on high dimensional text data. The partitioning algorithms perform better, and can also be used to produce hierarchies of higher quality than those returned by the agglomerative algorithms. They argue that this is due to the possibility of errors in early mergings for the agglomerative algorithms, as described in the previous section. If a partitioning algorithm is used to split clusters in a divisive algorithm it uses information about the entire set when splitting the top level (Manning *et al.*, 2008).

[4]Obviously I will differ between bisections, but we can use a predefined maximal number of iterations.

In the work on text clustering presented in this thesis the K-Means algorithm has been used. See Section 4.1 for a discussion.

3.2 Text Representation

For text clustering the vector space model of the previous chapter can provide the representation and the similarity measure. The collocation effect (described in Section 2.5) probably plays an important part in text 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 co-occurrences will not improve clustering that much; clustering already uses the co-occurrence information (especially partitioning clustering[5]).

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)

Groups of Texts

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 the dot product is used as the measure of similarity between normalized texts and the centroids are not normalized, the similarity $\text{sim}(d, D)$ becomes the average of the similarities between the text and all texts in the group. Also, the average similarity of all texts in two groups c_i and c_j, is easily calculated as the dot product between their centroids, since[6]:

$$\begin{aligned} \text{sim}(c_i, c_j) &= c_i \circ c_j = \\ &= \frac{1}{|c_i||c_j|} \sum_{d_u \in c_i} \sum_{d_v \in c_j} \text{sim}(d_u, d_v), \end{aligned} \tag{3.2}$$

where d_u and d_v are texts.

[5] The centroids of K-Means.

[6] Here we use c_i to denote both the group and the centroid of that 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.

Some words can be filtered out, removed from the representation, based on simple statistics without affecting the similarity between texts too much. Words that appear in only one text can be removed as they do not contribute to the similarity between any texts. Words that appear in just a few texts might also be unnecessary if these texts share other words. At the other end, words that appear in many texts (and those in a stoplist of common words) can be filtered out as these make almost all texts a little similar to each other. The impact of using filtering of high and low frequency words might differ depending on the cluster setting. We present a small investigation of this in Section A.5.

There are more sophisticated ways of reducing the number of words. In (Dhillon *et al.*, 2003) two term (feature) selection techniques are investigated. Using these the authors get similar results in quality with significantly fewer terms than with a full representation. The first is based on the variance of the frequency of the terms in the texts and the second on term co-occurrence.

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 2.5) is also investigated.

Term normalization, discussed in Section 2.4, reduces the number of (different) terms by merging related ones. Stemming, lemmatization, and compound splitting can be viewed as linguistically motivated projections. The effect of using them depend on the language. In this thesis we study their effect for Swedish.

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 Random Indexing (see Section 2.5) one could circumvent this.

3.3 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 clustering, and if it is constructed as an intermediate step in further processing, or if it is to be used directly by a human.

In an interesting study by Macskassy *et al.* (1998) ten persons manually clustered a few small sets of documents retrieved by a search engine. They found that the clusterings created by any two subjects had little similarity. Although this study is very small it confirms that clustering quality is subjective. Thus a text clustering tool, that is to be used directly by humans, probably needs to be very flexible in order to meet different demands.

Still, we need some way to evaluate clustering, and we would prefer to do it automatically. It would be too time and money consuming to do manual evaluation for even a small set of experiments. Further, even if such an evaluation would be made it would only reflect the opinions of one or a few persons.

It is common to distinguish between intrinsic and extrinsic evaluation. Internal quality measures use no external knowledge, but are based on what was available for the clustering algorithm.

External quality measures takes advantage of some external context. The quality of a clustering could be assessed through some other task in which it plays a part. This could for instance be as a part in a dimensionality reduction experiment, or as an aid for a search task, see Section 3.4. Other external measures compare the clustering to another partition, such as a manual categorization. We will only consider these here.

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 $k^{(j)}$ ($\sum_{j=1}^{\kappa} n^{(j)} = n$). Also, let the γ by κ confusion matrix $M = \{m_i^{(j)}\}$ 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 $k^{(j)}$. Table 3.1 is an example of such a confusion matrix.

Internal Measures

When clustering a set of objects using a similarity definition it is assumed that this similarity 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 similarity measure. This has a very close connection to the objective functions (see Section 3.1) as these are defined to drive the algorithms to clusterings with cohesive and/or well separated clusters. The objective function of the K-Means algorithm (as presented in Section 3.1) is (Zhao

and Karypis, 2004):

$$\chi_{\text{intra}}(C) = \sum_{c_i \in C} \sum_{d \in c_i} \text{sim}(d, c_i), \tag{3.3}$$

where d is a document. If the K-Means algorithm iterates until no objects change clusters it reaches a local optimum; $\chi_{\text{intra}}(C)$ will not increase if any text is moved to another cluster.

The objective function $\chi_{\text{intra}}(C)$ 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:

$$\chi_{\text{inter}}(C) = \sum_{1 \leq i < j \leq \gamma} \text{sim}(c_i, c_j). \tag{3.4}$$

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

We approximate χ_{intra} and χ_{inter} with the following two measures as they are easily implemented and relatively fast to calculate:

$$\Phi_{\text{intra}}(C) = \frac{1}{|C|} \sum_{c_i \in C} |c_i| \cdot \text{sim}(c_i, c_i), \tag{3.5}$$

$$\Phi_{\text{inter}}(C) = \frac{1}{|C|} \sum_{c_i \in C} |c_i| \cdot \text{sim}(c_i, C). \tag{3.6}$$

If we do not normalize centroids and use the dot product as similarity measure, the later becomes $\text{sim}(C, C)$, the average similarity between all texts in the entire text set. It does not serve as a good approximation to χ_{inter}, but can be a good reference for Φ_{intra}.

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

In (Zhao and Karypis, 2004) several objective functions for partitioning 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 similarity measure. 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 the external measures we will discuss here do. Thus they are dependent on

	Entertainment	Foreign	Domestic	Economy	Sports	Total
Cluster 1	327	20	24	2	0	373
Cluster 2	48	17	25	1	467	558
Cluster 3	35	19	88	205	5	352
Cluster 4	40	46	82	317	4	489
Cluster 5	52	354	308	4	10	728
Total	502	456	527	529	486	2500

Table 3.1: Confusion matrix for a K-Means clustering of a set of 2500 Swedish newspaper articles (text set A1 in Section 4.1). Clusters vs. sections in the newspaper. Clusters one and two contain mostly articles from the entertainment and sports sections. Economy news are split between clusters three and four. Cluster five contains most of the foreign and domestic news.

the manual categorization – if it is odd in any sense the evaluation becomes effected. Several categorizations may help in making the evaluation more trustworthy (Rosell *et al.*, 2004).

Evaluation using external measures is a reasonable way to decide which of several representations 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.

External measures could be divided into two groups: those that count the number of single texts that appear in each cluster and category to compare the clustering to the categorization, and those that count the number of pairs of texts.

Most of the following measures are defined using the confusion matrix $M = \{m_i^{(j)}\}$, as defined earlier. Table 3.1 is an example; the distribution of texts over a K-Means clustering of a set of newspaper articles and the sections of the newspapers.

Counting Single Texts

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

$$p_i^{(j)} = p(k^{(j)}|c_i) = \frac{m_i^{(j)}}{n_i}, \tag{3.7}$$

which is the probability that a text drawn at random from cluster c_i belongs to category $k^{(j)}$. Similarly the recall, r:

$$r_i^{(j)} = p(c_i|k^{(j)}) = \frac{m_i^{(j)}}{n^{(j)}}. \tag{3.8}$$

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):

$$\rho_i = \max_j \{p_i^{(j)}\} \tag{3.9}$$

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:

$$\rho = \sum_i \frac{n_i}{n} \rho_i = \frac{n_{\max}}{n}, \tag{3.10}$$

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, $F_i^{(j)}$, see Section 2.3. Larsen and Aone (1999) define the F-measure for a hierarchical clustering as follows. The F-measure for each class is:

$$F^{(j)} = \max_i F_i^{(j)}, \tag{3.11}$$

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

$$F = \sum_j \frac{n^{(j)}}{n} F^{(j)}. \tag{3.12}$$

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, while the purity tries to capture how well the clusters on average match the categories.

The *entropy* (H) (Steinbach *et al.*, 2000) for a cluster c_i with regards to the categorization K is defined:

$$H(K|c_i) \quad = \quad -\sum_j p(k^{(j)}|c_i) \log p(k^{(j)}|c_i) \tag{3.13}$$

Unlike precision, recall and the F-measure, entropy take all categories into account. The entropy measures how unordered a cluster is with regards to to the categorization, and we want the cluster to be ordered. Entropy can be used to find the best single cluster in a clustering.

A cluster with texts from only one category gets entropy zero. The entropy is maximized when the number of texts from all categories are equal: $\max(H(K|c_i)) =$

$\log(\kappa)$, so a normalized entropy (see (Strehl, 2002)) (taking values in $[0, 1]$) for a cluster c_i is:

$$\tilde{H}(K|c_i) = H(K|c_i)/\log(\kappa). \tag{3.14}$$

To get a measure for the entire clustering the weighted sum can be used:

$$H(K|C) = \sum_i p(c_i)H(K|c_i), \tag{3.15}$$

where $p(c_i) = n_i/n$ is the probability that a text drawn at random from the entire text set belongs to cluster c_i. The normalized entropy, $\tilde{H}(C, K)$, can be defined similarly.

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, $H(K)$:

$$IG(K|C) = H(K) - H(K|C), \tag{3.16}$$

$$H(K) = -\sum_j p(k^{(j)}) \log p(k^{(j)}), \tag{3.17}$$

where $p(k^{(j)}) = n^{(j)}/n$ is the probability that a text drawn at random from the entire set belongs to category $k^{(j)}$. Similarly, we can define the entropy over the clustering: $H(C) = -\sum_i p(c_i) \log p(c_i)$. $H(C)$ and $H(K)$ measure how balanced the clustering and the categorization are. They are maximized for clusters (categories) of equal size.

Rather than averaging over the clusters, as in entropy, $H(K|C)$, one may consider the whole set of texts at once. Now, let $p_i^{(j)} = p(c_i, k^{(j)}) = m_i^{(j)}/n$, the probability that a text picked at random from the whole set, belongs to both cluster c_i and category $k^{(j)}$. Also, let $p_i = p(c_i)$ and $p^{(j)} = p(k^{(j)})$. The *mutual information* (MI) of the clustering and the categorization is (Strehl *et al.*, 2000):

$$MI(C, K) = \sum_i \sum_j p_i^{(j)} \log(\frac{p_i^{(j)}}{p_i p^{(j)}}) \tag{3.18}$$

$$= \sum_i \sum_j \frac{m_i^{(j)}}{n} \log(\frac{m_i^{(j)} n}{n_i n^{(j)}}). \tag{3.19}$$

In fact, the mutual information and the information gain is the same measure and the order of the clustering and categorization does not matter: $MI(C, K) = MI(K, C) = IG(K|C) = IG(C|K)$.

Strehl and Ghosh (2003) observe that there are several possibilities for normalization of the mutual information based on that $MI(C, K) \leq min(H(C), H(K))$. They use the geometric mean of $H(C)$ and $H(K)$, which is an upper bound for

	Same category	Different categories
Same cluster	tp	fp
Different clusters	fn	tn

Table 3.2: Number of pairs

$\min(H(C), H(K))$, for normalization. A normalized mutual information measure (NMI, taking values in $[0, 1]$) is thus:

$$NMI(C, K) = \frac{MI(C, K)}{\sqrt{H(C)H(K)}}. \quad (3.20)$$

Another upper bound for $\min(H(C), H(K))$ is the arithmetic mean of the entropy for the clustering and the categorization. Further we know that $\max(H(C)) = \log(\gamma)$ and $\max(H(K)) = \log(\kappa)$, so another possible normalization is[7]:

$$NMI(C, K) = \frac{2MI(C, K)}{\log(\kappa\gamma)}. \quad (3.21)$$

When comparing a clustering to a categorization the normalized mutual information punishes clusterings with larger numbers of clusters. It is a good measure when the categorization is considered entirely correct.

The normalized entropy, $\tilde{H}(C, K)$, does not take the number of clusters into account. If smaller clusters are less unordered it will reflect that. However, a cluster consisting of one text will always have entropy zero.

Counting Pairs of Texts

A clustering could also be compared to a categorization based on pairs of texts, rather than single texts as in the previous subsection. See for instance (Manning *et al.*, 2008; Halkidi *et al.*, 2001).

Each pair of texts can be either in the same or in two different groups in both the clustering and the categorization used for comparison. This gives us the four counts presented in Table 3.2. tp is for true positives, the number of pairs of texts that appear in the same cluster in the clustering *and* in the same category in the categorization. fp, fn, and tn are for false positives, false negatives, and true negatives. Using these several measures can be constructed. Considering Section 2.3 the most straightforward perhaps precision, p, recall, r, and the F-Measure, F_β:

$$p = \frac{tp}{tp + fp}, \quad (3.22)$$

[7]In earlier of the work by Strehl et. al. on cluster ensembles this version of NMI was used as they wanted balanced clusters (clusters of equal size).

$$r = \frac{tp}{tp+fn}, \tag{3.23}$$

$$F_\beta = \frac{(\beta^2+1)pr}{\beta^2 p + r}. \tag{3.24}$$

The Rand Index (RI) measures the percentage of pairs that are correctly put in the same or different group by the clustering algorithm:

$$RI = \frac{tp+tn}{tp+fn+fp+tn}. \tag{3.25}$$

$$\tag{3.26}$$

Several other measures can be constructed. For instance the Jaccard Coefficient (JC) and Folkes and Mallows index (FM):

$$JC = \frac{tp}{tp+fn+fp}, \tag{3.27}$$

$$FM = (\frac{tp}{tp+fn}\frac{tp}{tp+fp})^{1/2}. \tag{3.28}$$

All these "pair measures" and the "single text measures", of the previous section, agree in the extreme cases: a clustering identical to the categorization will get a high value, and random clusterings will get low values. However, they may differ for "rather good" clusterings. Pair measures do not directly take the distribution of the pairs over the clusters into account, and the single text measures do not directly consider pairs of texts.

There is an imbalance between the counts (tf, fp, fn, tn); in a grouping there are usually many more pairs of texts that are in different groups than there are pairs of texts in the same group. If we are interested in the pairs that are in the same category recall is the best measure.

Recall could be used in a situation where we only know that some objects (texts) are similar. In Rosell *et al.* (2009) we use this to evaluate word clusterings based on a list of synonyms. Considering this it would be possible to create an evaluation resource by asking one or several persons whether randomly selected pairs of texts are similar or not. We have not tried this.

3.4 Some Applications of Text Clustering

Text clustering exposes structure in a text set; texts that are deemed related are assigned to the same cluster, while unrelated texts are assigned to different clusters. These relations may be previously unknown. In this sense text clustering is a text (data) mining technique, as defined in (Hearst, 1999b). The result can be used directly, or combined with other knowledge, automatically or manually.

Text clustering can be useful as a preprocessing tool in several areas of language technology. Multi-text summarization, for instance, tries to present a summary of

similar texts within a larger set, and also to describe the differences between these texts. The similar texts can be extracted using clustering techniques. See for instance (Maña-López *et al.*, 2004). Columbia Newsblaster[8] automatically extracts major topics in the news each day through clustering and summarizes them (McKeown *et al.*, 2002).

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

Text clustering have merits on its own: as a tool for end users. The two following subsections discuss this.

Clustering of Search Results

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 query put to a search engine. However, there is no clear evidence that clustering is beneficial as a preprocessing step in search engines (Manning *et al.*, 2008).

The idea of using clustering for preprocessing could be argued against, as in (Hearst and Pedersen, 1996). Similar documents are probably relevant to the same requests, but it 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.

There is a lot of work on clustering after retrieval, and are several search engines that implement it, like for instance Clusty[9] and iBoogie[10]. Zamir et. al. (Zamir *et al.*, 1997; Zamir and Etzioni, 1998) has shown an efficient way to cluster web search engine results. In (Tombros *et al.*, 2002) hierarchical clustering is used. Search result clustering have also been shown to be beneficial in a user study (Käki, 2005).

Clustering Exploration

The Scatter/Gather system (or any clustering method) has also been proposed for browsing any document collections (Cutting *et al.*, 1992). The 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

[8] http://newsblaster.cs.columbia.edu/
[9] http://clusty.com/
[10] http://www.iboogie.com/

collection and get an overview of its content as well as find particular themes that appear in it.

We believe this interaction is very important to exploit the potential of text clustering as an exploration tool; the system can provide a result, but a human has to come to an understanding of it. In order for this interaction to be tolerable the system has to be fast and provide useful information.

3.5 Result Presentation

For a text clustering exploration tool to be useful the results have to be comprehendable; the content of the clusters have to be as easily grasped as possible. The following two subsections discuss two main approaches for clustering result presentation.

Textual Presentation

The text clustering system Scatter/Gather (Cutting *et al.*, 1992) presents clustering results as a list of clusters. For each cluster it displays a *cluster digest*, which consists of: *topical words* (usually words with high weight in the cluster centroid) and *typical titles* (the titles or file names of the texts that are most similar to the centroid).

The topical words that are presented for each cluster are often called a *label*, but we prefer cluster *description*, as *label* may be confused with the cluster *name*, which could be any arbitrary string (like "Cluster 1" for instance)[11]. A cluster description could in theory be a short text describing the content of the cluster (even in words not appearing in it), but usually is just a set of words that are extracted from the cluster.

Several methods for description generation utilize two criteria to find good labels. Descriptions should be both *descriptive* (representative for the cluster) and *discriminating* (set the cluster apart from the other clusters) (Kulkarni and Pedersen, 2005; Mei *et al.*, 2007).

There are several clustering algorithms that first find suitable cluster descriptions, and then form the text clusters around these. Frequent Term-Based Text Clustering (Beil *et al.*, 2002) constructs text clusters by considering texts that contain words from sets of frequent terms/words. The algorithm in (Käki, 2005) is very similar.

In Suffix Tree Clustering (Zamir *et al.*, 1997) word-n-gram phrases with information on which texts they belong to are put into a trie. The nodes of the trie represent possible text clusters that share a part of such a phrase. That part also serves as a cluster description.

As noted by Dhillon (2001) and several others a text clustering has a dual word clustering – for each text cluster a corresponding word cluster with the highest

[11] Also *labeling* could mean to assign texts to clusters, i.e. give each text the label of belonging to a certain cluster.

weighted words in the text cluster. The word clusters could be considered extensive text cluster descriptions for their respective text clusters. In (Dhillon, 2001) the text and word clusters are constructed simultaneously.

A word clustering could similarly be used to construct a text clustering. There are many ways words could be clustered, for instance via the word vectors of the word-document-matrix or the word representations in Random Indexing, see Section 2.5. Word clusterings can also be used for, among other things, word sense disambiguation, see Jurafsky and Martin (2000).

Visualization

A textual result presentation is limited by the amount of screen estate. Also, there is only so much text a user is willing to study. A review of visual presentation in IR and text clustering is (Hearst, 1999a).

The basic operator used in many IR and clustering methods is the similarity between texts. It is therefore quite natural to try to visualize this. Several methods try to map the multi-dimensional space of the text-to-text-similarities to a two-dimensional presentation. A very nice way to achieve this is inherent in the clustering method *Self Organizing Maps* (Haykin, 1999), which has been applied to text collections (Lagus *et al.*, 2004).

Unfortunately, so far, visualization systems are harder to use for non-expert users than text based systems like Scatter/Gather. This is because the content of texts is best understood by reading them. However, there is a firm belief that the user could gain from visualization as pictures can communicate some kinds of information very fast. (Hearst, 1999a)

Part II

Contributions

Chapter 4

Contributions

This section provides an overview of the papers in this thesis, summarizing the content by topic. Our main contributions are an investigation of representations for texts in Swedish and extensions to the work on how to use clustering for exploration of text sets. The work can be divided into four areas, that will be summarized in a section each: representation of Swedish texts in the vector space model (Section 4.2), evaluation of clustering results (Section 4.3), somewhat off-topic work on synonyms (Section 4.4), and text clustering as an exploration tool (Section 4.5). Before these four sections we discuss, in general, the experiments we have conducted in Section 4.1; the different parameters, which ones have been investigated, which ones have not, and why. We have also included a few additional experiments in the appendix.

After the summary, a short account of who has done what in the papers is given in Section 4.6. Finally, in Section 4.7 we discuss some possible future work.

4.1 Parameters and Experimental Settings

The partition of a set of texts into groups of (hopefully) content-wise related texts could be achieved in an unlimited number of ways. Any investigation must be restricted in scope. In this section we will describe some of the restrictions that have been used throughout this work and try to motivate them. The appendix includes some additional experiments that investigate a few of the parameters that have not been discussed in the rest of our work.

The fundamental motivation for our work and the scope of it, guided by the principles of the research focus of the human language technology group at KTH CSC, was described in Sections 1.3 and 1.4. Text clustering could be used for several things in several ways. We are interested in its potential as an exploration tool and argue that this has to be realized in an interactive manner.

Clustering Algorithm and Number of Clusters

There are many clustering algorithms available. We have only used the K-Means algorithm and its derivation Bisecting K-Means. The reasons are several. K-Means is fast compared to other algorithms, so it fulfills the criteria of being efficient in the research focus. It also makes it suitable for interactive exploration, which is the main motivation for our work. Further, it is a well-known algorithm, which make the results we present easier to understand, compare and verify.

We usually cluster to a small number of clusters. In the interactive exploration scenario a user is probably not willing to study many clusters. Also, the K-Means algorithm is faster for fewer clusters. We usually set the maximal number of iterations to 10 or 20. In Sections A.2 and A.3 we have made small investigations of the effect of the number of iterations and the number of clusters for K-Means.

Text Representation and Similarity

We represent each text by the words that appear in them in the common vector space model, see Chapter 2. In our work on representation for Swedish, see Section 4.2, we investigate the impact of changing the definition of a *word* in this context.

In our clustering experiments we have used the following weighting scheme:

$$w_{i,j} = tf_{i,j} \cdot idf_i \tag{4.1}$$

$$\mathrm{tf}_{i,j} = \begin{cases} c_1 + (1-c_1)\frac{n_{i,j}}{\max_i n_{i,j}} & \text{if } n_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

$$\mathrm{idf}_i = c_2 + \log \frac{n - n_{\mathrm{word}(i)}}{n_{\mathrm{word}(i)}}, \tag{4.3}$$

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 $n_{\mathrm{word}(i)}$ is the number of documents word i appears in. We found this scheme in (Frakes and Baeza-Yates, 1992), and have used $c_1 = 0.5$ and $c_2 = 0.5$.

To calculate similarity between texts we have used the cosine measure, see Section 2.2. There are many similarity measures to choose from, but we have used it in all our experiments to make them comparable and since it is commonly used. We also always normalize the text vectors after weighting in order to speed up calculations. We define the similarity between a text and a cluster as the dot product between the normalized text vector and the (unnormalized) centroid, as this gives the appealing average similarity between the text and all texts in the cluster.

In Section A.1 we have made a small investigation of the impact of the weighting scheme and the similarity measure.

Text Sets

Another very important matter, when considering clustering experiments, is which text sets are used. We have used the following text sets in our investigations As

	L1	L2	L3	L4	L5
AMSYK	11	28	114	361	969
YK80	12	59	288		

Table 4.1: The Occupation Classification Systems for text set Occ (number of categories per level)

we are particularly interested in Swedish text representation the majority are in Swedish.

- A few sets of newspaper articles from the Swedish newspapers Dagens Nyheter (DN, DN150) and Aftonbladet (A1, A2, A3, A150). All are extracted from the KTH News Corpus (Hassel, 2001). The sections of the newspapers provide a categorization with five categories for each set: culture/entertainment, economy, domestic, foreign, and sports.

- A set of open answers (in Swedish) 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, called AMSYK and YK80. Table 4.1 shows the number of categories on each level of these systems. See paper II for more information.

- A set of medical papers from Läkartidningen[2] (Med). Each paper has one 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 for more information.

- We have used two parts of the freely available English text set 20 Newsgroups[4], (Lang, 1995). It is, in the version we got, divided into training and test sets for supervised machine learning. We used the test set (20ngA) as one of the sets. We also selected about 500 texts from each of five of the categories[5] as a second text set (20ngB).

Table 4.2 gives some statistics for these sets. All counts are after removing stopwords and, for Swedish, splitting compounds into their components, not keeping the original word. We have also applied stemming, and filtered out very frequent and infrequent words, before counting the average number of words (stems) per document and how many of the documents the words appear in on average. For

[1] http://www.meb.ki.se/twinreg/index_en.html
[2] http://www.lakartidningen.se/
[3] http://www.nlm.nih.gov/mesh/meshhome.html
[4] Downloaded from `http://people.csail.mit.edu/jrennie/20Newsgroups/`
[5] The 20 Newsgroup categories comp.os.ms-windows.misc, sci.med, talk.religion.misc, rec.sport.hockey, and talk.politics.misc.

Text set	cat.	texts	words	stems	stems/ text	texts/ stem
20ngA	20	7 529	977 344	17 896	79	33
20ngB	5	2 592	513 266	13 547	98	19
DN	5	6 954	652 324	15 877	60	26
DN150	5	3 485	466 151	13 506	83	21
A1	5	2 500	140 920	7 096	34	12
A2	5	2 500	140 886	7 192	35	12
A3	5	2 500	142 930	7 261	35	12
A150	5	5 325	795 290	16 307	86	28
Med	*	2 422	4 383 169	26 102	317	29
Occ	*	43 341	453 105	6 179	9	65

Table 4.2: Text set statistics. * means there are several categorizations. Text sets 20ngA and 20ngB are in English, the rest in Swedish.

Swedish we have used the stemmer described in (Carlberger *et al.*, 2001), and for English an implementation of the Porter stemmer (Porter, 1980). This means that the statistics for 20ngA and 20ngB can not be compared to the others. Also they are the hardest sets to preprocess (there are many character sequences that are not proper words).

There are some differences in statistics for the text sets in the different papers due to somewhat differing preprocessing. In later papers we use lemmatization instead of stemming for Swedish. Therefore it is not possible to compare the actual evaluation measure values in the different papers.

4.2 Swedish Text Representation

Papers I, III and VI are mainly concerned with the impact of using Swedish language specific tools when building the vector space representation. That is, how to define a *word* in the word-document matrix.

Stemming improve clustering results moderately in paper I. In paper VI lemmatization improves results a lot for certain settings, however it depends on other parameters such as the number of clusters and which text set is clustered. We see no apparent reason for that stemming and lemmatization would result in fundamentally different results. Differences in results are probably due to specific implementations. In principle, it should be possible to tune a stemming algorithm to produce excellent results for a particular text type. On the other hand, lemmas are better suited for result presentation, as they are proper words.

Results never deteriorate when using stemming or lemmatization and they have further advantages. They both reduce the number of different terms, which speeds up similarity calculations. Also, the effect that the similarity of texts using dif-

ferent forms of a word increase probably becomes more and more important with decreasing number of words.

Papers I and III 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.

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. We have, however, not pursued this any further.

In paper VI we investigate what can be seen as the opposite of stemming and lemmatization: instead of trying to bring closely related words of different forms together we wanted to separate *homographs*, i.e. words that have the same form but mean different things. We tried to achieve this by taking the part-of-speech (PoS) of the word into account. Each word was augmented with a tag indicating its PoS. For nouns we also tried including the gender in the tag. Although this in theory separates at least a large part of homographs it did not improve results compared to the ordinary lemmatized representation. A possible explanation is that the centroid representation captures the different meaning of homographs through contextual information; the centroid contains words that appear in the same set of texts (the cluster).

4.3 Clustering Evaluation

We have put a lot of effort into the, sometimes very hard, problem of how to evaluate clusterings in most of the work. We use both external and internal evaluation. For external evaluation we use manually created resources, but never include humans directly.

A result as measured by any quality measure does not give any information in isolation. A reference is needed. In all evaluations we compare at least two methods. Sometimes we also use a random method as another reference. In paper I, III, and VI we compare clustering results for different Swedish text representations, see Section 4.2.

The two papers about text clustering exploration, that are discussed in Section 4.5, are concerned with evaluation. In paper V we use evaluation to find interesting clusters, and in paper VIII we introduce visual evaluation for similar reasons.

Paper II and VII are directly about evaluation. In paper VII we introduce a new evaluation scheme for word relation models, see Section 4.4.

Paper II presents two ways of using two different classifications of the same text set for clustering evaluation. Most extrinsic evaluation measures compare a partition (normally a clustering) with one 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*, see Eugenio and Glass (2004). Here we consider the clustering and one of the classifications to be classification attempts following the second classification. The kappa coefficient measures the agreement of two classification attempts.

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

4.4 Synonyms

Papers IV and VII contain some work that is not directly associated to text clustering, although in the long run it may lead to better text representations. They deal with synonyms, or related words. In paper IV we describe the construction of a list of Swedish synonyms, called the People's Dictionary of Synonyms, by large-scale cooperation over the internet. Many users of the online dictionary Lexin[6] have answered questions about possible synonyms. They were allowed to grade each pair of word on a scale from zero to five depending on how "synonymous" they considered them to be. After some filtering this has produced a list of synonyms with average grades from three to five, that is freely available.

In paper VII we use the automatic word relation extraction method Random Indexing. We compare it to the list of synonyms from paper IV by means of word clustering based on the Random Index. This evaluation takes all words into consideration. Usually, word relation models are evaluated directly on the pairs in the synonym list.

The evaluation in paper VII deems the syntagmatic versions of RI to be better than the paradigmatic ones[7]. This is quite opposite to earlier results and can be explained in at least two ways:

- The syntagmatic versions of Random Indexing are better to model all the relations between words than the paradigmatic, while at the same time the paradigmatic versions are better for individual pairs of synonyms (which earlier evaluations have shown).

[6] http://lexin.nada.kth.se/

[7] See Section 2.5 for a discussion of syntagmatic and paradigmatic relations.

- The list of synonyms we created in paper IV is not comparable to the resources that have been used in earlier evaluations.

Although we are inclined to hold the first for most likely, we will discuss the second possibility. The list of synonyms we (or rather, the users of Lexin) created is different from other resources in that it lets the users define what synonymy is. However, as the results in paper VII does not differ when we only evaluate on the pairs that have been graded five, we do not think this is important. Indeed, as the word relation that automatic methods extract does not fully agree with the definition of synonymy, our list of synonyms might be a better resource for evaluation.

4.5 Text Clustering Exploration

Papers V and VIII are concerned with text clustering as an exploration tool. The first paper presents a method that could aid exploration and that could be used in any clustering system. Paper VIII describes our program Infomat.

Hypotheses from Free Text

The main contribution of paper V is a method for extracting information from free text answers (text answers to open questions) in questionnaires. These are usually never studied as it requires that a human goes through them all, which is too time-consuming and costly. Using the answers to a closed question (a question with a fixed number of possible answers) the respondents can be divided into as many groups as there is possible answers. These groups can be considered a categorization and used to evaluate a clustering of a free text answer using external quality measures. The quality for a particular cluster reveals a relation between its content and the closed answer. The clusters can be sorted in order of their quality helping the explorer to find interesting clusters.

In the paper we worked with a free text answer where respondents have described their occupation (the text set Occ) and a closed answer regarding smoking (yes/no). We were able to extract the relation "farmers smoke less than the average", something we also verified by reading earlier studies. This was a first verification that this method has potential. We hope to go on and try it on other questionnaires and other data sets containing both free text and restricted data, such as medical records.

A relation that an application of the method has as a result should in general not be considered conclusive, but rather a hypothesis that should be further studied. The method should be seen as way of generating hypotheses from a combination of free text and restricted data.

Infomat – Visualization, Exploration, Experimentation

Paper VIII describes our system Infomat[8]. It is implemented in Java[9] and can be run as a visual exploration tool, and also as a command prompt program for textual result generation as well as experimentation.

When Infomat is used as a visual exploration tool, it presents any sparse matrix as a scatter plot. For text clustering it is the word-document-matrix. The similarities of texts (and words) appear as visual distributional patterns in the picture. Clusterings and other orderings of texts and words expose similarity trends in the data. The user can explore the matrix in many different ways, among others: zooming in and out of the picture, deleting rows, columns, and matrix elements.

In parallel with the matrix picture Infomat also provides a lot of textual information in different ways. The user may retrieve information about any part of the matrix, and lists of clusters with lists of texts or words.

There are functions for ordinary evaluation, both internal and external allowing any two groupings to be compared. The interface also encourage *visual evaluation*. In internal visual evaluation the user compares two or more clusters or clusterings by inspecting the density of their distributions. External visual evaluation is achieved by color coding a second grouping: each group is assigned a color. This makes it possible to see how its groups are distributed over the currently displayed grouping.

Infomat can also be run from the command prompt and generate xml results[10], that can be viewed in a browser. A main page list clusters with a cluster digest. Each cluster is a hyper link to a page with more information about the cluster. The texts in the clusters are also links which lead to the actual texts if the program have this information.

Infomat is implemented with a lot of flexibility in mind. Among other things the parameters of each method is realized as an instance of a special properties class. It has a graphical interface, which makes how to set parameters congruent, and also makes it easy to set up experiments using the special experimentation class.

Experiment results in evaluation files are also constructed in a systematic way. There are classes for generating result tables that can be exported to several different formats for presentation and processing in other programs.

Clustering Results Unraveled

We conclude this summary by discussing Picture 4.1[11]. It demonstrates the combination of our visualization (the Infomat interface) and an idea that we are currently working on, that we call *cluster trimming*. The text set A1 has been clustered to five text clusters, which are divided by horizontal separator lines.

[8] http://www.csc.kth.se/tcs/projects/infomat/infomat/

[9] http://java.sun.com/

[10] Such results can also be exported from the exploration interface.

[11] It is also the picture on the cover of this thesis with the separator lines removed, the background changed from white to blue, and the grey scale (white to black) changed to a monochrome scale from blue to white.

For each text cluster the centroid is considered a text cluster description consisting of several words. A word cluster is built from each description consisting of ten of the words with highest scores (centroid value). A word that is in several descriptions (word clusters) is put in the one where it has the highest score. These clusters are presented as column groups, separated by vertical lines. The words are ordered by description score (highest to the left) within each group. In Table 4.3 the words for each cluster/description are presented.

The text clusters are *trimmed* to fit their descriptions: only texts that have a similarity greater than zero to the corresponding description are retained. They are ordered, within each cluster, by their similarity to the description (higher at the top).

This picture has several advantages. As the number of texts and, particularly, words are reduced the construction of the picture is much faster. This leads to quicker interaction.

Most importantly this picture is more comprehendable. We know that at least one of the description words is present in all texts for each cluster. There are quite a lot of texts removed in the trimming, see the caption of Figure 4.1, texts that might have nothing to do with the descriptions.

Although, when using a word clustering with all words, it is possible to zoom in and inspect single words, this view is more direct, presenting only some very interesting words. If there are natural subgroups of texts within a cluster these can become apparent as they may use a subset of the description words, that is not used by the other texts.

The picture also shows how the description words for each cluster are distributed over all the other clusters as well. This is a great advantage over just textual descriptions as it makes it possible to understand which words are important and how the clusters relate to each other. Also, it is possible for the user to remove less interesting words and redo the trimming based on the smaller descriptions.

Looking at this particular picture (4.1) it is quite obvious that clusters three and four are very related – they have many words from both descriptions. The other clusters are more homogenous. Table 3.1 shows the confusion matrix for the original clustering. It verifies that texts in clusters three and four are related; most of them are from the economy section of the newspaper. Cluster one capture the entertainment section rather well, and cluster two captures almost all of the sports section. Most of the texts from both the foreign and the domestic sections are found in cluster five. They obviously have a lot of words in common.

4.6 Author Contributions

This section contains a short account of who have contributed to the work in the included papers. I have received excellent feedback from my supervisor Viggo Kann during all the stages of their development.

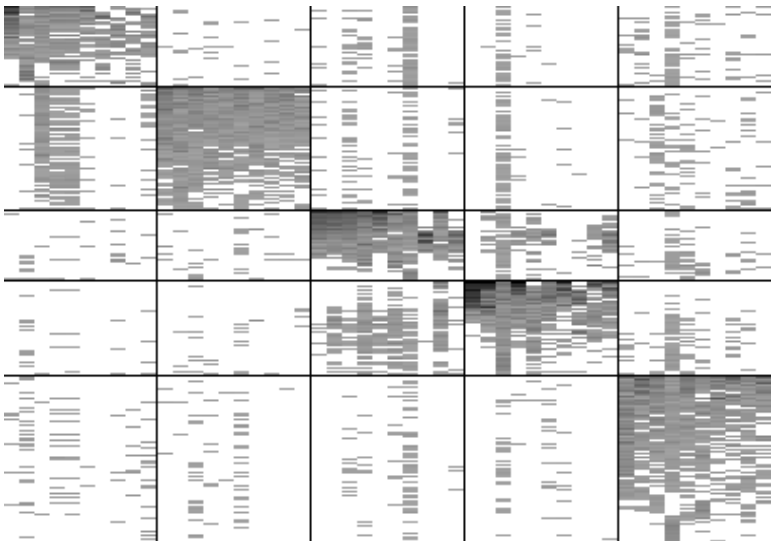


Figure 4.1: Clustering Results Unraveled. Text set A1 (2500 texts) clustered to five clusters, along the rows. Original size of clusters: 373, 558, 352, 489, 728. Descriptions of 10 words are constructed for each text cluster and presented as word clusters, along the columns. The text clustering is trimmed to fit them, retaining 1997 texts, with clusters of sizes: 307, 454, 263, 354, 619. The word clusters/descriptions are shown in Table 4.3. See Table 3.1 for a confusion matrix for the original clustering compared to the newspaper sections.

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

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

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 Jan-Eric and Viggo Kann. Some formulations are Henrik Eriksson's.

nr	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5
1	film	match	message	index	police
2	aftonbladet[1)]	win	reuter[2)]	s. ex.[3)]	death
3	play	team	press	krona[4)]	injury
4	afton[1)]	club	company[5)]	increase	state[6)]
5	blad[1)]	player	rs[2)]	market	person
6	role	goal	company[5)]	trade	report[6)]
7	album	victory	write	general	death
8	band	game	tokyo s. ex.[3)]	interest	car
9	festival	ball	stock	ericsson	city
10	release	season	broad	share price	accident

Table 4.3: Word clusters/descriptions. See Figure 4.1. Words are translated from Swedish. Compare these descriptions to Table 3.1, that is a confusion matrix for the original text clustering compared to the newspaper sections. 1) The Swedish newspaper Aftonbladet is sometimes split into its parts afton (evening) and blad (meaning newspaper here). 2) Reuters - the news service. *rs* is our abbreviation for *reutersstockholm*. Stockholm - the capital of Sweden. Obviously it is quite common that the two words are written together. 3) Our abbreviation for stock exhcange. 4) The Swedish currency, a crown. 5) Two different words for company or corporation. 6) Both in the sense, that something is conveyed.

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

The interest in phrases and in particular for medical texts was collective within the Infomat project. Sumithra Velupillai 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.

Paper IV. Viggo Kann and Magnus Rosell: "Free Construction of a Free Swedish Dictionary of Synonyms". (Kann and Rosell, 2005)

This paper is primarily by Viggo Kann. I helped with the automatic refinement of the initial list of possible synonyms using Random Indexing. I wrote parts of the paper and assisted Viggo overall.

Paper V. Magnus Rosell, Sumithra Velupillai: "Revealing Relations between Open and Closed Answers in Questionnaires through Text Clustering Evaluation". (Rosell and Velupillai, 2008)

The idea was mine. I implemented Infomat (see Paper IX). Sumithra Velupillai and I made the experiments and wrote the paper. We had invaluable discussions with Jan-Eric Litton and Catharina Rehn. Catharina provided

search services from the Library at KI, helping us to find related studies of questionnaires including occupational questions.

Paper VI. Magnus Rosell: "Part of Speech Tagging for Text Clustering in Swedish", (Rosell, 2009b)

Viggo Kann and I had discussed using the information in part-of-speech-tags to try to separate homographs for a long time. I made the experiments and wrote the paper.

Paper VII. Magnus Rosell, Martin Hassel, Viggo Kann: "Global Evaluation of Random Indexing through Swedish Word Clustering Compared to the People's Dictionary of Synonyms", (Rosell *et al.*, 2009)

I came up with the idea. Martin Hassel made the preprocessing and constructed most of the Random Indexes. I implemented the clustering and evaluation code, and performed all experiments. I wrote the paper with assistance from Viggo and Martin.

Paper VIII. Magnus Rosell: "Infomat – Visualizing and Exploring Vector Space Model Data Matrixes", (Rosell, 2009a)

The participants in the Infomat project thought a graphical user interface could be useful. The ideas for this particular interface were mine. I designed and implemented the program, and wrote the paper.

4.7 Future Work

The Infomat system makes it easy to set up new experiments. For the moment we do not have any new ideas that we believe would improve results for Swedish, but there are several other representational issues left to investigate. We would particularly like to try denser representations using Random Indexing. Also, we have not tried many different similarity definitions so far.

There are many clustering algorithms that could be better for exploration. In particular, simultaneous clustering of texts and words would be interesting.

We hope to apply our exploration methods and Infomat to other text sets and perhaps even make a more thorough user study concerning the interface. The trimming method we discussed briefly in this chapter has great potential. In particular we would like to use the hypothesis generation method of paper V on other questionnaires and medical records.

Bibliography

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

P. Behrkin. 2006. *Grouping Multidimensional Data Recent Advances in Clustering*, chapter A Survey of Clustering Data Mining Techniques, pages 25–71. Springer Berlin Heidelberg. ISBN 978-3-540-28349-2.

F. Beil, M. Ester, and X. Xu. 2002. Frequent term-based text clustering. In *KDD '02: Proc. of the 8th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 436–442, New York, NY, USA. ACM. ISBN 1-58113-567-X.

M. W. Berry, Z. Drmac, and E. 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. S. Dhillon. 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In *KDD '01: Proc. 7th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 269–274, New York, NY, USA. ACM. ISBN 1-58113-391-X.

I. S. 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.

M. Halkidi, Y. Batistakis, and M. Vazirgiannis. 2001. On clustering validation techniques. *J. of Intelligent Information Systems*, 17(2-3):107–145. URL `http://citeseer.ist.psu.edu/article/halkidi01clustering.html`.

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.*

S. Haykin. 1999. *Neural Networks: A Comprehensive Foundation*. Prentice Hall International, Upper Saddle River, NJ, USA. ISBN 0-13-908385-5.

M. A. Hearst. 1999a. *Modern Information Retrieval*, chapter User interfaces and visualization, pages 247–323. Addison-Wesley. ISBN 0-201-39829-X.

M. A. Hearst. 1999b. 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.

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`.

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.

M. Käki. 2005. Findex: search result categories help users when document ranking fails. In *CHI '05: SIGCHI conference on Human factors in computing systems*, pages 131–140, New York, NY, USA. ACM. ISBN 1-58113-998-5.

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.*

V. Kann and M. Rosell. 2005. Free construction of a free Swedish dictionary of synonyms. In *Proc. 15th Nordic Conf. on Comp. Ling. – NODALIDA '05.* URL `http://www.nada.kth.se/theory/projects/infomat/rapporter/kannrosell05.pdf`.

S. Kaski. 1998. Dimensionality reduction by random mapping: Fast similarity computation for clustering. In *Proceedings of IJCNN'98, International Joint Conference on Neural Networks*, volume 1, pages 413–418. IEEE Service Center, Piscataway, NJ. URL `citeseer.ist.psu.edu/kaski98dimensionality.html`.

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

A. Kulkarni and T. Pedersen. 2005. SenseClusters: unsupervised clustering and labeling of similar contexts. In *ACL '05: Procc of the ACL 2005 on Interactive poster and demonstration sessions*, pages 105–108, Morristown, NJ, USA. Association for Computational Linguistics.

K. Lagus, S. Kaski, and T. Kohonen. 2004. Mining massive document collections by the websom method. *Inf. Sci.*, 163(1-3):135–156. ISSN 0020-0255.

T.K. Landauer and S. T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240.

K. Lang. 1995. Newsweeder: Learning to filter netnews. In *Proc. of the Twelfth Int. Conf. on Machine Learning*, pages 331–339.

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.

M. J. Maña-López, M. De Buenaga, and J. M. Gómez-Hidalgo. 2004. Multidocument summarization: An added value to clustering in interactive retrieval. *ACM Trans. Inf. Syst.*, 22(2):215–241. ISSN 1046-8188.

S. Macskassy, A. Banerjee, B. Davison, and H. Hirsh. 1998. Human performance on clustering web pages: A preliminary study. In *The Fourth Int. Conf. on Knowledge Discovery and Data Mining (KDD-98)*. URL `citeseer.comp.nus.edu.sg/4251.html`.

C. D. Manning, P. Raghavan, and H. Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press. ISBN 978-0521865715. URL `http://www-csli.stanford.edu/~hinrich/information-retrieval-book.html`.

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

K. R. McKeown, R. Barzilay, D. Evans, V. Hatzivassiloglous, J. L. Klavans, A. Nenkova, C. Sable, B. Schiffman, and S. Sigelman. 2002. Tracking and summarizing news on a daily basis with columbia's newsblaster. In *Proceedings of the second international conference on Human Language Technology Research*, pages 280–285, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Q. Mei, X. Shen, and C. Zhai. 2007. Automatic labeling of multinomial topic models. In *KDD '07: Proc. of the 13th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 490–499, New York, NY, USA. ACM. ISBN 978-1-59593-609-7.

T. M. Mitchell. 1997. *Machine Learning*. McGraw-Hill, New York. ISBN 0-07-115467-1.

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.F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

R. Rapp. 2003. Word sense discovery based on sense descriptor dissimilarity. In *Machine Translation Summit IX*. URL `http://www.amtaweb.org/summit/MTSummit/papers.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*. URL `http://www.nada.kth.se/~rosell/publications/papers/rosell03.pdf`.

M. Rosell. 2005. *Clustering in Swedish – The Impact of some Properties of the Swedish Language on Document Clustering and an Evaluation Method*. Licentiate thesis, School of Computer Science and Communication, Royal Institute of Technology, Stockholm, Sweden. ISBN 91-7178-166-8. URL `http://www.nada.kth.se/~rosell/thesis/rosell_lic_thesis.pdf`.

M. Rosell. 2009a. Infomat – visualizing and exploring vector space model data matrixes. Submitted.

M. Rosell. 2009b. Part of speech tagging for text clustering in Swedish. Submitted.

M. Rosell, M. Hassel, and V. Kann. 2009. Global evaluation of random indexing through Swedish word clustering compared to the people's dictionary of synonyms. Submitted.

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. URL `http://www.nada.kth.se/~rosell/publications/papers/rosellkannlitton04.pdf`.

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*. URL `http://www.nada.kth.se/~rosell/publications/papers/rosellvelupillai05.pdf`.

M. Rosell and S. Velupillai. 2008. Revealing relations between open and closed answers in questionnaires through text clustering evaluation. In European Language Resources Association (ELRA), editor, *Proc. of the 6th Int. Language Resources and Evaluation (LREC'08)*. URL `http://www.lrec-conf.org/proceedings/lrec2008/pdf/438_paper.pdf`.

L. Råde and B. Westergren. 1995. *Mathemetics Handbook for Science and Engineering*. Studentlitteratur, Lund, Sweden. ISBN 91-44-25053-3.

M. Sahlgren. 2006. *The Word-Space Model: using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. PhD thesis, Stockholm University, Stockholm, Sweden. ISBN 91-7155-281-2. URL `http://eprints.sics.se/437/01/TheWordSpaceModel.pdf`.

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

H. Schütze. 1993. Word space. In S. Hanson, J. Cowan, and C. Giles, editors, *Advances in Neural Information Processing Systems 5*. Morgan Kaufmann Publishers. URL `citeseer.ist.psu.edu/schutze93word.html`.

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 and J. Ghosh. 2003. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617. ISSN 1533-7928.

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`.

A. Tombros, R. Villa, and C.J. Van Rijsbergen. 2002. The effectiveness of query-specific hierarchic clustering in information retrieval. *Inf. Process. Management*, 38(4):559–582. ISSN 0306-4573.

C. J. Van Rijsbergen. 1979. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow. URL `http://www.dcs.gla.ac.uk/Keith/Preface.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`.

Y. Zhao and G. Karypis. 2002. Evaluation of hierarchical clustering algorithms for document datasets. In *CIKM '02: The 11th Int. Conf. on Information and Knowledge Management*, pages 515–524, New York, NY, USA. ACM. ISBN 1-58113-492-4.

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

Appendix A

Some Additional Experiments

There are many parameters that could be changed, and that might have an effect on the clustering results. The number of combinations are too many for them all to be investigated. Some need to be kept constant. In the following we present some additional experiments that investigate the impact of a few of the most apparent and important. They are meant to be a complement to the other experiments in this thesis and serve as a background to and motivation for some of the choices we have made on what to investigate in those.

We have clustered the text sets DN, Occ, and 20ngA (see Section 4.1) using K-Means, with a maximum of 20 iterations. When nothing else is indicated we clustered DN to 5 clusters, and Occ and 20ngA to 20 clusters. For all of them we have removed stopwords, and in most cases filtered out very uncommon and very common words. For the Swedish sets, DN and Occ, we have used lemmatization and compound splitting, and for 20ngA stemming. In all experiments, were nothing else is stated, we use the weighting scheme defined by Equations 4.2 and 4.3, normalize texts, and use the dot product to calculate similarity between texts and cluster centroids.

Where nothing else is indicated we present average results with standard deviations, for 20 runs of K-Means for each setting. Results are measured using NMI (see Equation 3.20) and, when appropriate, the "*self similarity*" Φ_{intra} (see Equation 3.5), which we abbreviate as Φ in the graphs.

For the external evaluation we compare DN and 20ngA to their manual categorizations, and Occ to level 2 of the AMSYK categorization, see Section 4.1. For the results presented through graphs, we do not include the results for text set DN, as they are very similar in tendency to those for 20ngA.

In the first section (A.1) we investigate the impact of a few weighting schemes and similarity measures. The following sections (A.2 through A.5) present and discuss experiments on the effect of the number of iterations in K-Means, the number of clusters, the number of texts, and the number of words. Before summarizing the results in the last section (A.7), we present one evaluation using several evaluation

measures (A.6) and discuss them.

A.1 Weighting and Similarity

Which weighting scheme and similarity measure (see Section 2.1) is used can have quite an impact on results in many applications. We compare two term frequencies (tf) and two inverse document frequencies (idf), combined and separately. These are the tfs:

$$\mathrm{tf}_{i,j} = \frac{n_{i,j}}{\sum_i n_{i,j}}, \tag{A.1}$$

$$\mathrm{tf}^{(2)}_{i,j} = \begin{cases} c_1 + (1-c_1)\frac{n_{i,j}}{\max_i n_{i,j}} & \text{if } n_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{A.2}$$

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 $c_1 = 0.5$. These are the idfs:

$$\mathrm{idf}_i = \log \frac{n}{n_{\mathrm{word}(i)}}, \tag{A.3}$$

$$\mathrm{idf}^{(2)}_i = c_2 + \log \frac{n - n_{\mathrm{word}(i)}}{n_{\mathrm{word}(i)}}, \tag{A.4}$$

where $n_{\mathrm{word}(i)}$ is the number of documents that word i appears in, and we have used $c_2 = 0.5$. When we use the idfs separately we set $w_{i,j} = 0$ when $n_{i,j} = 0$. $\mathrm{tf}^{(2)}_{i,j}$ and $\mathrm{idf}^{(2)}_i$ are the same as those in Equations 4.2 and 4.3, while $\mathrm{tf}_{i,j}$ and idf_i are the simplest versions, see Section 2.1.

We also try three different similarity measures: the dot product and the cosine measure, which sometimes coincide, see Section 2.2, and the inverse of the Cartesian distance:

$$\mathrm{sim}(d_u, d_v) = \frac{1}{\mathrm{dist}(d_u, d_v)} = \frac{1}{\sqrt{\sum_i (w_{i,u} - w_{i,v})^2}}. \tag{A.5}$$

We use the centroid to represent clusters. It is always the average vector. We have tried to normalize the texts and not to do it. This gives us six different combinations with the three similarity measures. All six are tried with the different weighting schemes. In Table A.1 we present four of these, as the results did not differ for the cosine measure and the dot product between using normalized or not normalized texts. We have marked the weighting-similarity-combination that is used in other experiments in the thesis in bold face.

We see that the cosine similarity and the dot product perform better than the Cartesian similarity. It is beneficial to normalize texts for the Cartesian similarity, but it does not matter whether we do it or not for the others.

Similarity	Weighting, $w_{i,j}$ =	DN	Occ	20ngA
Cosine	tf	0.43 (0.02)	0.22 (0.01)	0.34 (0.02)
(normalized	idf	0.55 (0.04)	0.38 (0.01)	0.54 (0.03)
texts)	$\mathrm{tf} \cdot \mathrm{idf}$	0.51 (0.04)	0.37 (0.01)	0.54 (0.02)
	$\mathrm{tf}^{(2)}$	0.48 (0.03)	0.21 (0.01)	0.35 (0.02)
	$\mathrm{idf}^{(2)}$	0.53 (0.05)	0.37 (0.01)	0.55 (0.02)
	$\mathrm{tf}^{(2)} \cdot \mathrm{idf}^{(2)}$	0.54 (0.05)	0.37 (0.01)	0.57 (0.03)
Dot Product	tf	0.44 (0.03)	0.21 (0.01)	0.34 (0.01)
(normalized	idf	0.54 (0.05)	0.37 (0.01)	0.53 (0.03)
texts)	$\mathrm{tf} \cdot \mathrm{idf}$	0.52 (0.05)	0.37 (0.01)	0.54 (0.02)
	$\mathrm{tf}^{(2)}$	0.47 (0.03)	0.22 (0.02)	0.35 (0.02)
	$\mathrm{idf}^{(2)}$	0.54 (0.05)	0.37 (0.01)	0.55 (0.02)
	$\mathbf{tf^{(2)} \cdot idf^{(2)}}$	**0.54 (0.06)**	**0.37 (0.01)**	**0.56 (0.03)**
Cartesian	tf	0.39 (0.06)	0.16 (0.01)	0.23 (0.02)
(texts not	idf	0.31 (0.03)	0.32 (0.02)	0.34 (0.01)
normalized)	$\mathrm{tf} \cdot \mathrm{idf}$	0.46 (0.07)	0.30 (0.01)	0.44 (0.02)
	$\mathrm{tf}^{(2)}$	0.35 (0.03)	0.17 (0.01)	0.27 (0.01)
	$\mathrm{idf}^{(2)}$	0.31 (0.02)	0.31 (0.02)	0.36 (0.02)
	$\mathrm{tf}^{(2)} \cdot \mathrm{idf}^{(2)}$	0.34 (0.03)	0.33 (0.01)	0.42 (0.03)
Cartesian	tf	0.35 (0.04)	0.19 (0.01)	0.28 (0.01)
(normalized	idf	0.49 (0.03)	0.33 (0.01)	0.50 (0.03)
texts)	$\mathrm{tf} \cdot \mathrm{idf}$	0.40 (0.03)	0.32 (0.02)	0.48 (0.02)
	$\mathrm{tf}^{(2)}$	0.45 (0.04)	0.18 (0.02)	0.33 (0.02)
	$\mathrm{idf}^{(2)}$	0.50 (0.03)	0.33 (0.02)	0.51 (0.03)
	$\mathrm{tf}^{(2)} \cdot \mathrm{idf}^{(2)}$	0.49 (0.03)	0.33 (0.01)	0.53 (0.02)

Table A.1: Clustering results for different weighting schemes and similarity measures on different text sets. Average NMI for 20 K-Means clusterings (standard deviations). The method in bold face is the one used in other experiments in this thesis.

The inverse document frequency seems to be really important and the term frequency seems not to contribute to the results. However, there is no difference between the versions of term frequency and inverse document frequency. There is no scheme that performs better than the one we have used (indicated in bold face).

A.2 Number of Iterations

Figure A.1 presents the result for increasing number of iterations of the K-Means algorithm. We have evaluated the result of all iterations in 20 different runs. Iteration zero is the initial random partition.

The quality increases very rapidly during the first few iterations, and the improvement is very modest after only ten iterations.

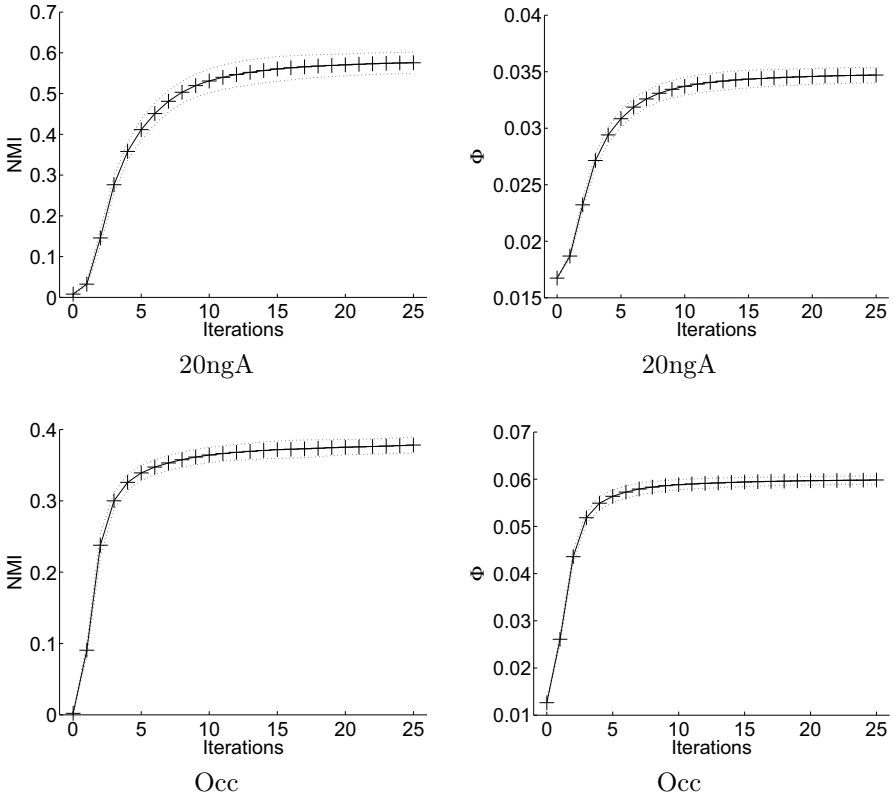


Figure A.1: The effect of increasing number of iterations in K-Means on different text sets (rows of graphs) as measured with normalized mutual information (NMI) and self similarity Φ_{intra} (denoted Φ here). Average values over 20 K-Means clusterings. The dotted lines represent the standard deviations.

A.3 Number of Clusters

Figure A.2 gives the results for clusterings to 2, 3, 4, 5, 10, 20, 30, 40, 50, 100 and 200 clusters. Here we present the result in mutual information (MI), in addition to NMI and Φ_{intra}, see Section 3.3.

The result in Φ increases with the number of clusters; the average similarity within clusters increase when the number of clusters increases (and their sizes decrease). This is intuitive and is reasonable considering the model (the vector space model); if the clustering works, small clusters of related texts should be more self similar than larger.

The external evaluation gives a somewhat different result. For text set Occ the

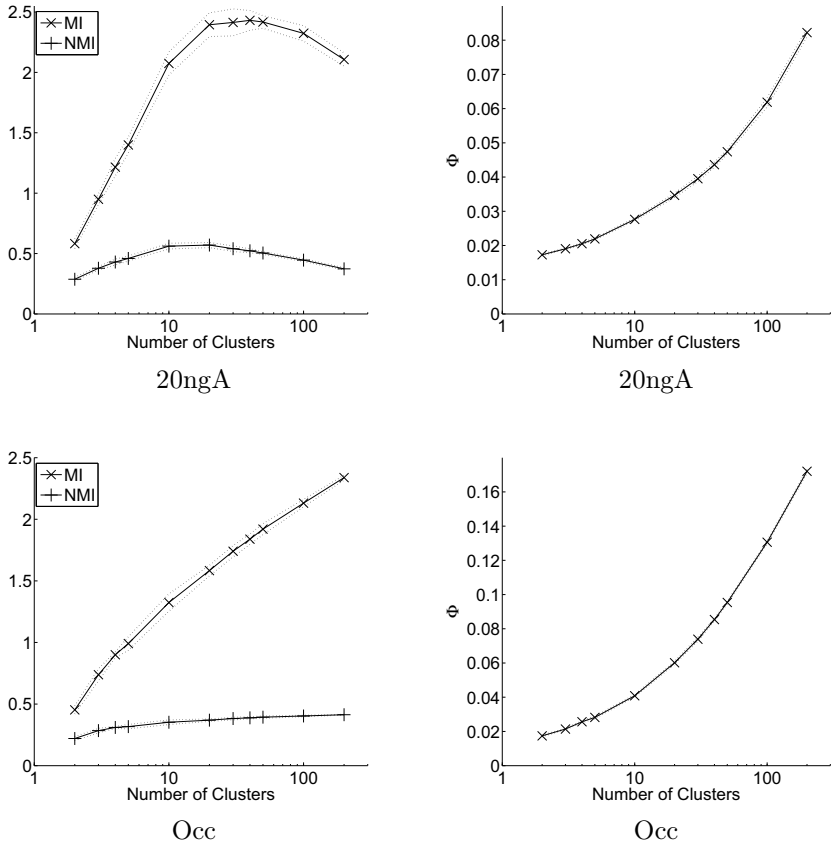


Figure A.2: The effect of increasing the number of clusters on different text sets (rows of graphs) as measured with mutual information (MI), normalized mutual information (NMI), and self similarity Φ_{intra} (denoted Φ here). The number of clusters are presented on a logarithmic scale. Average values over 20 K-Means clusterings. The dotted lines represent the standard deviations.

quality grows with the number of clusters, but for 20ngA (and DN) there is a peak in quality for a few numbers of clusters. The peak is related to the number of categories in the manual categorization used for the set. The mutual information (MI) reaches its peak around the number of clusters corresponding to the number of categories (slightly after for DN). It stays the same for some higher numbers of clusters, but then decreases with growing numbers.

The normalized mutual information reaches its peak for a slightly lower number of clusters than the number of categories, and then degrades for larger numbers. The

difference between NMI and MI is due to the denominator of NMI, that depends on the number of categories (which is fixed) and grows with the number of clusters (as $\sqrt{H(C)}$, see Equation 3.20). It punishes clusterings of larger numbers of clusters, something that might be reasonable, but depends on the situation.

To compare the results of internal and external evaluation for these experiments exposes the problem with external evaluation; it depends on the categorization. Consider the newspaper sections, used to evaluate DN: culture/entertainment, economy, domestic, foreign, and sports. Some news might be about the acquisition of players or the finances of a sports club. Depending on how these news items are pitched they might end up in the economy or the sports section. If a clustering of many clusters finds this group of texts and put them all in a cluster, that cluster will be considered being of lower quality than two clusters that split them. It is possible that there are effects of this kind that are the reasons for result in mutual information for larger numbers of clusters in the graph for 20ngA.

The result for text set Occ in the external evaluation might show that the categorization of occupations used in AMSYK (see Section 4.1) is hard to extract using only the short text answers; the categorization is built using more external knowledge.

A.4 Number of Texts

We removed a series of predefined numbers of texts at random from the original text set yielding several text sets with decreasing number of texts. These sets were preprocessed, weighted, and then clustered 10 times with K-Means (20 iterations), and random "clusterings" for comparison. We repeated the whole procedure ten times resulting in 100 clustering results for each number of texts[1].

To be able to evaluate a clustering on a reduced text set we constructed a corresponding categorization by removing the same texts from the original categorization. With decreasing number of texts this might by chance lead to easier categorizations, as for instance an entire category *could* not be present in a reduced text set. Hence, we repeat the entire reduction procedure ten times as described above.

The results are shown in Figure A.3. For low numbers of texts K-Means does not perform better than a random clustering, both as measured using external (NMI) and internal measures (Φ).

Mutual information is (and essentially all external measures are) defined using $p_i^{(j)} = m_i^{(j)}/n$ as a probability, the probability that a text picked at random from the text set belongs to cluster c_i and category $k^{(j)}$, see Section 3.3. For few texts, small n, we can not do this estimation. The fluctuations in the numbers $m_i^{(j)}$ that happen by chance become too significant, and the measure breaks down. Results for sets of few texts for which K-Means and random clustering perform similarly can thus not be evaluated.

[1] Actually it is an average number of texts, as sometimes a few texts are removed in the preprocessing, since they include only very rare words in the reduced text set.

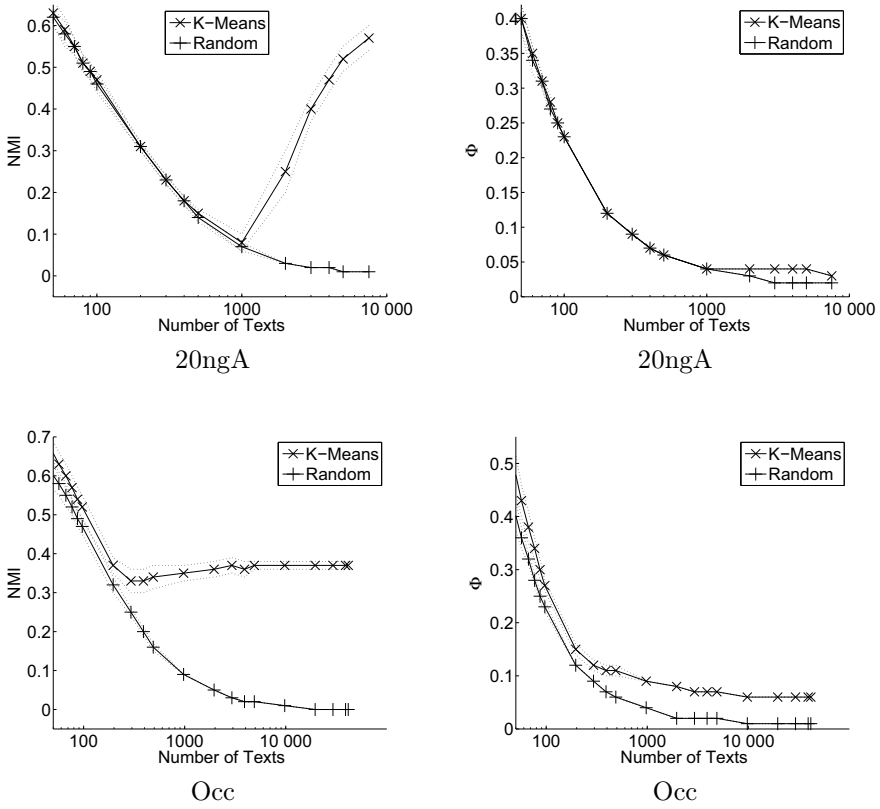


Figure A.3: The effect of the number of texts on different text sets (rows of graphs) as measured with normalized mutual information (NMI), and self similarity (Φ). The number of texts are presented on a logarithmic scale. Average values over 20 K-Means and random "clusterings". The dotted lines represent the standard deviations.

When there is a difference between random clustering and K-Means we learn that K-Means clustering perform better the more texts it considers for text sets 20ngA and DN. For text set Occ the improvement using more texts is rather modest. This also confirms the discussion about the AMSYK categorization in the previous section; it probably is defined using external knowledge that is not in the text answers.

While the external evaluation breaks down for few texts there is nothing that suggests that this ought to be the case for internal evaluation. However, K-Means performs similarly to random clustering for few texts measured also by Φ, the self

similarity. This shows that the representation model (the vector space model for text clustering) does not contain enough information to find more similar groups than random partition for few texts. Judging by the result for more texts, the difference in similarity between the random clustering and the K-Means clustering does not have to be big to find real differences (compare to the external evaluation).

That Φ decreases with growing number of texts is expected, since the number of texts in each group increases. The self similarity for a set of few texts is usually higher than for a group of many texts. Also, Φ includes the similarity of each text to itself. When the texts are very few this contribution is not negligible.

To summarize, in order for text clustering to work we need to have a not insignificant number of texts. For texts of the kind investigated here around 1000 seems to be enough, but the more the better.

A.5 Number of Words

We also investigate the impact of the number of words in the representation. There are many ways in which words could be removed. Here, we study which words are more important: the ones with high document frequency (number of texts they appear in, compare to idf – inverse document frequency) or the ones with low.

Figure A.4 report some experiments where we have removed words with successively higher document frequency (left column of graphs) and with successively lower document frequency (right column). The dashed line gives the portion of remaining words after the removal, and the whole line the average result over 20 K-Means runs in NMI (with standard deviation as dotted lines). We do not present results in Φ_{intra} here, as we change the representation when we remove words.

For these text sets we can remove up to 80% of the words with lowest document frequency (i.e. up to a document frequency of roughly 30) without the results deteriorating. These words do not contribute much to the similarity between most of the texts. However, for a higher number of clusters, words with lower document frequency will probably be more important as the algorithm needs to construct smaller clusters, and differentiate texts on a more detailed level.

The words with high document frequency are much fewer. Most of these are very important for the clustering result, but we can remove the most frequent; those appearing in more than around 1000 texts. This corresponds well to the use of a stoplist. It may be so that when clustering to a large number of clusters these words might be detrimental to clustering results, following the same line of reasoning as for the words with low document frequency.

In conclusion, we can remove a large portion of the words; those that appear in few texts. They contribute little to the similarity between most texts.

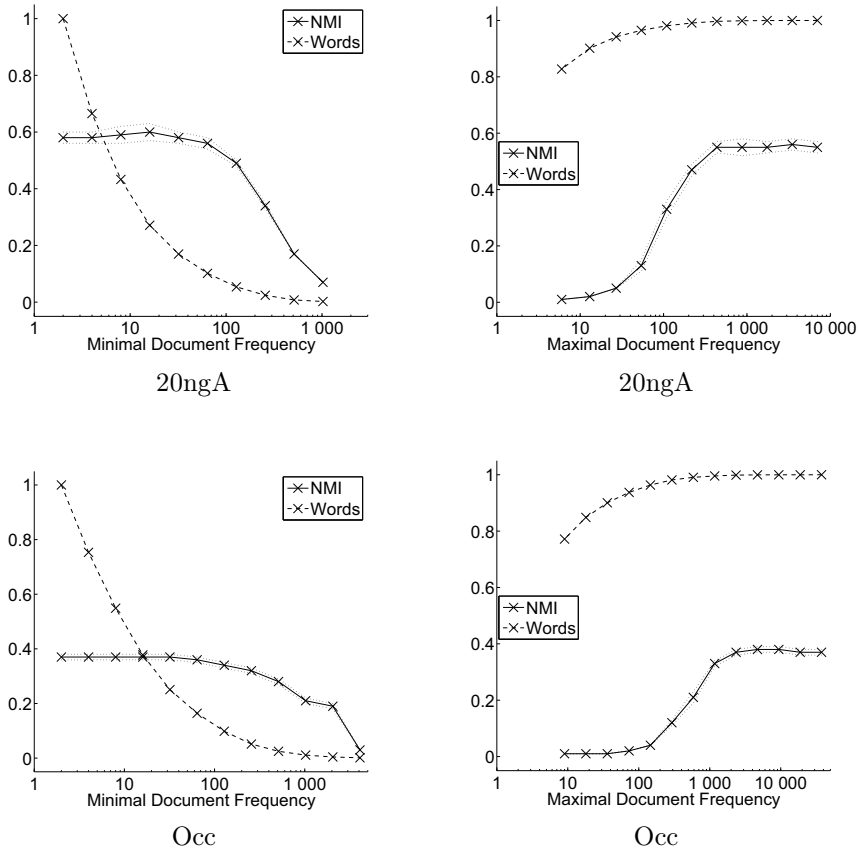


Figure A.4: The effect of words with different document frequencies on different text sets (rows of graphs) as measured with normalized mutual information (NMI). The document frequencies are presented on a logarithmic scale. Average values over 20 K-Means clusterings. The dotted lines represent the standard deviations. The dashed lines indicate the portion of words that are left after removing the corresponding words.

A.6 Evaluation Measures

Table A.2 shows evaluation using many different measures. The results are for the experiments with most texts in Section A.4; average results (and standard deviations when appropriate) for 100 clusterings using K-Means and a "random clusterer". We only give the results for text sets 20ng and Occ. The measures are defined in Section 3.3.

	20ng		Occ	
Measure	K-Means	Random	K-Means	Random
Φ_{intra}	0.035 (0.001)	0.017 (0.000)	0.060 (0.001)	0.013 (0.000)
$\text{sim}(C, C)$	0.014	0.014	0.012	0.012
ρ	0.54 (0.04)	0.07 (0.00)	0.46 (0.02)	0.14 (0.00)
$H(K\|C)$	1.92 (0.11)	4.28 (0.00)	2.71 (0.05)	4.30 (0.00)
$\tilde{H}(K\|C)$	0.44 (0.03)	0.99 (0.00)	0.63 (0.01)	1.00 (0.00)
MI	2.40 (0.11)	0.03 (0.00)	1.60 (0.05)	0.01 (0.00)
NMI	0.57 (0.03)	0.01 (0.00)	0.37 (0.01)	0.00 (0.00)
$H(C)$	4.10 (0.05)	4.32 (0.00)	4.25 (0.02)	4.32 (0.00)
$H(K)$	4.31	4.31	4.31	4.31
p	0.51 (0.03)	0.05 (0.00)	0.27 (0.01)	0.05 (0.00)
r	0.39 (0.03)	0.05 (0.00)	0.31 (0.01)	0.06 (0.00)
RI	0.93 (0.00)	0.90 (0.00)	0.92 (0.00)	0.89 (0.00)
JC	0.28 (0.02)	0.03 (0.00)	0.17 (0.01)	0.03 (0.00)
FM	0.44 (0.03)	0.05 (0.00)	0.29 (0.01)	0.06 (0.00)

Table A.2: Results evaluated using several measures, all defined in Section 3.3. Average values (with standard deviation when appropriate) for 100 clusterings of two different text sets (separated by a double vertical line). The horizontal lines separate internal measures (top), and external measures, divided into "single text measures" (middle), and "pair measures" (bottom).

The first measure is internal. We see that clustering leads to an improvement in average similarity between texts (Φ_{intra} compared to $\text{sim}(C, C)$), and that a very small part of this is from just dividing the set of texts into groups (the result for the random clusterer).

The rest of the measures are external, except for $H(C)$ and $H(K)$ that are not really quality measures; they indicate how balanced the clustering and the categorization are. It is a coincidence that $H(K)$ is the same for the two text sets; they differ in the third decimal.

Remember that a lower entropy, $H(K|C)$ and $\tilde{H}(K|C)$, indicates a better result. The normalized measures are more convenient as their theoretical maximum and minimum values are obvious, although it might not be possible to achieve those. The Rand Index (RI) considers all pairs. As most pairs are in different groups (clusters or categories) the result does not vary that much.

Measures can not be compared across text sets and/or categorizations, as these vary in difficulty[2]. It is only possible to compare different clusterings of the same set to each other; the measures can be used to confirm improvements.

The external measures basically count the number of correctly classified texts (according to the categorization). However, texts are not equally hard to classify.

[2]See Section 4.1 for a description of these text sets.

We prefer the information theoretic external "single text measures" (entropy and mutual information) as these take all categories and the distribution of these over the clusters into account. The external "pair measures" can serve as a complement, while the internal measures provide a fundamentally different perspective.

All measures indicate substantial improvements when using K-Means as compared to using the random clusterer.

A.7 Discussion

The results in the previous sections show that K-Means text clustering is rather robust. To get a high quality clustering we should pick a number of clusters not too far off to some unknown "correct" number, apply it to a rather large number of texts, not remove words that appear in more than a low number of texts, and use a similarity measure like the cosine measure.

This discussion is obviously (only) valid for text sets of the kind we have investigated, and we have applied good preprocessing in the form of lemmatization and compound splitting, or stemming, something which the papers in this thesis suggest that we should. By "having high quality" we mean as compared to the alternatives (the wrong number of clusters, a small text set, remove the wrong words, and use the Cartesian similarity) as measured by the external evaluation measure (NMI).

Notice that the general tendency is similar for Φ and NMI in all experiments where we have used both. This suggests that the vector space model captures something related to content in the sense of what is used to build the categorizations, and that the K-Means algorithm can utilize this information.

Part III

Papers

Paper I

Improving Clustering of Swedish Newspaper Articles using Stemming and Compound Splitting

Improving Clustering of Swedish Newspaper Articles using Stemming and Compound Splitting

Magnus Rosell
Department of Numerical Analysis and Computer Science
Royal Institute of Technology
100 44 Stockholm, Sweden
rosell@nada.kth.se

Abstract

The use of properties of the Swedish language when indexing newspaper articles improves clustering results. To show this a clustering algorithm was implemented and language specific tools were used when building the representation of the articles.

Since Swedish is an inflecting language many words have different forms. Thus two documents compared based on word occurrence (i.e. the vector space model and cosine measure of Information Retrieval) do not necessarily become similar although containing the same word(s). To overcome this we have used a stemmer.

Compounds are regularly formed as one word in Swedish. Hence indexing on words leaves the information in the components of compounds unused. We use the spell checking program STAVA to split compounds into their components.

Newspapers sort their articles into sections such as Economy, Domestic, Sports etc. Using these we calculate entropy for the clusterings and use as a measure of quality.

We have found that stemming improves clustering results on our collections by about 4 % compared to not using it. Compound splitting improves results by about 10 % (by 13 % in combination with stemming). Keeping the original compounds in the representation does not improve results.

1 Introduction

As document collections grow larger the need for more advanced tools to extract information from them increase. Clustering has many possible applications, such as browsing as described in (Cutting *et al.* 92) and grouping of retrieval results as in the search engine Vivisimo[1].

Many natural language processing applications work better when combined with language specific tools. Swedish is poorly known from the IR perspective (Hedlund *et al.* 01). We compare representations of documents in clustering of Swedish newspaper articles using stemming and splitting of compounds.

[1] http://vivisimo.com

2 Representation

Each document is represented by a vector d according to the vector space model of information retrieval (see for instance (vanRijsbergen 79)). We remove stopwords and words that appear only once in the entire collection and apply standard tf×idf-weighting. The cosine measure defines similarity between two documents, d_1 and d_2:

$$\text{sim}(d_1, d_2) = \frac{d_1 \cdot d_2}{\|d_1\| \times \|d_2\|} \tag{1}$$

where $\|v\|$ is the norm of a vector v.

The clusters are represented in the vector space by a centroid, the component wise average of all document vectors in them. Hence the similarity between a document d and a cluster c is the average similarity between the document and all the documents in the cluster: (Steinbach *et al.* 00)

$$\text{sim}(d, c) = \frac{d}{\|d\|} \cdot \frac{1}{|c|} \sum_{d_c \in c} \frac{d_c}{\|d_c\|} = \frac{1}{|c|} \sum_{d_c \in c} \text{sim}(d, d_c) \tag{2}$$

where $|c|$ is the number of documents in cluster c, and d_c is a document in cluster c. The similarity between two clusters (i.e. centroids) is the average similarity between the documents in the two clusters. The similarity between a cluster and itself is a measure of the cohesiveness of the cluster.

Truncation of the cluster centroids (i.e. only using words with high weighting) is proved efficient and even improving results for information retrieval through clustering in (Schütze & Silverstein 97). Preliminary results suggest that this is true in our case too. Hence we use truncation of the cluster centroids to 150 words, making the reasoning in equation (2) only approximately true.

2.1 Stemming

Since Swedish is an inflecting language many words have different forms. Thus two documents compared based on word occurrence (as in our case) do not necessarily become similar although

containing the same word(s). To overcome this problem one can use a stemmer or a lemmatizer when building the representation. Stemming improves precision and recall by 15 and 18 %, respectively, in information retrieval for Swedish (Carlberger *et al.* 01). Using the same stemmer we get improvements in clustering (see section 6).

2.2 Compounds

Compounds are regularly formed as one word, solid compounds, in Swedish. Hence building the representation from words leaves the information in the components unused. According to (Hedlund *et al.* 01) it is important that solid compounds are split into their components in information retrieval for Swedish. A later study (Hedlund 02) 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 natural language.

To split the compounds we used the spell checking program Stava (Kann *et al.* 01) that splits compounds into their components (words, fogemorphemes and some of the derivational infixes and particles) and checks them separately. The components are not lemmatized so we used the stemmer to normalize them although it is not intended for this.

Information retrieval for languages with solid compounds (Swedish, German, Finnish ...) has both benefits and drawbacks compared to information retrieval for other languages, using open compounds. A solid compound as a search query will retrieve documents concerning that specific concept, while an open compound will retrieve documents concerning the components as well as the compound. This is either good or bad depending on the information need. Another issue is how to split compounds combined of more than two words.

While for information retrieval the main purpose of the representation is the possibility to search, in clustering it is to reflect the actual content of texts, or rather to reflect the similarity in content of texts. In information retrieval, how and when to split compounds may be crucial, wheras in clustering this is less important as long as the documents and compounds are processed the same way.

How should the components of a compound be weighted within each document? The corresponding weighting in a language with open compounds (like English, for example) would be a simple frequency weight, i.e. the same weight to each of the components. We compare a few different weightings of the compound and components within the documents in section 6. Naturally one could also give the components different weights based on their place in the compound and perhaps on a list of significance in different contexts etc, but we have not yet considered this.

From an information representation point of view, some compounds should not be split at all, since their components obscure their meaning. This is often the case for frequently used compounds which usually denote common concepts and are lexicalised[2]. The word *godkänd* (approved), for instance, is composed of *god* (good or tasty) and *känd* (known, famous). The meanings of these do not add up to the meaning of the compound, at least superficially.

Similarly a lot of Swedish surnames are compounds made by combining names of things, preferably from nature, such as *Almkvist – alm* (elm) and *kvist* (branch). Many other names, of different kinds, such as names of organizations and places, often are compounds which makes more sense left unsplit.

There are also words that by chance could be analyzed as compounds, like the first name *Svante* that definitely not is a compound of *svan* (swan) and *te* (tea).

We have made a list of about 2000 compounds (wordforms) that should not be split. They are manually collected from a lexicon and the newspaper Svenska Dagbladet[3] (SvD) by examining both frequently used compounds and compounds with frequently used components. We have also made a list of about 3000 names (plus genitive forms) of different kinds that should not be split, collected from SvD and publicly available lists.

Some components do not contribute much to the meaning of (information in) the compounds they are a part of, like *upp* (up) in *uppdela* (split up), where *dela* means divide. Other components can not be used as words on their own, like *sär* (something like *not together* or *apart*) in *särbeskattning* (individual taxation, not as a married

[2] See (Hedlund 02) for a general discussion of compounds in information retrieval

[3] in the KTH News Corpus, see section 5.

couple), where *beskattning* means taxation. We have added 200 such components to our stoplist and stop them when splitting the compounds.

3 K-mean Algorithm

We use the partitioning, non-hierarchical clustering algorithm K-mean, see for instance (Steinbach *et al.* 00). It starts by randomly picking k (the number of desired cluster) documents from the collection to represent the clusters. The clusters are then updated iteratively by letting each document belong to the most similar cluster. We have used ten iterations.

Because of the random initial clusters, K-mean is indeterministic; different runs give different results. We make multiple runs of every clustering and calculate the mean and standard deviation for all measures on the final clustering (the measures are presented in section 4).

4 Measures

Evaluating a clustering is hard. What is a good partition of documents for one purpose, may not be so for another. Furthermore humans probably disagree on what is a good partition for a specific purpose.

There are, as recognized in (Steinbach *et al.* 00), two types of measures on clustering results: *internal quality measures* and *external quality measures*. Internal quality measures are based on the representation in some manner and are not appropriate for comparisons of results due to differences in representation. One example is to use the similarity measure; the similarity of a cluster to itself gives, as mentioned in section 2, a measure of the cohesiveness of the cluster. A weighted average of the similarities of every cluster gives a quality measure of the entire clustering, which we call *cluster self similarity* ((Steinbach *et al.* 00) calls it overall similarity).

We compare the cluster self similarity to the *corpus self similarity*, which we define as the similarity of the entire corpus to itself, treated as a cluster (truncated as described in section 2).

External quality measures are based on some external knowledge, like a categorization or groups of answers to search queries in a search corpus. These measures compare the results with the categorization and therefore depend on the quality of it, leading back to the original question of what is a good partition.

We compare the clusterings by the external quality measure *entropy*. The distribution of the documents from the different categories in the clusters, gives us probabilities p_{ij} that a document picked at random from a cluster i, belongs to a category j. The entropy of each cluster is

$$E_i = -\sum_j p_{ij} \log(p_{ij}) \tag{3}$$

and the weighted average entropy for the entire clustering is

$$E = \sum_i \frac{n_i}{n} E_i, \tag{4}$$

where n_i is the number of documents in cluster i and n is the number of documents in total.

The *Information gain* (InfoGain) of a clustering is a measure based on entropy, used in for instance (Fayyad 98). We have InfoGain $= E_{\text{tot}} - E$, where

$$E_{\text{tot}} = -\sum_j p_j \log(p_j) \tag{5}$$

is the entropy of the entire collection and p_j is the probability that a document picked at random from the entire collection belongs to category j.

5 Corpus

The KTH News Corpus (Hassel 01) contains newspaper articles downloaded from the web. Two of the newspapers, Dagens Nyheter (DN) and Aftonbladet (A), are divided into five sections each. These are used to evaluate the clusterings, as described in Section 4.

Table 1 shows the number of articles in the different text collections. As we have chosen not to use articles with less than 20 words, these are not accounted for. The number of articles from Aftonbladet (A) is so big, that we decided to take three random samples (A1, A2, and A3) instead of using them all. We also made one document collection from each newspaper with all articles containing 150 words or more (DN150 and A150).

The entropy of the collections (E_{tot}), table 2, are almost equal, apart from A150 which has a very uneven distribution of the articles over the categories.

6 Results

Every collection was clustered twenty times to five, ten and twenty clusters with all different representations. The result tables 4, 5, and 7 show

	Ekonomi (Economy)	Inrikes (Domestic)	Kultur (Culture)	Sport (Sports)	Utrikes (Foreign)	Total
DN	1 699	1 549	645	1 717	1344	6 954
DN150	573	807	348	1 067	690	3 485

	Ekonomi (Economy)	Nöje (Amusement)	Sport (Sports)	Sverige (Sweden)	Världen (World)	Total
A	6 765	2 874	8 486	8 503	2 974	29 602
A1	529	502	486	527	456	2500
A2	483	472	532	536	477	2500
A3	477	486	565	483	489	2500
A150	877	143	3486	734	85	5325

Table 1: *Papers: number of articles*

	DN	DN150	A	A1	A2	A3	A150
Entropy	2.255	2.234	2.180	2.320	2.320	2.319	1.458

Table 2: *Entropies (E_{tot}) of the collections*

average improvements in percent with standard deviations.

Table 3 shows the abbreviations for the representations used in the following tables. The stoplist of common single words is used on its own (*stoplist*) and in all other representations. Compound splitting is tried on its own (*splitting*), with stoplist for certain components (*wordStop*), with stoplist for compounds (*compStop*), and with both of them (*bothStop*). Different weighting of components and compounds (*X:Y*) is tried with both stoplists. Stemming is tried on its own (*stemming*) and together with compound splitting using both stoplists and weighting (*X:Y, stem*).

In table 6 the average entropy and standard deviation (20 clusterings) for all text collections and number of clusters are shown. Table 7 gives the corresponding average improvements with standard deviations in percent of the results when using only a stoplist of common single words. We use these to calculate the average improvement for every text collection, every number of cluster and over all the experiments (Total).

In tables 4 and 5 the average change in four measures over all experiments are shown (corresponding to *Total* in table 7).

Stemming improves the internal quality measures dramatically (40 % for cluster self similarity and 70 % for corpus self similarity). Compound splitting improves cluster self similarity and corpus self similarity, but the stoplists of compounds and components diminish this effect, especially for the corpus self similarity. Using only the components in the representation improves both measures. Combining stemming and compound splitting gives the greatest improvement (up to 80 and 115 % respectively).

The external quality measures are also improved. Here stemming gives an improvement of approximately 4 %, while compound splitting gives 10 %. The combination improves results by 13 %.

The standard deviations are very large but in the same order of magnitude for all results (for each measure), around 7 %.

7 Discussion

In this section we discuss the results and try to give plausible explanations to some of them.

The standard deviations are very big but almost equal. They probably originate from the indeterminism of the clustering algorithm, rather than from the constant representations.

In table 7 one can compare the improvements in entropy over the collections. They are the same in principle for all collections, but A150. The reason is probably its very different distribution of articles over the categories, being very heavy on one category (see table 1). This is not ideal for the k-mean algorithm but suggests that further experiments on more varied text collections should be done to make the results we present more reliable.

Stemming works as a dimensionality reduction,

Abbreviation	Explanation
stoplist	stoplist of common single words (used as reference)
stemming	stemming
splitting	splitting of compounds
wordStop	stoplist of certain components
compStop	stoplist of certain compounds
bothStop	stoplists of ceratain compounds *and* components
X:Y	weighting: components by a factor X and compounds by Y
X:Y, stem	as above plus stemming

Table 3: *Explanation of abbreviations for representations*

giving a representation using fewer words, as different wordforms are represented by one stem. When the representation gets smaller the clustering gets faster.

Compound splitting makes the representation bigger and thereby the clustering slower. However, if only the components are used in the representation (the compounds are given no weight at all) the number of words decreases and clustering becomes faster. In combination with stemming this reduction is even bigger, giving the smallest representation.

Splitting of compounds introduces new information into the representation. This should be reflected in the measures in some way although the new information is very noisy as the meaning of some of the compounds not are the sum of the meaning of their components. Introducing a stoplist of such compounds should remove some of this noise.

About 20 % of the morphemes that are interesting for an information representation are found in the compounds. One could expect the upper limit on the improvement to be related to this, perhaps being in the same order of magnitude.

To reach this limit the stoplists of compounds and components need to be extended. Another important factor is which words to consider. A lot of different methods could be used here: extending the stoplist of single words, finding content bearing words by for instance part-of-speech tagging, etc.

7.1 Internal quality measures

The great increase in our similarity-based internal quality measures due to stemming is expected since the weights of individual words in the representation become greater and the words are shared by more documents.

Compound splitting on its own also increases internal quality measures as the representation in these cases gets many new words by which the documents may be equal to each other. When removing common components and compounds this effect is diminished, since the number of common words decreases. To weight the components higher than the compounds improves results since these are more common and are more likely to appear in many documents.

Since compound splitting introduces new (common) words into the representation stemming improves results even more in combination with it.

7.2 External quality measures

The improvement of the clusterings from an information point of view (entropy) does not follow the improvement of the similarity in the representation model (the vector space model).

Stemming gives some improvement since new information is found when words with different forms, but similar in meaning, become equal.

Compound splitting extracts new information and makes the representation more precise and improves results more than stemming. The use of a stoplist for some compounds removes bad information and make the improvements bigger. The stoplist of components does not improve results. These words are very frequent and their importance will be reduced by the idf-weight. Our component stoplist is very short – a longer one could perhaps improve results.

The different weightings of the compounds and components do not change the results in any significant way. We choose the representation with fewest words, i.e. not using the compounds at all. This result also indicates that the representation used for languages with open compounds is rather good.

The combination of compound splitting and

	stoplist	stemming	splitting	wordStop	compStop	bothStop
CluSim	0.00 (5.82)	42.52 (7.33)	11.85 (6.55)	9.96 (6.76)	7.89 (6.76)	8.05 (6.63)
CorpSim	0.0	69.5	10.3	5.9	1.1	−0.2

	0.5 : 1	1 : 0.5	1 : 0	1 : 0, stem	1 : 0.5, stem
CluSim	5.83 (6.44)	9.29 (7.01)	18.97 (7.46)	82.86 (8.42)	59.81 (8.62)
CorpSim	−1.1	1.9	14.3	116.0	83.1

Table 4: Change in internal quality measures in percent (stddev)

	stoplist	stemming	splitting	wordStop	compStop	bothStop
Entropy	0.00 (6.31)	−3.63 (5.33)	−5.66 (6.55)	−5.99 (6.71)	−9.41 (6.53)	−10.45 (6.49)
InfoGain	0.00 (7.54)	4.53 (6.88)	6.09 (7.88)	6.58 (7.30)	10.98 (7.78)	12.08 (7.83)

	0.5 : 1	1 : 0.5	1 : 0	1 : 0, stem	1 : 0.5, stem
Entropy	−9.32 (6.29)	−10.98 (5.98)	−10.14 (6.42)	−12.33 (5.86)	−12.09 (6.29)
InfoGain	10.48 (7.50)	12.49 (7.41)	11.59 (8.00)	14.83 (7.57)	14.62 (8.04)

Table 5: Change in external quality measures in percent (stddev)

stemming improves the result even more than the use of compound splitting alone. This is explained in the same way as the improvements due to stemming alone, extended with components from the compounds.

8 Conclusions

Stemming improves clustering results measured with external quality measures by about 4 %. Further it reduces the representation and thereby makes the clustering faster.

Compound splitting improves results by about 10 % when using a stoplist of compounds that should not be split. It also makes the representation bigger but the stoplist of compounds diminishes this effect.

The combination of compound splitting and stemming improves results by about 13 %.

All improvements have a standard deviation of about 7 %.

Weighting the compound and components differently does not change results. Representing only on the components in combination with stemming gives the smallest representation and the fastest clustering.

The text collections we have used are very similar in distribution over categories. To make the results more reliable, experiments on more varied text collections need to be done.

Acknowledgments

I would like to thank my supervisor professor Viggo Kann and Vetenskapsrådet (The Swedish Research Council) for funding.

References

(Carlberger *et al.* 01) J. Carlberger, H. Dalianis, M. Hassel, and O. Knutsson. Improving precision in information retrieval for Swedish using stemming. In *NODALIDA '01 – 13th Nordic Conference on Computational Linguistics*, 2001.

(Cutting *et al.* 92) Douglass R. Cutting, Jan O. Pedersen, David Karger, and John W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 318–329, 1992.

(Fayyad 98) Paul S. Bradley & Usama M. Fayyad. Refining initial points for K-Means clustering. In *Proc. 15th International Conf. on Machine Learning*, pages 91–99. Morgan Kaufmann, San Francisco, CA, 1998.

(Hassel 01) M. Hassel. Automatic construction of a Swedish news corpus. In *NODALIDA '01 – 13th Nordic Conference on Computational Linguistics*, 2001.

(Hedlund 02) T. Hedlund. Compounds in dictionary-based cross-language information retrieval. *Information Research*, 7(2), 2002.

(Hedlund *et al.* 01) T. Hedlund, A. Pirkola, and K. Järvelin. Aspects of Swedish morphology and semantics from the perspective of mono- and cross-language information retrieval. *Information Processing & Management*, 37(1):147–161, 2001.

(Kann *et al.* 01) V. Kann, R. Domeij, J. Hollman, and M. Tillenius. *Text as a Linguistic Paradigm: Levels, Constituents, Constructs. Festschrift in honour of Ludek Hrebicek*, volume 60, chapter Implementation aspects and applications of a spelling correction algorithm. 2001.

(Schütze & Silverstein 97) H. Schütze and C. Silverstein. Projections for efficient document clustering. In *Proceedings of SIGIR'97*, pages 74–81, 1997.

(Steinbach *et al.* 00) M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.

(vanRijsbergen 79) C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

	stoplist	stemming	splitting	wordStop	compStop	bothStop	0.5 : 1	1 : 0.5	1 : 0	1 : 0, stem	1 : 0.5, stem
A1											
5 clust.	1.59 (0.09)	1.53 (0.09)	1.49 (0.11)	1.46 (0.07)	1.41 (0.11)	1.40 (0.09)	1.45 (0.12)	1.43 (0.11)	1.45 (0.14)	1.38 (0.11)	1.36 (0.13)
10 clust.	1.34 (0.09)	1.25 (0.06)	1.27 (0.07)	1.29 (0.07)	1.20 (0.08)	1.19 (0.07)	1.19 (0.08)	1.19 (0.09)	1.19 (0.09)	1.12 (0.06)	1.11 (0.04)
20 clust.	1.27 (0.05)	1.18 (0.06)	1.16 (0.06)	1.16 (0.05)	1.09 (0.03)	1.10 (0.06)	1.11 (0.04)	1.10 (0.05)	1.10 (0.04)	1.05 (0.04)	1.03 (0.02)
A2											
5 clust.	1.51 (0.10)	1.47 (0.11)	1.49 (0.12)	1.48 (0.11)	1.38 (0.11)	1.40 (0.12)	1.45 (0.08)	1.42 (0.10)	1.40 (0.07)	1.32 (0.12)	1.37 (0.11)
10 clust.	1.30 (0.06)	1.20 (0.08)	1.21 (0.03)	1.23 (0.06)	1.16 (0.09)	1.15 (0.05)	1.15 (0.07)	1.13 (0.06)	1.15 (0.08)	1.10 (0.05)	1.11 (0.06)
20 clust.	1.18 (0.07)	1.09 (0.04)	1.11 (0.04)	1.12 (0.04)	1.06 (0.03)	1.08 (0.04)	1.07 (0.03)	1.05 (0.05)	1.03 (0.04)	1.01 (0.04)	1.02 (0.03)
A3											
5 clust.	1.48 (0.08)	1.43 (0.10)	1.48 (0.10)	1.44 (0.07)	1.41 (0.09)	1.35 (0.10)	1.38 (0.08)	1.34 (0.08)	1.36 (0.08)	1.28 (0.11)	1.29 (0.10)
10 clust.	1.31 (0.09)	1.19 (0.06)	1.28 (0.05)	1.23 (0.06)	1.14 (0.05)	1.16 (0.06)	1.17 (0.06)	1.14 (0.06)	1.16 (0.08)	1.10 (0.07)	1.10 (0.08)
20 clust.	1.21 (0.05)	1.13 (0.03)	1.13 (0.06)	1.15 (0.05)	1.07 (0.06)	1.08 (0.04)	1.07 (0.03)	1.07 (0.03)	1.06 (0.05)	1.03 (0.03)	1.04 (0.04)
A150											
5 clust.	0.67 (0.07)	0.67 (0.04)	0.63 (0.09)	0.63 (0.08)	0.63 (0.09)	0.63 (0.09)	0.62 (0.08)	0.59 (0.08)	0.61 (0.08)	0.62 (0.05)	0.61 (0.07)
10 clust.	0.58 (0.05)	0.60 (0.04)	0.56 (0.04)	0.56 (0.08)	0.54 (0.04)	0.52 (0.04)	0.53 (0.06)	0.52 (0.03)	0.53 (0.04)	0.56 (0.04)	0.56 (0.05)
20 clust.	0.52 (0.02)	0.52 (0.03)	0.49 (0.03)	0.50 (0.03)	0.49 (0.03)	0.49 (0.04)	0.48 (0.03)	0.48 (0.03)	0.49 (0.04)	0.49 (0.03)	0.51 (0.03)
DN											
5 clust.	1.26 (0.08)	1.27 (0.06)	1.21 (0.07)	1.21 (0.08)	1.21 (0.07)	1.15 (0.09)	1.19 (0.06)	1.19 (0.07)	1.19 (0.06)	1.18 (0.08)	1.17 (0.08)
10 clust.	1.13 (0.08)	1.10 (0.06)	1.04 (0.07)	1.03 (0.08)	1.00 (0.07)	0.99 (0.06)	1.00 (0.08)	0.98 (0.06)	0.99 (0.06)	0.96 (0.08)	0.98 (0.09)
20 clust.	0.95 (0.06)	0.89 (0.05)	0.86 (0.07)	0.85 (0.05)	0.86 (0.08)	0.82 (0.05)	0.84 (0.05)	0.81 (0.04)	0.82 (0.03)	0.80 (0.04)	0.81 (0.04)
DN150											
5 clust.	1.22 (0.09)	1.22 (0.05)	1.11 (0.06)	1.17 (0.08)	1.10 (0.06)	1.09 (0.08)	1.10 (0.07)	1.06 (0.06)	1.11 (0.09)	1.12 (0.06)	1.12 (0.08)
10 clust.	1.05 (0.05)	1.02 (0.06)	0.98 (0.08)	0.92 (0.07)	0.93 (0.08)	0.89 (0.07)	0.97 (0.09)	0.92 (0.08)	0.92 (0.09)	0.89 (0.09)	0.89 (0.09)
20 clust.	0.88 (0.05)	0.84 (0.04)	0.81 (0.05)	0.80 (0.07)	0.78 (0.04)	0.79 (0.04)	0.78 (0.02)	0.80 (0.04)	0.80 (0.04)	0.77 (0.05)	0.76 (0.02)

Table 6: *Entropy (stddev)*

	stoplist	stemming	splitting	wordStop	compStop	bothStop	0.5 : 1	1 : 0.5	1 : 0	1 : 0, stem	1 : 0.5, stem
Total	**0.00** (6.31)	**−3.63** (5.33)	**−5.66** (6.55)	**−5.99** (6.71)	**−9.41** (6.53)	**−10.45** (6.49)	**−9.32** (6.29)	**−10.98** (5.98)	**−10.14** (6.42)	**−12.33** (5.86)	**−12.09** (6.29)
A1	**0.00** (5.44)	**−5.72** (4.95)	**−6.60** (5.65)	**−6.63** (4.69)	**−11.91** (5.40)	**−12.00** (5.27)	**−10.58** (5.80)	**−11.59** (5.85)	**−10.97** (6.75)	**−15.72** (5.06)	**−16.86** (5.13)
5 clust.	0.00 (5.50)	−3.32 (5.43)	−6.19 (7.01)	−7.70 (4.66)	−11.11 (6.81)	−11.58 (5.62)	−8.30 (7.77)	−10.08 (6.88)	−8.85 (8.87)	−13.32 (6.77)	−14.54 (8.11)
10 clust.	0.00 (6.35)	−6.55 (4.30)	−5.13 (5.14)	−3.45 (5.27)	−10.44 (5.98)	−10.87 (5.46)	−10.87 (5.66)	−11.19 (6.36)	−11.02 (6.92)	−16.50 (4.55)	−17.43 (3.31)
20 clust.	0.00 (4.25)	−7.27 (5.04)	−8.47 (4.49)	−8.72 (4.05)	−14.20 (2.31)	−13.55 (4.68)	−12.57 (2.91)	−13.51 (3.84)	−13.05 (3.19)	−17.34 (3.18)	−18.59 (1.49)
A2	**0.00** (5.97)	**−6.07** (5.80)	**−4.70** (5.16)	**−4.28** (5.27)	**−9.87** (5.97)	**−9.04** (5.66)	**−8.45** (4.68)	**−10.33** (5.29)	**−10.87** (4.91)	**−14.08** (5.47)	**−12.42** (5.47)
5 clust.	0.00 (6.94)	−2.59 (7.49)	−1.66 (7.67)	−1.81 (7.01)	−8.72 (7.32)	−7.18 (8.12)	−4.16 (5.61)	−6.24 (6.94)	−7.59 (4.95)	−12.34 (7.81)	−9.19 (7.52)
10 clust.	0.00 (4.69)	−7.79 (5.89)	−6.47 (2.65)	−5.52 (4.53)	−10.40 (6.74)	−11.03 (4.00)	−11.73 (5.17)	−13.00 (4.59)	−11.62 (5.88)	−15.04 (3.99)	−14.21 (4.95)
20 clust.	0.00 (6.06)	−7.83 (3.16)	−5.97 (3.72)	−5.51 (3.69)	−10.49 (2.81)	−8.92 (3.79)	−9.47 (2.73)	−11.76 (3.83)	−13.41 (3.63)	−14.85 (3.58)	−13.87 (2.93)
A3	**0.00** (5.60)	**−6.40** (4.88)	**−2.92** (5.35)	**−4.66** (4.54)	**−9.53** (4.76)	**−10.32** (5.19)	**−9.60** (4.13)	**−11.37** (4.45)	**−10.63** (5.32)	**−14.72** (5.27)	**−14.51** (5.46)
5 clust.	0.00 (5.23)	−3.92 (6.84)	−0.37 (6.81)	−2.94 (4.88)	−4.69 (5.76)	−8.73 (6.82)	−6.94 (5.09)	−9.62 (5.40)	−8.09 (5.33)	−13.44 (7.11)	−13.39 (6.51)
10 clust.	0.00 (6.98)	−9.18 (4.27)	−2.23 (4.09)	−6.16 (4.55)	−12.88 (3.52)	−11.45 (4.56)	−10.59 (4.42)	−13.03 (4.79)	−11.51 (6.41)	−15.82 (5.05)	−16.22 (6.08)
20 clust.	0.00 (4.23)	−6.12 (2.52)	−6.17 (4.77)	−4.89 (4.16)	−11.02 (4.73)	−10.76 (3.68)	−11.27 (2.37)	−11.46 (2.71)	−12.28 (3.92)	−14.89 (2.71)	−13.92 (3.16)
A150	**0.00** (8.05)	**1.00** (6.11)	**−5.12** (9.60)	**−4.51** (10.38)	**−6.40** (9.51)	**−7.65** (9.69)	**−8.16** (9.68)	**−9.90** (8.42)	**−7.89** (8.98)	**−5.52** (6.92)	**−4.89** (8.50)
5 clust.	0.00 (9.84)	0.09 (5.95)	−5.50 (13.54)	−5.60 (11.25)	−6.16 (13.88)	−6.21 (13.38)	−7.55 (12.57)	−11.68 (12.38)	−8.60 (11.76)	−6.77 (7.66)	−9.01 (11.15)
10 clust.	0.00 (9.25)	3.20 (6.66)	−3.60 (7.08)	−4.10 (12.93)	−6.97 (6.77)	−10.69 (7.12)	−8.49 (9.77)	−10.75 (5.53)	−9.47 (7.32)	−4.11 (6.84)	−3.85 (8.29)
20 clust.	0.00 (3.47)	−0.28 (5.66)	−6.27 (6.57)	−3.84 (5.45)	−6.06 (5.73)	−6.06 (7.22)	−8.45 (5.24)	−7.28 (5.34)	−5.60 (7.05)	−5.67 (6.17)	−1.80 (4.90)
DN	**0.00** (6.54)	**−2.51** (5.10)	**−7.25** (6.30)	**−7.70** (6.53)	**−8.18** (6.66)	**−11.80** (6.19)	**−9.70** (5.76)	**−11.09** (5.02)	**−10.48** (4.55)	**−12.44** (5.97)	**−11.64** (6.32)
5 clust.	0.00 (6.05)	0.62 (4.43)	−4.24 (5.45)	−3.98 (6.33)	−3.95 (5.19)	−9.40 (7.38)	−5.99 (4.83)	−6.13 (5.65)	−5.69 (4.48)	−6.32 (6.58)	−7.25 (6.10)
10 clust.	0.00 (7.12)	−2.80 (5.14)	−8.31 (6.30)	−9.24 (7.46)	−11.51 (6.29)	−12.63 (5.48)	−11.80 (7.28)	−13.26 (4.90)	−12.67 (5.70)	−15.16 (6.99)	−13.48 (7.79)
20 clust.	0.00 (6.39)	−5.35 (5.66)	−9.20 (7.05)	−9.88 (5.69)	−9.08 (8.15)	−13.37 (5.50)	−11.32 (4.80)	−13.89 (4.43)	−13.09 (3.12)	−15.84 (3.82)	−14.20 (4.67)
DN150	**0.00** (5.93)	**−2.09** (5.02)	**−7.36** (6.23)	**−8.16** (7.01)	**−10.55** (5.81)	**−11.89** (5.82)	**−9.42** (6.17)	**−11.56** (6.03)	**−9.99** (6.94)	**−11.49** (6.29)	**−12.24** (6.25)
5 clust.	0.00 (7.08)	0.44 (4.43)	−8.30 (5.10)	−3.90 (6.42)	−9.48 (5.34)	−10.44 (6.38)	−9.78 (5.74)	−12.59 (5.15)	−8.66 (7.16)	−7.52 (5.02)	−7.57 (6.73)
10 clust.	0.00 (5.23)	−2.46 (6.03)	−6.07 (7.46)	−11.80 (7.14)	−11.06 (7.34)	−14.87 (6.58)	−7.48 (8.65)	−12.58 (7.91)	−11.96 (8.68)	−15.12 (8.12)	−15.20 (8.20)
20 clust.	0.00 (5.28)	−4.25 (4.42)	−7.71 (5.89)	−8.77 (7.44)	−11.12 (4.34)	−10.34 (4.20)	−11.00 (2.55)	−9.51 (4.47)	−9.34 (4.24)	−11.82 (5.24)	−13.94 (2.18)
5 clust.	**0.00** (6.94)	**−1.45** (5.87)	**−4.38** (8.10)	**−4.32** (7.10)	**−7.35** (7.97)	**−8.92** (8.35)	**−7.12** (7.44)	**−9.39** (7.49)	**−7.92** (7.54)	**−9.95** (6.89)	**−10.16** (7.87)
10 clust.	**0.00** (6.77)	**−4.26** (5.46)	**−5.30** (5.71)	**−6.71** (7.56)	**−10.54** (6.23)	**−11.92** (5.64)	**−10.16** (7.09)	**−12.30** (5.80)	**−11.37** (6.89)	**−13.63** (6.11)	**−13.40** (6.70)
20 clust.	**0.00** (5.06)	**−5.18** (4.57)	**−7.30** (5.54)	**−6.94** (5.24)	**−10.33** (5.06)	**−10.50** (5.00)	**−10.68** (3.62)	**−11.23** (4.18)	**−11.13** (4.40)	**−13.40** (4.29)	**−12.72** (3.45)

Table 7: *Change in Entropy in percent (stddev)*

Paper II

Comparing Comparisons: Document Clustering Evaluation Using Two Manual Classifications

Comparing Comparisons: Document Clustering Evaluation Using Two Manual Classifications

Magnus Rosell
KTH Nada
SE-100 44 Stockholm
Sweden
rosell@nada.kth.se

Viggo Kann
KTH Nada
SE-100 44 Stockholm
Sweden
viggo@nada.kth.se

Jan-Eric Litton
MEB, Karolinska Institutet
SE-171 77 Stockholm
Sweden
jan-eric.litton@meb.ki.se

Abstract

"Describe your occupation in a few words", is a question answered by 44 000 Swedish twins. Each respondent was then manually categorized according to two established occupation classification systems. Would a clustering algorithm have produced satisfactory results? Usually, this question cannot be answered. The existing quality measures will tell us how much the algorithmic clustering deviates from the manual classification, not if this is an acceptable deviation. But in our situation, with two different manual classifications (in classification systems called AMSYK and YK80), we can indeed construct such quality measures. If the algorithmic result differs no more from the manual classifications than these differ from each other (comparing the comparisons) we have an indication of its being useful. Further, we use the kappa coefficient as a clustering quality measure. Using one manual classification as a coding scheme we assess the agreement of a clustering and the other. After applying both these novel evaluation methods we conclude that our clusterings are useful.

1 Introduction

Open answers in questionnaires are very expensive to categorize manually. If an automatic clustering algorithm would perform nearly as well it could save both time and money. However, it is very difficult to evaluate a clustering of documents. What is a good partition of a text set varies. For different purposes different partitions may highlight valuable aspects of the texts. In [Frakes and Baeza-Yates, 1992] clustering *evaluation* is described as the attempt to decide which algorithm is appropriate for a specific data set, while *validation* is used to decide whether a clustering result characterizes the data. Both could be assessed using the same clustering quality measures. Evaluation of clustering results must be considered by any user of clustering algorithms and consequently most discussions of clustering contain descriptions of evaluation. See for instance [Steinbach *et al.*, 2000] and [Larsen and Aone, 1999]. Clustering validation techniques have been reviewed in [Halkidi *et al.*, 2001].

When two manual classifications are available new possibilities for clustering evaluation arise. One may compare a clustering with both classifications and the classifications to each other. We may then compare the comparisons: a clustering comparing comparatively well with some useful classification, would indicate that it too is useful.

We present several clustering quality measures and the evaluation methods in Section 4, but prior to that we give a clustering example in Sections 2 and 3. The results of this clustering is presented in Section 5 and discussed in Section 6. In Section 7, finally, we draw some conclusions and mention some possible extensions to our present work.

2 Questionnaire

Karolinska Institutet (Swedish Medical University) administrates The Swedish Twin Registry, the largest twin registry in the world with more than 140 000 twins[1]. In one of the questionnaires all twins born in or before 1958 (approximately 44 000 twins answered the questionnaire) were asked, among other things, to describe their occupation in a few words or sentences. Our interest in clustering of this particular text set is motivated by the fact that in any kind of questionnaire there is a lot of information hidden in the open answers. The huge questionnaires in the Swedish Twin Registry contain valuable (medical, biological, sociological, etc.) information. Manual treatment of these are slow and costly. Clustering has been suggested as a tool for exploration of unknown text sets [Cutting *et al.*, 1992]. The possibility to discover a trend in the open answers of the registry, that perhaps later could be verified by further studies, is of genuine interest. Another, related, possible usage of clustering is as an aid for human classifiers; classifying groups (or clusters) of answers rather than single ones. A clustering could also be used as a basis for creating a new manual classification system.

The answers regarding occupation were manually classified by SCB (Statistics Sweden, `www.scb.se`) following two established occupation classification systems, called AMSYK and YK80. AMSYK is used by AMS (The Swedish National Labour Market Administration) and is based on ISCO88 (The International Standard Classification of Occupations). YK80 was used in The Swedish Population and Housing Census 1980. Both are hierarchical, with groups of related occupations at several levels. A group at a higher level is divided into subgroups at the next lower level. The number of groups at each level in the systems is given in Table 1.

	L1	L2	L3	L4	L5
AMSYK	11	28	114	361	969
YK80	12	59	288		

Table 1: The Classification Systems (number of groups per level)

3 Clustering Method

As we compare the clustering results with hierarchical classifications we prefer to have a hierarchical clustering method, i.e. a method giving a cluster hierarchy as a result. We use the divisive algorithm

[1] See http://www.meb.ki.se/twinreg/index_en.html for more information.

Bisecting K-means [Steinbach *et al.*, 2000] that repeatedly split clusters in two using the K-means algorithm. In our experiment we have made ten iterations of K-means in each split.

We use the vector space model with tf*idf-weighting to represent the texts and the cosine measure for calculating similarity between texts and clusters. The cluster centroids are calculated as the word-wise average of the text vectors in them.

Swedish is an inflecting language with a rich production of solid compounds. Hence, after applying a stoplist, we split compounds using the spell checking program Stava [Kann *et al.*, 2001] and use a Swedish stemmer [Carlberger *et al.*, 2001]. Improvements in clustering results using these methods on Swedish news texts have been reported in [Rosell, 2003]. After all preprocessing 43341 texts remained having on average 9.3 different words (including parts of compounds). There were only 6179 different words in total and each word occurred in on average 65 texts[2].

4 Evaluation

As explained in our introduction, evaluating clustering results is very difficult. The performance of a clustering algorithm may be judged differently depending on which measure one uses. To be more confident in results one should use several measures [Steinbach *et al.*, 2000]. Any new measure of, or view on, clustering quality might add to the understanding of clustering. We introduce the kappa statistics as a means for evaluating clustering results in Section 4.5. In Section 4.4 we present a way to compare comparisons of text set partitions that may use any measure for such comparisons. Before that we give a few such measures.

Measures of clustering quality naturally generalize to quality of any text set partition. We distinguish between manual *classifications* consisting of *classes* and automatic *clusterings* consisting of *clusters*. In the general case we will speak of *partitions* consisting of *groups*. Measures of clustering quality may be divided into internal and external measures [Steinbach *et al.*, 2000]. Internal measures (for instance the similarity measure) are based on the representation. External measures compare any partition with a reference partition (typically a clustering with a manual classification). As we have clustering and two classifications (called AMSYK and YK80, see Section 2) we may make six different comparisons, as shown in Figure 1. For many measures, the comparison of two partitions depends on the direction (which is compared with which) since the number of groups, the number of texts in each group and which texts belong to which groups differ.

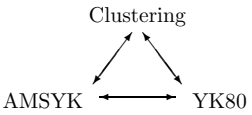


Figure 1: Possible comparisons

[2]After we had removed words that only occur in one text.

4.1 Precision, Entropy and F-measure

In the context of clustering, precision (p) and recall (r) compare each cluster i with each class j in the classification: $p_{ij} = \frac{n_{ij}}{n_i}$, $r_{ij} = \frac{n_{ij}}{n_j}$, where n_{ij} is the number of texts from class j in cluster i, n_i is the number of texts in cluster i and n_j is the number of texts in class j. If the purpose of clustering is to classify clusters of texts rather than single texts the *purity* of each cluster is an appealing measure: $\rho_i = \max_j\{p_{ij}\}$. We may use the weighted average purity over all clusters as a measure of quality of the whole clustering: $\rho = \sum_i \frac{n_i}{n}\rho_i = \frac{n_{\max}}{n}$, where n is the number of texts in the whole text set and $n_{\max}$ 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.

The precision is also the probability that a text drawn at random from cluster i belongs to category j. Hence the entropy of a cluster may be calculated as: $E_i = -\sum_j p_{ij} \log p_{ij}$. The entropy for the whole clustering is the weighted average over all clusters: $E = \sum_i \frac{n_i}{n} E_i$.

The F-measure for a cluster i and class j is: $F_{ij} = \frac{2 \cdot r_{ij} \cdot p_{ij}}{p_{ij}+r_{ij}}$. 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: $F_j = \max_i\{F_{ij}\}$, where the maximum is over all clusters at all levels. The F-measure of the whole clustering hierarchy is: $F = \sum_j \frac{n_j}{n} F_j$. The average is made over the classes rather than the clusters as in the entropy and precision measures. The F-measure tries to capture how well the groups of the investigated partition at the best match the groups of the reference. Purity tries to capture how well the groups of the first on average match those of the reference. While entropy and the precision measures compare flat partitions (which may be a single level of a hierarchy) with another flat partition the F-measure compares an entire hierarchy with a flat partition.

4.2 Pair Measures

The previously defined measures are all based on the distribution of single texts. One may also define measures based on the distribution of pairs of texts. We will use the following such pair measures that has been used for cluster validation (see [Halkidi *et al.*, 2001]): Rand Statistic, $R = \frac{a+d}{m}$, Jaccard Coefficient, $J = \frac{a}{a+b+c}$ and Folkes and Mallows index, $FM = (\frac{a}{a+b}\frac{a}{a+c})^{1/2}$,

- a – number of pairs in the same group in the first partition and in the second,
- b – number of pairs in the same group in the first partition, but in different in the second,
- c – number of pairs in different groups in the first partition, but in the same in the second,
- d – number of pairs in different groups in the first partition and in the second.

The Rand Statistic is the accuracy of pairwise decisions. Pair measures reflect degrees of overlap and are thus not dependent on the direction of the comparison as many other measures.

4.3 "Random Clustering"

One may compare any partition with a random partition of the same text set. For measures calculated for a random partition we add the superscript "rand" (ρ^{rand} for instance). These measures give us

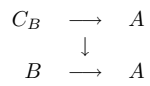


Figure 2: Comparing Comparisons

a hint on how complicated the partition is that the random partition is being compared with. The quality of any other partition may be compared with the quality of the random partition. One example of a measure that does this is the Information Gain: $IG = E^{\text{rand}} - E$. All measures, of course, could be compared with the value for a random partition. We choose not to use those measures that are directly dependent on the number of texts in the groups of the partition under investigation (e.g. F-measure and the pair measures).

4.4 Comparing Comparisons

Having two classifications (A and B) it is possible to compare the comparison of the clustering (C_B) with one classification ($C_B \rightarrow A$) with the comparison of the other classification with the same classification ($B \rightarrow A$), see Figure 2. We mimic categorization B with clustering C_B[3], that is we cluster to as many clusters as groups in B. In the case of an hierarchical classification we cluster to as many clusters as classes in all levels of the hierarchy. For any level of classification B and any level of classification A we may calculate any quality measure m (like those defined in Sections 4.1 through 4.3). Let us call the value of this measure $m_{B\rightarrow A}$. Then take the the clustering C_B, that mimics classification B on some level, and for all levels of classification A we may calculate the corresponding measure $m_{C_B\rightarrow A}$. We want to compare the values $m_{C_B\rightarrow A}$ and $m_{B\rightarrow A}$ and therefore calculate the fraction $f_{m_{B\rightarrow A}} = m_{C_B\rightarrow A}/m_{B\rightarrow A}$, which indicates how well C_B follows B, with respect to measure m, for each particular level to level comparison.

To get an overall value on how well C_B follows B for each measure, in the hierarchical case, we calculate the average fraction, $f^{\text{avg}}_{m_{B\rightarrow A}}$, over all level to level comparisons. Notice that we can not calculate the average of the measures $m_{C_B\rightarrow A}$ and $m_{B\rightarrow A}$ and then the fraction as the number of classes on each level is different in A and B. We don't know how difficult it is to make a partition with x groups compared with one with y groups. In fact, even two partitions with x groups may be very different. Despite this we also calculate the absolute average measures $m^{\text{avg}}_{C_B\rightarrow A}$ and $m^{\text{avg}}_{B\rightarrow A}$ and hope that they will point in the right direction.

Some of the measures are indifferent to the order of the average calculation; those that do not depend explicitly on the structure of the partitions, such as the purity and the information gain. Implicitly they do depend on the structure; they are calculated for each group.

[3]This may be done differently depending on the algorithm. Notice that Bisecting K-means forms a binary hierarchical structure, while the classifications may have different numbers of subgroups for each group.

It is tempting to go on and compare the average fractions[4] and say that clustering is more similar to classification A than B with respect to measure m if $f^{\text{avg}}_{m_{A\to B}} > f^{\text{avg}}_{m_{B\to A}}$. But when the structures of A and B are different the values are not directly comparable.

4.5 Kappa

Finally, we suggest using the kappa statistics (see for instance [Eugenio and Glass, 2004]) to assess agreement between two partitions relative a third, reference partition. This is done by considering the two investigated partitions as classification attempts following the reference. We label all texts in each group in the two partitions with the group from the reference partition which is most frequent in them (remember purity in Section 4.1). The kappa coefficient is: $\kappa = \frac{P(A)-P(E)}{1-P(E)}$, where $P(A)$ is the observed agreement between the two classification attempts and $P(E)$ is the agreement expected by chance.

There is a lot of information relevant to clustering that is not accounted for in this evaluation based on classification attempts (like the portions of documents in a group from other groups in the reference partition etc.). The advantage is the straight forward incorporation of both manual classifications in the evaluation and the possibility of significance assessment through statistic testing. The significance of the kappa coefficient may be assessed by testing the null hypothesis $\kappa = 0$ against the hypothesis $\kappa > 0$ through a well known probability density function [Siegel and Castellan, 1988]. In computational linguistics $\kappa > 0.67$ has often been required to draw any conclusions of agreement at all. However, there has been a number of objections to this standard and less strict evaluations consider $0.20 < \kappa < 0.40$ indicating fair agreement and $0.40 < \kappa < 0.60$ indicating moderate agreement [Eugenio and Glass, 2004]. The values reached in this particular usage are probably rather low as neither the clustering nor the classification are constructed with classification according to the reference in mind.

5 Example Results

In this section we report the results of the clustering example given in Sections 2 and 3 using the methods from Section 4. Tables 4 and 5 show some of the results from four of the six possible comparisons of Figure 1. The labels L1 through L5 in the headings refer to the levels of the classifications (see Table 1) and the numbers to the number of clusters in the clustering. When clustering is involved each value is the average over 50 runs. Each run is continued, so that it is compared at the appropriate number of clusters with the levels of the classification. The average values over the 15 different comparisons of each side of these tables are given in Tables 2 and 3. Table 5[5] gives the relative values to Table 4 (except for the kappa coefficient), that is the fractions $f_{m_{B\to A}}$ of Section 4.4. The average values of these ($f^{\text{avg}}_{m_{B\to A}}$) are given in Table 3, along with the absolute average values.

The average clustering results are almost as good as the result of the classifications for the single text measures. It ranges from 77% and 72% respectively (for the Information Gain) to 86% and 82%

[4] Comparing the comparison comparisons!

[5] The values for the measures on the random partitions are the same as for the corresponding place in Table 4, i.e. the relative values are 1.00.

Measures	YK80 ↓ AMSYK	AMSYK ↓ YK80
ρ^{rand}	0.10	0.16
E^{rand}	5.69	4.73
ρ	0.48	0.65
E	2.69	1.71
IG	3.00	3.02
F	0.41	0.53
R	0.90	0.90
J	0.18	0.18
FM	0.35	0.35

Table 2: AMSYK ↔ YK80 (avg.)

	C_{YK80} ↓ AMSYK		C_{AMSYK} ↓ YK80	
Measures	rel.	abs.	rel.	abs.
ρ^{rand}	1.00	0.10	1.00	0.16
E^{rand}	1.00	5.69	1.00	4.73
ρ	0.85	0.39	0.81	0.51
E	1.38	3.40	1.73	2.56
IG	0.77	2.29	0.72	2.17
F	0.86	0.34	0.82	0.42
R	1.03	0.93	1.00	0.90
J	0.61	0.08	0.41	0.07
FM	0.58	0.19	0.50	0.17
$P(A)$		0.40		0.51
$P(E)$		0.09		0.11
κ		0.34		0.45

Table 3: Clustering → Classifications (average relative values and average absolute values)

respectively (for the F-measures) of that of the two classifications. Clustering gives equally good results with respect to the Rand statistic but as low as an average of 41% in one of the cases for the Jaccard Coefficient. In the level to level comparisons of Table 5 the Jaccard Coefficient ranges from as low as 10% to 145% of the quality of the reference classification.

The kappa values from all level to level comparisons range from 0.18 to 0.64 with average values of 0.35 and 0.45 respectively. Almost all values fall within the fair to moderate agreement assessment of Section 4.5. Using the hypothesis testing of [Siegel and Castellan, 1988] we may conclude significant agreement at a significance level lower than 0.000005 for all level to level comparisons.

In Table 6 some example clusters from one single clustering to 59 clusters are compared with the third level of AMSYK with 114 categories. The words are those with highest tf*idf-weighting translated to English. The purity is the precision of the most frequent category, given in Table 7. It should be compared with the random purity, that is the fraction of all texts belonging to that category.

6 Discussion

Much could be said about the results. Notice first that all measures based on single text distribution (i.e. ρ, E and F) in Table 4 decrease (increase for entropy) with the number of groups in the partition that one is comparing with and increase (entropy decrease) with the number of groups in the partition one is comparing[6]. A similar trend is obvious in the kappa coefficient of Table 5. Both these trends reflect a greater ease to emulate one partition with fewer groups using one with more than the other way around. The changes are not easily related to the number of groups in each partition. (This further convinces us of the difficulties of averaging over all level to level comparisons.) There are no

[6]The underlying absolute values of Table 5 show the same trend.

		YK80 → AMSYK					AMSYK → YK80				
	Measures	L1	L2	L3	L4	L5	L1	L2	L3	L4	L5
L1	ρ^{rand}	0.19	0.14	0.11	0.04	0.03	0.32	0.32	0.32	0.32	0.32
	E^{rand}	3.12	4.32	5.88	7.24	7.91	2.75	2.75	2.75	2.75	2.75
	ρ	0.51	0.39	0.28	0.19	0.16	0.64	0.72	0.81	0.87	0.87
	E	1.81	2.66	3.93	5.06	5.69	1.45	1.10	0.80	0.58	0.48
	R	0.81	0.83	0.82	0.82	0.80	0.81	0.83	0.82	0.82	0.80
	J	0.25	0.17	0.10	0.05	0.04	0.25	0.17	0.10	0.05	0.04
L2	ρ^{rand}	0.19	0.14	0.11	0.04	0.03	0.11	0.11	0.11	0.11	0.11
	E^{rand}	3.12	4.32	5.88	7.24	7.91	4.87	4.87	4.87	4.87	4.87
	ρ	0.67	0.62	0.49	0.35	0.28	0.34	0.51	0.69	0.80	0.81
	E	1.34	1.70	2.45	3.38	3.97	3.10	2.26	1.44	1.02	0.83
	R	0.87	0.93	0.95	0.96	0.94	0.87	0.93	0.95	0.96	0.94
	J	0.16	0.25	0.25	0.17	0.12	0.16	0.25	0.25	0.17	0.12
L3	ρ^{rand}	0.19	0.14	0.11	0.04	0.03	0.06	0.06	0.06	0.06	0.06
	E^{rand}	3.12	4.32	5.88	7.24	7.91	6.57	6.57	6.57	6.57	6.57
	ρ	0.78	0.75	0.68	0.58	0.47	0.26	0.38	0.56	0.72	0.75
	E	0.91	1.15	1.54	2.12	2.65	4.35	3.39	2.23	1.45	1.17
	R	0.88	0.95	0.98	0.99	0.97	0.88	0.95	0.98	0.99	0.97
	J	0.11	0.20	0.31	0.34	0.26	0.11	0.20	0.31	0.34	0.26

Table 4: AMSYK ↔ YK80

such trends in the absolute values of the pair measures (R, J and FM) of Table 4 as the definition of them not include the number of groups in neither of the partitions.

We may observe a distorted trend opposite to the one for the absolute values of the single text measures of Table 4 in all the relative values in Table 5. This trend shows, to no surprise, that the clustering becomes less accurate in emulating the classification with increasing number of classes and decreasing number of clusters.

We stress once again that one probably should be very restrictive in comparing the parallel results presented in Tables 2 and 3 as the classifications are different in structure. Setting that aside it seems quite obvious that our clustering is more similar to the AMSYK classification than the YK80 classification. This is surprising as the clusterings that we compare with AMSYK has fewer clusters than the ones we compare with YK80. So, this may indicate that AMSYK is more suited to describe this particular text set than YK80 is.

7 Conclusions and Further Work

We have presented two methods for clustering evaluation using two manual classifications. The first one assesses the clustering quality by comparing clustering quality measures for the clustering and one of the classifications. This is useful as we expect the high quality of the manual classifications to be reflected in the quality measures. Of course any objections to the appropriateness of the measures one choose to use also apply here. The second method was the use of the kappa coefficient to assess agreement between clustering and a classification considered classification attempts following a second classification.

		$C_{\mathbf{YK80}} \rightarrow$ **AMSYK**					$C_{\mathbf{AMSYK}} \rightarrow$ **YK80**				
	Measures	L1	L2	L3	L4	L5	11	28	114	361	969
	ρ	0.90	0.99	0.95	0.99	0.98	0.87	0.87	0.84	0.81	0.84
L1	E	1.22	1.11	1.07	1.04	1.03	1.28	1.49	1.79	2.19	2.35
12	R	1.03	1.08	1.10	1.11	1.11	0.98	0.99	0.99	0.99	1.01
	J	0.58	0.93	1.17	1.34	1.45	0.67	0.56	0.32	0.21	0.14
	κ	0.25	0.27	0.27	0.19	0.18	0.37	0.49	0.57	0.63	0.64
	ρ	0.85	0.82	0.81	0.86	0.93	1.09	0.87	0.74	0.71	0.75
L2	E	1.40	1.46	1.41	1.27	1.20	1.10	1.34	1.78	2.17	2.29
59	R	1.00	1.00	1.01	1.02	1.02	1.02	1.00	0.99	0.99	1.02
	J	0.36	0.37	0.44	0.68	0.90	1.00	0.59	0.28	0.18	0.12
	κ	0.45	0.42	0.32	0.28	0.22	0.25	0.42	0.46	0.53	0.56
	ρ	0.79	0.76	0.70	0.69	0.75	0.87	0.82	0.70	0.64	0.68
L3	E	1.79	1.82	1.82	1.62	1.42	1.10	1.24	1.59	2.05	2.14
288	R	0.99	0.99	0.99	1.00	1.00	1.02	1.01	1.00	1.00	1.02
	J	0.15	0.15	0.15	0.21	0.31	0.82	0.63	0.32	0.16	0.10
	κ	0.55	0.53	0.45	0.38	0.33	0.25	0.34	0.37	0.44	0.48

Table 5: Clustering → Classifications (relative to Table 4, except for κ)

Both methods use two manual classifications of the same text set, which is not very common. However, the more important the text set is the more likely that you have several classifications. And at the same time the need for new partitions that give new insights increases, as does the demand that these can be properly evaluated. When many classifications exist the methods could compare a clustering to all of them and hence give stronger indications of usefulness.

In our particular clustering example (open answers regarding occupation from the Swedish Twin Registry) clustering compares well with each one of our two classifications compared with the other using the measures presented. Thus we conclude that the partitions provided by our clustering are useful. They could be used as reasonable aid for classification in this particular context and might even provide new insights.

Much could be done to extend this work. We would like to use the quality of the corresponding random partition in the comparisons. This could perhaps be done by construction measures analogous to Information Gain or the kappa coefficient. Further, different kinds of automatic classifications may be compared with clustering and the manual classifications. Manual classifications based on the clusters could be done and evaluated. The clustering may be compared with other questions in the questionnaire.

Acknowledgments

Our thanks go to VR (Vetenskapsrådet – the Swedish Research Council) for funding and to all participants in the project "Infomat" from KTH Nada (department of Computer Science and Numerical Analysis at the Royal Institute of Technology) and the department of Medical Epidemiology and Biostatistics (MEB) at Karolinska Institutet, both in Stockholm, Sweden. The Swedish Twin Registry is supported by grants from VR, the Department of Higher Education, and NIH (grant AG-08724).

	Most Important Words					Docs	Purity
1	auxiliary	care	home	service	old age	983	0.82
2	preschool	child	learn	educationalist	play	810	0.63
3	chauffeur	lorry	drive	taxi	driven	1125	0.80
4	country	agriculture	farm	animal	berry	901	0.47
5	building	carpenter	build	house	concrete	1083	0.65

Table 6: Example Clusters

	Description	Docs	Fraction
1	Personal care and related workers	4916	0.11
2	Pre-primary education teaching associate professionals	661	0.02
3	Motor vehicle drivers	1128	0.03
4	Crop and animal producers	1040	0.02
5	Building and construction workers	1283	0.03

Table 7: Biggest Categories in Example Clusters

References

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

[Cutting *et al.*, 1992] D. R. Cutting, J. O. Pedersen, D. Karger, and J. W. Tukey. 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*, 1992.

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

[Frakes and Baeza-Yates, 1992] W. B. Frakes and R. Baeza-Yates. *Information Retrieval Data Structures & Algorithms*. Prentice Hall, 1992.

[Halkidi *et al.*, 2001] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *J. of Intelligent Information Systems*, 17(2-3):107–145, 2001.

[Kann *et al.*, 2001] V. Kann, R. Domeij, J. Hollman, and M. Tillenius. *Text as a Linguistic Paradigm: Levels, Constituents, Constructs. Festschrift in honour of Ludek Hrebicek*, volume 60, chapter Implementation aspects and applications of a spelling correction algorithm. 2001.

[Larsen and Aone, 1999] B. Larsen and C. Aone. Fast and effective text mining using linear-time document clustering. In *Proc. 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 1999.

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

[Siegel and Castellan, 1988] S. Siegel and N. J. Castellan, Jr. *Nonparametric Statistics for the Behavioral Sciences*. McGraw–Hill, Inc., second edition, 1988.

[Steinbach *et al.*, 2000] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *Proc. Workshop on Text Mining, 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2000.

Paper III

The Impact of Phrases in Document Clustering for Swedish

The Impact of Phrases in Document Clustering for Swedish

Magnus Rosell and **Sumithra Velupillai**
KTH Nada
100 44 Stockholm
Sweden
{rosell, sumithra}@nada.kth.se

Abstract

We have investigated the impact of using phrases in the vector space model for clustering documents in Swedish in different ways. The investigation is carried out on two text sets from different domains: one set of newspaper articles and one set of medical papers.

The use of phrases do not improve results relative the ordinary use of words. The results differ significantly between the text types. This indicates that one could benefit from different text representations for different domains although a fundamentally different approach probably would be needed.

1 Introduction

For document clustering one normally uses the vector space model to represent texts. It is based on the distribution of single words over the texts in a set. We have investigated the impact of introducing phrases in this representation for Swedish in different ways and in different domains. Our hypothesis was that phrases would improve results and that the improvement would be greater for the medical papers than for the newspaper articles as we believe that phrases carry more significance in the medical domain.

To calculate similarity between documents with respect to their phrases we use a word trie (in one set of experiments). This approach has a lot in common with the method presented in (Hammouda and Kamel, 2004). They show improvements in clustering results on web pages using phrases combined with single words, using other algorithms than we. Another related method is the Phrase-Intersection Clustering method which has been proven efficient on web pages (Zamir and Etzioni, 1998). It is based on word-n-grams rather than phrases.

2 Text Sets

We have used a set of 2500 newspaper articles from KTH News Corpus (AB) (Hassel, 2001) and a set of 2422 medical papers from Läkartidningen[1] (Med). In Table 1 some statistics for the sets are given.

We need categorizations of the text sets for the evaluation. The newspaper articles have been categorized by the paper into five sections such as Economy and Sports etc.

The medical papers are categorized with The Medical Subject Headings (MeSH) thesaurus[2]. This thesaurus is (poly)hierarchical with a term and a unique code at each place in it. The terms are not unique and may occur at several places in the hierarchy. There are 15 *broad headings* at the top level.

Each paper has one or more terms from the thesaurus assigned to it. This categorization is very extensive, but also very hard to handle for clustering evaluation. Hence we have made four attempts to flatten and disambiguate it so that each paper belongs to only one of a set of non overlapping categories.

We have made three categorizations where we try to put each document into one of

[1] http://www.lakartidningen.se/
[2] http://www.nlm.nih.gov/mesh/meshhome.html

Text Set	Categories	Documents	Words	Unique Words
AB	5	2500	119401	5896
Med	15, 814	2422	4383169	26102

Table 1: Text Sets

15 categories corresponding to the 15 broad headings. The first, which we call General, is constructed by choosing the broad heading to which most of the MeSH-terms assigned to the paper belongs.

By choosing the broad heading under which the most specific term (the term deepest into the hierarchy) is found for each paper we have constructed the second categorization, which we call Specific.

Many of the papers have as one of the terms assigned to it one or several broad headings. In the third categorization we have chosen this (always one) as the categorization of those papers. The other papers are categorized using the same system as for our categorization Specific. We call this categorization Combined.

We have made a fourth categorization which we call Term. In this every paper is assigned the MeSH-term that has the highest frequency among the terms assigned to it. This leads to a categorization with 817 categories.

The categorizations General and Combined are those that seem most trustworthy. A paper may probably have a very specific term assigned without having its broad heading as the general focus (see Specific). Terms at different levels of the MeSH-hierarchy probably make up an unequal categorization (see Term).

3 Linguistics

We used the grammar checking program Granska[3] to extract nominal phrases from the texts and a stemmer (Carlberger et al., 2001) to stem all words. To prevent very similar but not identical phrases to be deemed unsimilar we removed stopwords within the phrases as well as from the single words.

Swedish solid compounds often correspond to phrases (or compounds) in other languages. We use the spell checking program Stava (Kann et al., 2001) to split them. An earlier study (Rosell, 2003) has proven this to improve clustering results for newspaper articles. We also try to represent the split compounds as phrases and try to split compounds within phrases (see Section 5).

4 Similarity

When calculating the similarity between two documents using phrases two natural alternatives are at hand. Either one chooses to deem phrases similar only if they are identical or one looks at the overlap of words between them. We have tried both. In the first case we have calculated the weight for each phrase in a document as the frequency of its appearance in that document multiplied by the sum of the idf-weight for the single words in it.

To find the overlaps of phrases in documents we have built a trie based on words for each document from the phrases appearing in them. Each phrase is put into the trie in its entire and with all but the first word, with all but the first two words, etc. In each node of the trie we save the number of times it has been reached. To calculate the overlap of phrases between two documents we follow all common paths in the tries and multiply relative appearances in each node weighted by the sum of the idf-weights for the words along the path.[4]

5 Representations

From the phrases and single words we built several different representations. Refer to Table 2 through this section.

Combining all the described possibilities (full phrases or overlap, using split com-

[3] http://www.nada.kth.se/theory/projects/granska/

[4] Compare with Phrase-Intersection Clustering in (Zamir and Etzioni, 1998).

Repr.	Description			
Worst	The worst possible result			
Rand	Random partiton of the set – average for ten iterations			
Best	The best possible result			
1	Only words, stemming			
2	Only words, stemming and splitting of compounds			
3	P	PM	NSP	NSC
4	P	PM	NSP	SC
5	P	PM	SP	NSC
6	P	PM	SP	SC
7	P	POM	NSP	NSC
8	P	POM	NSP	SC
9	P	POM	SP	NSC
10	P	POM	SP	SC
11	P&W	PM	NSP	NSC
12	P&W	PM	NSP	SC
13	P&W	PM	SP	NSC
14	P&W	PM	SP	SC
15	P&W	POM	NSP	NSC
16	P&W	POM	NSP	SC
17	P&W	POM	SP	NSC
18	P&W	POM	SP	SC

Abbr.	Explanation
P	Similarity only between phrases
P&W	Similarity using both phrases and words
PM	Phrase-match
POM	Phrase-overlap-match
SP	Use splitted compounds as phrases
NSP	Do not use splitted compounds as phrases
SC	Split compounds within phrases
NSC	Do not split compounds within phrases

Table 2: Representations

pounds as phrases or not, and split compounds within phrases or not) we get eight different representations based on phrases. By combining[5] these with the ordinary single word representation with split compounds we get eight more. This gives 16 representations (representations 3 through 18 in Table 2). We also made the reference representation (only words, 1) and the representation where solid compounds have been split (2), giving in total 18 different representations.

Finally, for comparison we also try a random "clustering" (Rand) and in the evaluation we present the theoretical worst (Worst) and best (Best) possible results (see Sections 7 and 8).

6 Clustering Algorithm

The clusterings have been made using the divisive algorithm Bisecting K-Means (Steinbach et al., 2000) which splits the worst cluster (i.e. largest) in two, using K-Means, until the desired number of clusters are reached. We have let the K-Means algorithm iterate ten times and for each split we ran it five times and picked the best split (evaluated using the similarity measure). Average results are calculated over ten runs to ten clusters for each representation.

[5]We use equal weight on the two different representations. In (Hammouda and Kamel, 2004) they try different weightings.

7 Evaluation

As we compare different representations we use extrinsic evaluation measures that requires a categorization of the the same text set to compare with. Among the extrinsic evaluation measures that have been used for text clustering are *the purity* and *the entropy*. These measures are well suited for evaluation of single clusters, but for evaluation of whole clusterings *the mutual information* is better. (Strehl et al., 2000)

Consider a text set N 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 $k^{(j)}$.

The γ by κ matrix M describes 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 $k^{(j)}$.

The mutual information of clustering C and

categorization K is:

$$MI(C,K) = \sum_{i=1}^{\gamma}\sum_{j=1}^{\kappa}\frac{m_i^{(j)}}{n}\log(\frac{m_i^{(j)}n}{n_i n^{(j)}}) \quad (1)$$

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

This normalization is theoretical and particular for each clustering-categorization-setting. We want to compare results on different such settings, with different text sets, having varying clustering complexity/difficulty. Therefore we need to normalize with regard to something else.

Since we want to know how much introducing phrases improve results we use the result from a clustering using only words as a reference. By comparing the results with this reference we take the complexity of the different text sets into account.

There are two simple and reasonable ways of normalizing the result using the word clustering result, $MI(C_{word}, K)$. We can divide the result by it:

$$MI_{word}(C,K) = \frac{MI(C,K)}{MI(C_{word},K)}, \quad (2)$$

or we can divide the improvement by the maximum possible improvement from the word clustering result:

$$MI_{imp}(C,K) = \frac{MI(C,K) - MI(C_{word},K)}{MI_{max}(C,K) - MI(C_{word},K)} \quad (3)$$

The first normalization is suitable when we have a decrease in performance. It puts the decrease in relation to the greatest possible decrease. The second normalization is suitable when we have an increase in performance.

8 Results

We present the results of our investigation in Tables 3 and 4. All values are average results over ten clusterings with standard deviation within parenthesis.

The first row of each part of these tables gives the results for the newspaper articles and the following the results on the medical papers compared to the different categorizations. In Table 4 we only give results for representations Term and General as the results for Combined, General and Specific are very similar.

The columns represent the different representations which were described in Section 2 and summarized in Table 2. In Table 3 we present the result for a random "clustering" (the average of 10 random partitions of the text set) and the theoretical worst and best possible results.

9 Discussion

When, in the following discussion, we refer to the results on the medical papers we consider the results on the categorization General (which is very similar to results on Combined and Specific). The results with respect to the categorization Term of the medical papers are a bit different than for the others. As we believe the other categorizations to be better we do not discuss this further.

To split *compounds* in the representation based only on words (representation 2 compared to 1) improve results when clustering the newspaper articles but not when clustering the medical papers. This may be because compounds in the medical papers would need a different analysis. We have also used a stoplist for certain words that should not be split based on other newspaper articles as described in (Rosell, 2003). An optimization for medical compounds here would perhaps improve results.

All variations of clustering using *phrases* performs worse than clustering using only words. Clustering performs worse when using only phrases (representations 3-10) than when using the combination of words and phrases (representations 11-18). Since clustering using words is superior the impact of phrases is diminished in the combined representations (11-18).

Looking at the *representations based only on phrases* (3-10) we see that results on news-

	Measures	Worst	Rand		Best	1		2	
AB	MI	0.000	0.009	(0.003)	2.822	0.947	(0.043)	1.093	(0.084)
	MI_{word}	−100.0%	−99.0%	(0.3%)	198.0%	0.0%	(4.6%)	15.4%	(8.9%)
	MI_{imp}	−50.5%	−50.0%	(0.2%)	100.0%	0.0%	(2.3%)	7.8%	(4.5%)
Combined	MI	0.000	0.038	(0.006)	3.614	0.407	(0.016)	0.415	(0.010)
	MI_{word}	−100.0%	−90.6%	(1.4%)	787.9%	0.0%	(4.0%)	2.0%	(2.4%)
	MI_{imp}	−12.7%	−11.5%	(0.2%)	100.0%	0.0%	(0.5%)	0.3%	(0.3%)
General	MI	0.000	0.041	(0.005)	3.614	0.478	(0.013)	0.486	(0.016)
	MI_{word}	−100.0%	−91.5%	(1.1%)	656.0%	0.0%	(2.7%)	1.7%	(3.4%)
	MI_{imp}	−15.2%	−13.9%	(0.2%)	100.0%	0.0%	(0.4%)	0.3%	(0.5%)
Specific	MI	0.000	0.038	(0.005)	3.614	0.396	(0.010)	0.397	(0.017)
	MI_{word}	−100.0%	−90.4%	(1.2%)	812.6%	0.0%	(2.6%)	0.1%	(4.2%)
	MI_{imp}	−12.3%	−11.1%	(0.1%)	100.0%	0.0%	(0.3%)	0.0%	(0.5%)
Term	MI	0.000	1.450	(0.008)	6.498	1.850	(0.023)	1.868	(0.018)
	MI_{word}	−100.0%	−21.6%	(0.5%)	251.2%	0.0%	(1.2%)	1.0%	(0.9%)
	MI_{imp}	−39.8%	−8.6%	(0.2%)	100.0%	0.0%	(0.5%)	0.4%	(0.4%)

Table 3: Text Clustering Results (stdv)

	Measures	3		4		5		6	
AB	MI	0.067	(0.020)	0.071	(0.017)	0.086	(0.024)	0.080	(0.032)
	MI_{word}	−93.0%	(2.1%)	−92.5%	(1.8%)	−91.0%	(2.6%)	−91.5%	(3.4%)
General	MI	0.112	(0.008)	0.117	(0.012)	0.028	(0.005)	0.030	(0.002)
	MI_{word}	−76.6%	(1.7%)	−75.4%	(2.5%)	−94.2%	(1.1%)	−93.7%	(0.4%)
Term	MI	1.547	(0.020)	1.547	(0.013)	0.574	(0.096)	0.585	(0.022)
	MI_{word}	−16.4%	(1.1%)	−16.4%	(0.7%)	−69.0%	(5.2%)	−68.4%	(1.2%)

	Measures	7		8		9		10	
AB	MI	0.095	(0.020)	0.150	(0.024)	0.071	(0.021)	0.058	(0.010)
	MI_{word}	−90.0%	(2.1%)	−84.1%	(2.5%)	−92.5%	(2.2%)	−93.9%	(1.0%)
General	MI	0.148	(0.011)	0.178	(0.015)	0.031	(0.005)	0.037	(0.025)
	MI_{word}	−69.0%	(2.4%)	−62.7%	(3.1%)	−93.5%	(1.0%)	−92.2%	(5.2%)
Term	MI	1.565	(0.033)	1.607	(0.027)	0.506	(0.045)	0.694	(0.269)
	MI_{word}	−15.4%	(1.8%)	−13.2%	(1.4%)	−72.6%	(2.5%)	−62.5%	(14.6%)

	Measures	11		12		13		14	
AB	MI	0.820	(0.051)	0.809	(0.057)	0.946	(0.078)	0.919	(0.100)
	MI_{word}	−13.4%	(5.4%)	−14.6%	(6.0%)	−0.1%	(8.2%)	−3.0%	(10.6%)
General	MI	0.148	(0.016)	0.168	(0.018)	0.210	(0.013)	0.216	(0.013)
	MI_{word}	−69.0%	(3.4%)	−64.8%	(3.8%)	−56.0%	(2.7%)	−54.9%	(2.8%)
Term	MI	1.562	(0.022)	1.566	(0.021)	1.314	(0.052)	1.336	(0.064)
	MI_{word}	−15.6%	(1.2%)	−15.4%	(1.1%)	−29.0%	(2.8%)	−27.8%	(3.5%)

	Measures	15		16		17		18	
AB	MI	0.746	(0.090)	0.734	(0.063)	0.954	(0.063)	0.940	(0.061)
	MI_{word}	−21.3%	(9.5%)	−22.5%	(6.7%)	0.8%	(6.7%)	−0.8%	(6.4%)
General	MI	0.226	(0.022)	0.230	(0.007)	0.217	(0.029)	0.247	(0.020)
	MI_{word}	−52.8%	(4.5%)	−52.0%	(1.5%)	−54.7%	(6.1%)	−48.3%	(4.3%)
Term	MI	1.642	(0.026)	1.649	(0.033)	1.460	(0.054)	1.486	(0.048)
	MI_{word}	−11.2%	(1.4%)	−10.9%	(1.8%)	−21.1%	(2.9%)	−19.7%	(2.6%)

Table 4: Results for Text Clustering with Phrases (stdv)

paper articles are almost as bad as random clustering for all of them. The performance on the medical papers, on the other hand, is better than random clustering as long as we do not use split compounds as phrases. It is also better here to use the word trie representation (POM) rather than the simple phrase match (PM). In all this is an indication that phrases contain more information in the medical papers than in the newspaper articles.

For the *combined representations* (11-18) the results are much harder to analyze as the word representation is so much better than the phrase representation. The results on the newspaper articles are much better than on the medical papers here. This could be since the phrase representations do not contain as much information for the newspaper articles as for the medical papers and they thereby obscure the clustering to a lesser extent. Concerning the medical papers, all what is stated for the representations using only phrases holds, except that here it is not negative to use the split compounds as phrases (SP). For the newspaper articles there is even a great increase in performance when using

the split compounds as phrases. This could be explained if the phrase representations using split compounds gives no information, which the results for representations 3-10 indicates. There is no reliable difference between the use of simple phrase match and the word trie representations for the newspaper articles as the standard deviation is very high.

No cases show any change in performance when splitting compounds within phrases (SC) or not. The reason for this could be beacuse the amount of compounds within phrases is small.

It is important to bear the great *differences of the two text sets* in mind. The differences in results between them show that clustering works differently on corpora with different contents (i.e. medical text vs. newspaper text). However, this difference might as well to a great extent be explained by other things, such as the structure and size of the texts and the difference of the categorizations. The medical papers are much longer than the newspaper articles. This could in fact explain all of the differences between them regarding information found in the phrases and the compounds. The categorization of the newspaper articles is probably much better than our categorizations of the medical articles.

10 Conclusions and Further Work

Phrases do not improve clustering in Swedish. At least with the representations tried here. The impact of phrases is more obvious on the medical papers. Overlap match between phrases performs better than simple match. It seems to be bad to consider split compounds as phrases and it does not matter whether one splits compounds within phrases or not.

Splitting solid compounds for the ordinary word representation improves results for the newspaper articles and does not make results worse for the medical papers.

The results are very different for the two text types, the newspaper articles and the medical papers. Maybe one would need to develop different representations for different text types. The information found in the phrases of the medical papers could perhaps be exploited using some other representation. But the same information might also be found in the ordinary representation using only words.

Our results are different from those presented in (Hammouda and Kamel, 2004). This is presumably, at least partially, because of differences between Swedish and English. Swedish solid compounds often correspond to phrases in English.

It could be interesting to try other variations of the representations using phrases presented here, but to really use the information that phrases contain relative to mere words a fundamentally different approach is probably needed. One interesting obvious extension of the present work is, however, to look at word-n-grams instead of phrases as these has proven useful in other works.

Acknowledgements

Our thanks go to professor Viggo Kann for supervision, to VR (Vetenskapsrådet – the Swedish Research Council) for funding and to all participants in the project "Infomat" from KTH Nada (department of Computer Science and Numerical Analysis at the Royal Institute of Technology) and the department of Medical Epidemiology and Biostatistics (MEB) at Karolinska Institutet, both in Stockholm, Sweden.

References

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.*

K. M. Hammouda and M. S. Kamel. 2004. Efficient phrase-based document indexing for web document clustering. *IEEE Transactions on Knowledge and Data Engineering*, 16(10):1279–1296. Student Member-Khaled M. Hammouda and Senior Member-Mohamed S. Kamel.

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

V. Kann, R. Domeij, J. Hollman, and M. Tillenius, 2001. *Text as a Linguistic Paradigm: Levels,*

Constituents, Constructs. Festschrift in honour of Ludek Hrebicek, volume 60, chapter Implementation aspects and applications of a spelling correction algorithm.

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. 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*.

A. Strehl, J. Ghosh, and R. Mooney. 2000. Impact of similarity measures on web-page clustering. In *Proc. AAAI Workshop on AI for Web Search (AAAI 2000), Austin*, pages 58–64. AAAI/MIT Press, July.

A. Strehl. 2002. *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*. Ph.D. thesis, The University of Texas at Austin.

O. Zamir and O. Etzioni. 1998. Web document clustering: A feasibility demonstration. In *Research and Development in Information Retrieval*, pages 46–54.

Paper IV

Free Construction of a Free Swedish Dictionary of Synonyms

Free Construction of a Free Swedish Dictionary of Synonyms

Viggo Kann and **Magnus Rosell**
KTH Nada
SE-100 44 Stockholm
Sweden
{viggo, rosell}@nada.kth.se

Abstract

Building a large dictionary of synonyms for a language is a very tedious task. Hence there exist very few synonym dictionaries for most languages, and those that exist are generally not freely available due to the amount of work that have been put into them.

The Lexin on-line dictionary[1] is a very popular web-site for translations of Swedish words to about ten different languages. By letting users on this site grade automatically generated possible synonym pairs a free dictionary of Swedish synonyms has been created. The lexicon reflects the users intuitive definition of synonymity and the amount of work put into the project is only as much as the participants want to.

Keywords: Synonyms, dictionary construction, multi-user collaboration, random indexing.

1 Introduction

The Internet has made it possible to create huge resources through voluntary cooperation of many people. The size of, or effort put into, each contribution does not matter – with many participators the sum may be great and useful. The most well-known example is the free-content encyclopedia Wikipedia[2] that anyone can edit. It has more than a thousand articles in 75 different languages. The English version has about 700,000 articles.

Wiktionary[3] is the lexical companion of Wikipedia. It is, as Wikipedia, a collaborative project, with the aim to produce a free dictionary in every language. The English Wiktionary has about 90,000 articles, and the Swedish about 3,750.

Both Wikipedia and Wiktionary use the copyleft[4] license GNU FDL[5], which means that the content is free to use. This often motivates people to contribute as they know that their work will be available to everyone.

To start a new similar project requires a lot of users that are interested in the matter and want to help. A suitable and very popular Swedish web site is the Lexin on-line dictionary[6]. Our plan was to let the Lexin users cooperate to build a free Swedish dictionary of synonyms.

To construct the dictionary of synonyms we followed these steps:

1. Construct lots of possible synonyms.
2. Sort out bad synonyms automatically.
3. Let the Lexin users grade the synonyms.
4. Analyze gradings and decide which pairs to keep.

The rest of the paper deals with these steps and presents the results so far of the efforts of the Lexin users. The project started in March 2005, and five months later a free Swedish dictionary of synonyms consisting of 60,000 graded pairs of synonyms was completed.

[1] http://lexin.nada.kth.se

[2] Wikipedia (http://www.wikipedia.org/) was founded in January 2001 by Jimmy Wales and Larry Sanger.

[3] http://www.wiktionary.org/

[4] http://www.gnu.org/copyleft/copyleft.html

[5] http://www.gnu.org/copyleft/fdl.html

[6] http://lexin.nada.kth.se

2 Possible synonym pairs

The first step is to create a list of possible pairs of Swedish synonyms. If you have access to a dictionary D_1 from Swedish to another language X and a dictionary D_2 from X to Swedish you can collect possible synonym pairs by translating each Swedish word to X and back again to Swedish, i.e.,

$$\{(w, v) : \exists y : y \in D_1(w) \wedge v \in D_2(y)\}$$

We may also consider only the dictionary D_1 from Swedish to X:

$$\{(w, v) : \exists y : y \in D_1(w) \wedge y \in D_1(v)\}$$

Similarly we may also consider only D_2. The pairs obtained in this way will sometimes be synonyms, but due to ambiguous word senses there will also be lots of rubbish.

If there are dictionaries available between Swedish and other languages one can get lists of word pairs from them using the same method. Such lists can then be used either to complement or to refine the original list. If (w, v) is a pair included in many lists it becomes more probable that w and v are real synonyms.

By using this technique we have constructed a list of 616,000 pairs of possible synonyms.

3 Automatic Refinement

A possible way to improve the quality of the list would be to part-of-speech tag the words and only keep pairs containing words that may have the same word class. We chose not to do this, because words of different word classes could be (seldomly) synonyms, for example words of the word classes participles and adjectives. In retrospect it was a mistake not to remove words of different word classes, because it is annoying for many users to be asked whether for example a noun and a verb are synonymous.

We also refined the list of synonyms using a method called Random Indexing or RI (Kanerva et al., 2000). In RI each word is assigned a random label vector of a few thousand elements. Using these vectors one constructs a co-occurrence representative vector for each word by adding the random vectors for all words appearing in the context of each occurrence of the word in a large training corpus. For each word pair (w, v) the cosine distance between the co-occurrence vectors of w and v is a measure of relatedness between the two words; words that appear in similar contexts get a high value. Synonyms often appear in similar contexts. So this is a suitable method for deciding on whether a pair of possible synonyms are likely to be actual synonyms.

To find as many related words as possible we have used several different corpora. Table 1 gives some statistics for them. The three top sets are extracted from the KTH News Corpus (Hassel, 2001). Aftonbladet and DN are two Swedish newspapers and DI is a daily economic paper. Med is a set of medical papers from Läkartidningen[7] and Parole is a part of the Swedish Parole Corpus[8].

Before building the Random Index we removed stopwords and lemmatized all words. We chose random vectors with 1800 dimensions and eight randomly selected non-zero elements (four ones and four minus ones). When building the context vectors we used four words before and four words after as the context for each word and added the random labels for these context words to the context vector of the center word weighted by 2^{1-d}, where d is the distance between the context word and the center word.

For each text set we then calculated the cosine of all word pairs in the list of possible synonyms and chose the maximum of these as their similarity. Of the 616,000 possible synonym pairs 435,000 appeared in any of the texts sets. Figure 1 shows the number of pairs (the vertical axis) with higher similarity than certain values (horizontal axis).

We chose to remove all pairs with a similarity value lower than 0.1 after studying this figure and some examples. This left 226,000 pairs.

[7] http://www.lakartidningen.se/
[8] http://spraakbanken.gu.se/lb/parole/

Text set	Texts	Words	Lemmas
Aftonbladet	29,602	751,804	34,262
DN	6,954	593,055	63,164
DI	19,488	1,606,743	70,539
Med	2,422	2,146,788	150,627
Parole	–	1,694,556	135,205

Table 1: Text set statistics (stopwords not included)

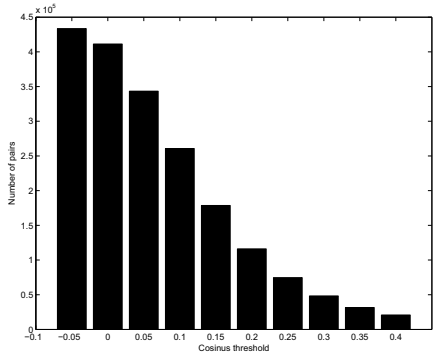


Figure 1: Number of pairs with cosine threshold

4 Manual Refinement

The Lexin on-line dictionary is a very popular web-site for translations of Swedish words to about ten different languages. During the year 2004 the number of lookups in Lexin was 101 millions. This means more than three lookups each second of the year. During 2005 this has increased to five lookups each second.

As the users of Lexin ask language (translation) questions they obviously like the idea of an on-line dictionary. Therefore they are probably motivated to put a small effort in producing a free Swedish dictionary of synonyms.

Many users are of course not native Swedes and are using Lexin to learn Swedish. In order to not bother them with questions about Swedish synonyms we chose to only include the synonym question in the Swedish-English dictionary with Swedish user interface. This will still cover two thirds of the total number of lookups of Lexin.

The Swedish Agency for School Improvement has allowed us to use the Lexin lookup answer web page. As a user gets an answer to a translation question she is also presented with the possibility to answer a question in order to help with the dictionary of synonyms. A question could for example be: *Are 'spread' and 'lengthen' synonyms? Answer using a scale from 0 to 5 where 0 means 'I do not agree' and 5 means 'I fully agree', or answer 'I do not know'.*

When a user have answered this question a web page of the growing synonym dictionary opens and the user may choose to grade more pairs, suggest new synonym pairs, lookup in the synonym dictionary or download the synonym dictionary. Prototypes of the programs taking care of the answers and the synonym dictionary were developed by a student project group at KTH.

5 Synonymity

It is interesting to note that the exact meaning of "synonym" does not need to be defined. The users will grade the synonymity using their intuitive understanding of the concept and the words in the question. The produced dictionary of synonyms will therefore use the People's definition of synonymity, and hopefully this is exactly what the people wants when looking up in the same dictionary.

Of this reason we called the dictionary *The People´s Dictionary of Synonyms.*

6 Abuse

Every web page that invites the public to participate will be subjected to attempts of abuse. Thus the synonym dictionary must have ways to prevent this. Our solution is

threefold. First, many gradings of a pair are needed before it is considered to be a good synonym pair and become possible to lookup in the synonym dictionary.

Second, the pair that a user is asked to grade has been randomly picked from the list of about quarter of a million pairs. The same user will almost never be asked to grade the same pair more than once. If most of the users answer honestly and have an acceptable idea of the synonymity when they think they have, the quality of the synonym dictionary should be good.

Third, the word pairs that users suggest themselves are first checked using a spelling checker and are then added to the long list of pairs, and will eventually be graded by other users. The probability that a user will be asked to grade his own suggested pair is extremely small.

7 Results and Discussion

In five months 2.1 million gradings were made (1.2 million gradings during the first two months). They were distributed over the different grades as shown in Figure 2. The distribution between grades 1–5 is remarkably even. Only the 0 grade is a lot more common than the other grades.

Table 2 gives a few examples of user graded synonyms. Pairs given grade 5 or 4 are very good synonyms. Pairs of grade 3 are synonyms to a less degree, for example *cistern* and *pot*, and pairs of grade 2 are often of different word classes (for example one adjective and one verb). Words of grade 1 are relatad but not synonymous, and grade 0 words are not even (obviously) related.

As the pairs are picked at random from the list some pairs are graded more times than others. Figure 3 shows the distribution of how many times the pairs were graded. Note that when a word has received three 0 gradings it will be removed from the list. Therefore there is a maximum point at 3.

In Figure 4 the number of pairs with different mean gradings are presented. Pairs with very small and very large mean gradings have several times during the period been removed from the list of words to be graded.

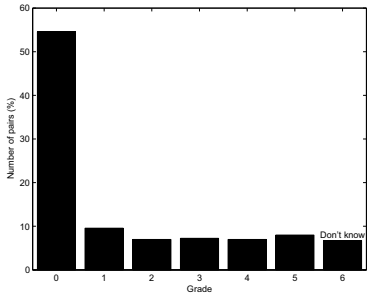


Figure 2: Gradings made by the users

Figure 5 confirms that Random Indexing is useful for automatic refinement. The cosine similarity between pairs that are graded 0 by users is considerably lower than between all pairs on average.

The users could propose new synonym pairs. During the first five months 55,000 pairs (23,000 unique pairs) were proposed. After spelling correction end removal of non-serious words 15,000 pairs remained. These were regularly added to the list of pairs to be graded.

After five months we have 60,000 pairs in the dictionary of synonyms. All these pairs have been given a grade larger than 0 at least three times with a mean value of at least 2.0. When a user makes a lookup in the dictionary the synonyms are presented with their mean grades. This means that this dictionary of synonyms is much more useful than a standard one that only gives a bunch of words that may be more or less synonymous.

The 25,000 best synonym pairs have been publically available for downloading in a simple XML format for about a month. More than 50 downloadings each day are currently being performed, which shows that there is indeed a large need for a free dictionary of synonyms in Swedish.

Many users have commented that there were too many bad pairs. Lots of pairs were graded 0 (not at all synonyms) by all users. After some weeks 25,000 such pairs were re-

5		4		3	
hipp	häftig	bisarr	konstig	cistern	burk
betraktare	iakttagare	dåraktig	idiotisk	folkskola	grundskola
gäng	grupp	hall	foajé	kamrat	väninna

2		1		0	
ansenlig	åtskilligt	bestiga	stiga	hake	kröka
fackförening	union	fatta	följa	ynklig	deltagande
glida	slinka	feja	ren		
hölja	omfatta	hård	tätt		

Table 2: Examples of user graded pairs

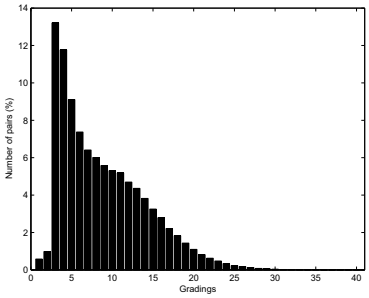


Figure 3: Number of gradings

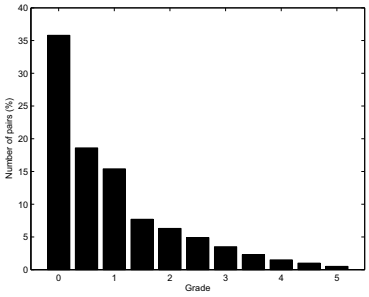


Figure 4: Mean gradings

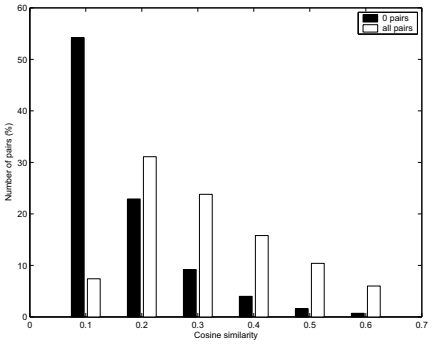


Figure 5: Cosine similarity for pairs graded 0 and for all pairs

moved. Later 60,000 more pairs were removed, improving the quality of the remaining pairs considerably.

8 Lessons learned

The list of suggested synonyms should be huge, as we want to find as many synonyms as possible. But bad pairs irritate the users. Therefore it is important to improve the quality of the list as much as possible. This could be done automatically, using for instance Random Indexing, word class tagging, and other dictionaries, for example for different languages. As the number of answers grows it is also a good idea to remove pairs that often get a zero grading.

9 Conclusions

We have found that it is possible to create a free dictionary of synonyms almost for free. The constructed dictionary is even more useful than a standard one, since the synonyms are presented with gradings.

There is no reason to believe that the method presented in this paper may not be used to create lists of other word relations, such as hypernymy, hyponymy, holonymy and meronymy.

The growing and improving dictionary of synonyms can be found at http://lexin.nada.kth.se/cgi-bin/synlex.

Acknowledgements

We would like to thank the Swedish Agency for School Improvement, and especially Kiros Fre Woldu, for letting us use the Lexin on-line web page to ask users for gradings of synonym pairs.

Thanks to the KTH students Sara Björklund, Sofie Eriksson, Patrik Glas, Erik Haglund, Anna Hilding, Nicholas Montgomerie-Neilson, Helena Nützman, and Carl Svärd for building the dictionary system prototype.

References

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

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.*

Paper V

Revealing Relations between Open and Closed Answers in Questionnaires through Text Clustering Evaluation

Revealing Relations between Open and Closed Answers in Questionnaires through Text Clustering Evaluation

Magnus Rosell[*], **Sumithra Velupillai**[†]

[*] KTH CSC
100 44 Stockholm
Sweden
rosell@csc.kth.se
[†] DSV, KTH - Stockholm University
Forum 100
164 40 Kista
Sweden
sumithra@dsv.su.se

Abstract

Open answers in questionnaires contain valuable information that is very time-consuming to analyze manually. We present a method for hypothesis generation from questionnaires based on text clustering. Text clustering is used interactively on the open answers, and the user can explore the cluster contents. The exploration is guided by automatic evaluation of the clusters against a closed answer regarded as a categorization. This simplifies the process of selecting interesting clusters. The user formulates a hypothesis from the relation between the cluster content and the closed answer categorization. We have applied our method on an open answer regarding occupation compared to a closed answer on smoking habits. With no prior knowledge of smoking habits in different occupation groups we have generated the hypothesis that farmers smoke less than the average. The hypothesis is supported by several separate surveys. Closed answers are easy to analyze automatically but are restricted and may miss valuable aspects. Open answers, on the other hand, fully capture the dynamics and diversity of possible outcomes. With our method the process of analyzing open answers becomes feasible.

1. Introduction

Questionnaires are an important source for new research findings in many scientific disciplines, as well as for commercial exploitation. They may contain both closed ended and open ended questions. The answers to these are called closed and open answers, respectively. Closed answers are restricted to a fixed set of replies, while open answers are not. Statistical methods can be used to study closed answers in large questionnaires. Open answers must be reviewed manually.

Open answers contain valuable and detailed information that is very time-consuming to analyze manually. Methods for assisting the process of analyzing open answers in questionnaires are needed. Natural Language Processing tools could aid such processes, by enhancing the quality of the methods and therefore also the end results.

In Text Mining methods for discovering new, previously unknown information from large text sets are studied (Hearst, 1999). One such method is text clustering, which divides a set of texts into groups (clusters) of texts with similar content. As the content of clusters usually is divers, human investigation and interpretation is needed. The investigation can be aided by the clustering method in several ways. For clustering to be really useful both textual and visual presentation of the clusters should allow the user to explore the results, and interactively focus on interesting and intricate parts.

Collecting large sets of demographic and lifestyle data systematically is central for epidemiological studies. In (Ekman et al., 2006) the feasibility of using web-based questionnaires is discussed. Moving towards e-epidemiology increases the possibilities of conducting large population-based studies immensely, both with respect to cost-efficiency and availability (Ekman and Litton, 2007).

We present a method for hypothesis generation using text clustering, involving human judgement in crucial steps. The method is applied to a large epidemiological questionnaire with promising results.

2. Related Work

Swanson and Smalheiser (1997) describe a method for hypothesis generation by linking possibly related medical literature. Their method exploits existing literature in order to discover previously unknown information and involves user interaction.

In the Scatter/Gather-system (Cutting et al., 1992) clustering is used as a tool for exploration of text sets. Clusterings are presented in a textual format and the user can interactively choose to re-cluster parts of the result, homing in on interesting themes.

To our knowledge, little research has been performed on automatically revealing new information from open answers in questionnaire data. Li and Yamanishi (2001) present a method for analyzing open answers in questionnaires using rule analysis and correspondence analysis. They describe a few other systems, but information about these is not readily found.

Central to all exploration methods is human interaction. Exploration of unstructured information requiers human interpretation.

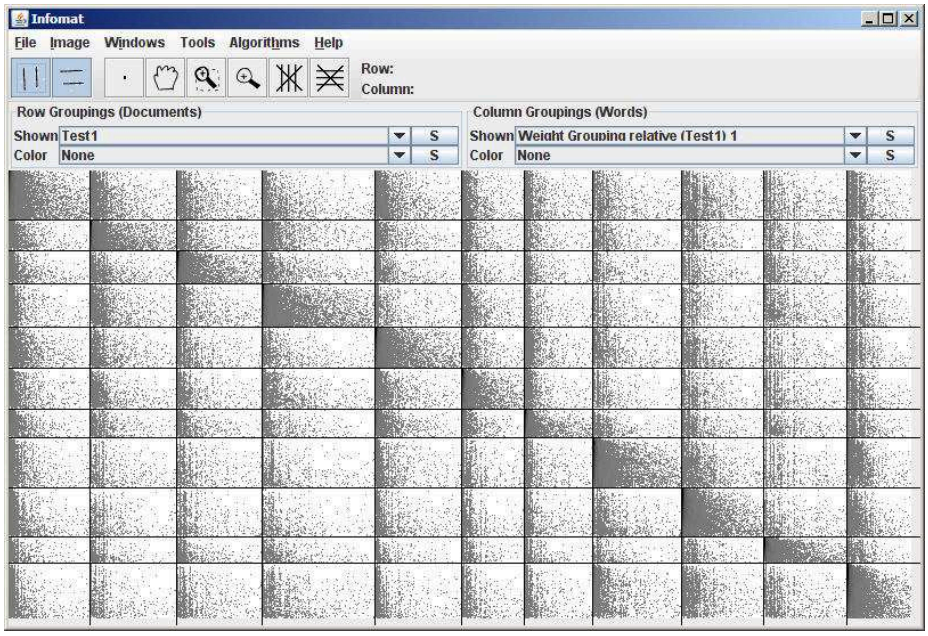


Figure 1: Infomat. 41 549 texts (rows) from the questionnaire presented in Section 4. clustered to 11 clusters (K-Means), represented by 5 978 words (columns). Clusters are separated by lines. The text clusters are sorted according to smoking purity, where those with the highest amount of smokers are found at the top. The texts in each cluster are sorted in order of similarity to the cluster centroid. The words are clustered using the algorithm of Figure 3. Within each word cluster the words are sorted in order of weight in the corresponding text cluster centroid. A distinct diagonal is visible in the 11-by-11 pattern as could be expected. (The opacity of each pixel is proportional to the sum of the weights of its matrix elements.)

3. Method

We propose a method for hypothesis generation from open answers in questionnaires based on text clustering. The method could be described as follows:

1. Cluster the text set
2. Identify interesting clusters
3. Explore cluster contents
4. Formulate potential hypotheses

These steps should be repeated several times. For each repetition different settings (text representation, different clustering algorithms, etc.) could be used. Any recurring hypotheses may be further studied, through literature studies or new surveys.

The proposed method is semi-automatic and can easily be applied using the Infomat tool (see Section 3.1.). User interaction is a central part of the process. Human judgement is required to draw relevant conclusions in each step.

3.1. The Infomat Tool

Infomat[1] is a vector space visualization tool aimed at Information Retrieval (IR) and text clustering in particular (Rosell, 2007). It incorporates the ideas from the Scatter/Gather-system (Cutting et al., 1992), adding new functionality.

Infomat presents information stored in a matrix as a scatter plot, where the opacity of each pixel is proportional to the weight(s) of the corresponding matrix element(s). Here texts are represented in the vector space model by a text-by-word matrix, see Figure 1 for an example.

By sorting the rows (texts) and columns (words) in different ways hidden relationships between the objects may be exposed as visual patterns. Since the rows and columns represent actual objects (texts and words), the visual patterns are possible to comprehend.

Textual information about the matrix can be obtained in different ways. For instance the text(s) and word(s) of each pixel are presented when the cursor is moved over the matrix. It is also possible to zoom in and out, in order to investigate parts of the matrix in more detail, see Figure 2.

Infomat allows the user to cluster both rows and columns. The algorithm introduced in Figure 3 constructs a clustering of the words relative to a text clustering. An extensive description of the content of a text cluster is given by the combination of the visual patterns and the corresponding relative word cluster. (Naturally, reading the actual texts in the clusters can provide further insights.)

[1] http://www.csc.kth.se/tcs/projects/infomat/infomat/

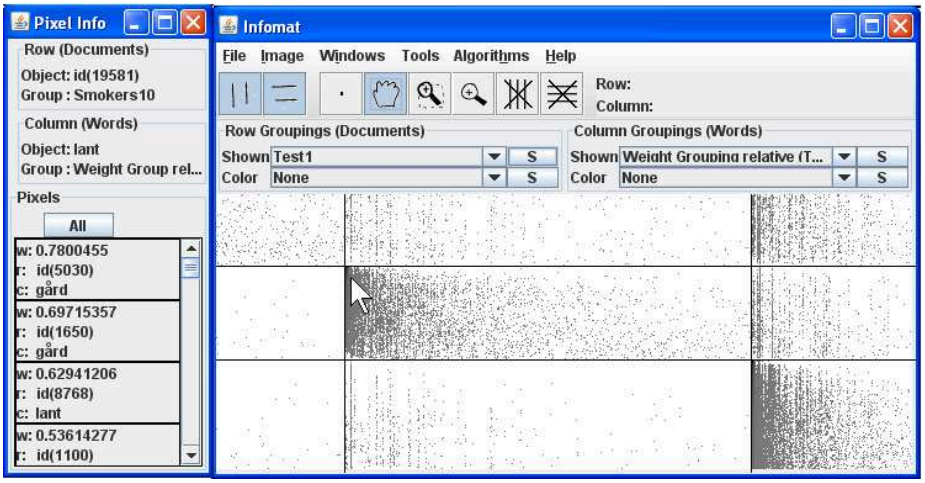


Figure 2: Infomat zoom example. A part of the picture in Figure 1 (centered around the second row and column clusters from the bottom right corner) is shown in the Infomat main window. The *Pixel Info* window to the left gives the matrix elements (weight, text, word) that are represented by the pixel indicated by the cursor. It also shows to which groups (along rows and columns) the texts and words belong. The Swedish word *gård* means "a farm" and *lant* could be translated to "country". There are several more words in the scroll list.

Input: a text set $\mathfrak{T}$,
a set $\mathfrak{W}$ of all words appearing in $\mathfrak{T}$,
a clustering of the texts $\{T_i\}$.

- For each text cluster T_i:
  - calculate the centroid $\overline{T_i}$
  - construct an empty corresponding word cluster W_i
- For each word $w \in \mathfrak{W}$:
  - find $\overline{T_k}$ with maximal weight for w
  - put w in W_k, ordered by its weight in $\overline{T_k}$

Output: a clustering of the words $\{W_i\}$.

Figure 3: *The Relative Clustering Algorithm*

3.2. Identifying Interesting Clusters

A closed answer in a questionnaire may be viewed as a categorization, making it possible to measure clusterings of open answers by ordinary clustering quality measures. If the categorization distribution (measured by a quality measure) in a cluster differs significantly from the entire set the cluster is potentially interesting. Whether a categorization distribution in a cluster differs sufficiently must be judged by the user and depends on the data set, the categorization, etc. In Infomat the clusters can be sorted in order of quality measure value, identifying the clusters with extreme values as the most interesting, see Figure 1 for an example.

In the context of clustering the quality measure *precision* (p) compares each cluster i to each category j in the categorization:

$$p_{ij} = \frac{n_{ij}}{n_i}, \tag{1}$$

where n_{ij} is the number of texts from category j in cluster i, and n_i is the number of texts in cluster i. From the dominating category we get the *purity* for each cluster:

$$\rho_i = \max_j \{p_{ij}\}. \tag{2}$$

The purity is a useful measure here as it is easy to understand. This helps in formulating the hypothesis, see Section 3.4..

3.3. Exploring Cluster Contents

One of the main challenges in text clustering is to describe the contents of the clusters to a user. Other text clustering tools, Scatter/Gather (Cutting et al., 1992) for instance, usually only present a headline consisting of some of the words with the highest weights in the cluster. However, short cluster headlines only provide a partial description of the cluster content, possibly omitting important characteristics.

For each text cluster a corresponding relative word cluster created by the algorithm in Figure 3 constitutes a cluster description. It provides an extensive overview of the cluster content, which can be grasped through browsing with the Infomat tool, as described in Section 3.1..

3.4. Formulating Hypotheses

If a cluster is deemed interesting, as described in Section 3.2., a hypothesis can be formulated from the cluster con-

tent (Section 3.3.). It can be expressed as a relation between the content and the closed answer distribution in the cluster. A hypothesis that recurs over several method iterations is worth investigating further.

3.5. Filtering Hypotheses

The generated hypotheses should be seen as starting points for further analysis. Therefore the exact quality measure values (in the identification of interesting clusters) are not that important – it is the tendencies that matters. Further, the hypotheses might not be novel as they are constructed solely from the investigated questionnaire. A domain expert can make well judged decisions on which tendencies to further pursue.

If the method produces an interesting hypothesis it can be considered useful. Whether the hypotheses holds can only be determined through further studies on material separated from the questionnaire.

3.6. Method Extensions

The method could be extended in several ways. In fact, the more ways the data is processed (revealing the same relations) the better. Several clusterings of rows and columns using different clustering algorithms can provide insights when combined. An especially interesting clustering technique, which is clearly related to the relative clustering algorithm in Figure 3, is *co-clustering* (Dhillon, 2001), where text and word clusterings are constructed simultaneously.

In the identification of interesting clusters, other quality measures, for instance *entropy*, could be used. They could be interesting as an aid in a general investigation of the text set. It is, however, harder to formulate a hypothesis using more abstract and complex measures than purity.

Several closed answers could be used in the identification of interesting clusters, for instance by constructing a categorization of the combination of them. If several open answers are available, clusterings of them could be used as well, considering any one of them a categorization. Further, the Infomat tool allows the user to view a second clustering or categorization along both rows and columns by coloring matrix elements depending on which cluster/category they belong to.

As presented here, the method relies heavily on human judgement. We believe this is unavoidable (and even desirable). Still, perhaps a more automated process could aid the human further in making these judgements. For instance, a predefined scheme of clusterings (and re-clusterings of parts of clustering results) could be run. The results of these could be presented in a condensed form, by for instance only displaying clusters that have been deemed sufficiently interesting automatically. This would make the identification of recurring relations more straightforward.

4. Text Set: Questionnaire

Karolinska Institutet (Swedish Medical University) administrates The Swedish Twin Registry[2], the largest twin registry in the world, containing information about more than 140 000 twins. See (Lichtenstein et al., 2002; Lichtenstein et al., 2006) for a description of the contents and some findings that have come from it.

The registry is based on information from questionnaires containing both closed and open answers. The combination of these provides a large set of valuable (medical, biological, sociological, etc.) information. Manual treatment of this is slow and costly.

The work presented here does not focus on revealing twin-specific information. Instead, the text set is used as an example to show how questionnaire data can be exploited.

	Gender	Smoking
ρ	0.52 (women)	0.71 (non-smokers)

Table 1: Gender and smoking purity for the entire set

	Women	Men
ρ	0.75 (non-smokers)	0.65 (non-smokers)

Table 2: The purity of smokers by gender for the entire set

4.1. An Open Answer on Occupation

Between 1998 and 2002, all twins born in or before 1958 were asked, among other things, to describe their occupation in a few words or sentences (in Swedish). The described occupation is either the last or the primary occupation during the respondent's lifetime. These answers provide a large set of texts with valuable but unaccessible information.

4.2. Representation of the Open Answer

In our experiments we have used the vector space model with tf*idf-weighting to represent the texts and the cosine measure for calculating similarity between texts and clusters. After applying a stoplist, we split compounds using the spell checking program STAVA (Kann et al., 2001) and conduct lemmatization using the grammar checking program Granska[3]. In (Rosell, 2003) improvements in clustering results on Swedish news texts using such techniques are reported.

After preprocessing 41 549 texts remained, having on average 10 different words (including compound parts). There were only 5 978 different words in total and each word occurred in on average 69 texts[4].

4.3. Closed Answers: Gender and Smoking

The questionnaire has several closed answers regarding smoking habits. We have constructed a categorization where we define *smokers* as respondents that have smoked more than a year, and *non-smokers* as all other. There are 12 244 smokers, that is 71% are non-smokers. Table 1 gives the smoking and gender purity for the entire set, and in Table 2 the purity of smokers by gender is shown.

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

[3] http://www.nada.kth.se/theory/projects/granska/

[4] After removing words that only occur in one text.

Clusters	Cluster A	Cluster B	Cluster C	Cluster D
Words	boss (chef)	drive (köra)	assistant (biträde)	country (lant)
	leader (ledare)	chauffeur (chaufför)	care (vård)	forest (skog)
	personell (personal)	car (bil)	home (hem)	farm (gård)
	company (företag)	driver (förare)	food (mat)	cultivator (brukare)
	work- (arbets)	lorry- (lastbils)	old (gammal)	animal (djur)
	task (uppgift)	lorry (lastbil)	cook (laga)	agriculture (lantbruk)
	administrative (administrativ)	truck (truck)	help (hjälpa)	agriculture (jordbruk)
	lead (leda)	taxi (taxi)	service (tjänst)	cow (ko)
	project (projekt)	load (lasta)	sick (sjuk)	worker (arbetare)
	responsibility (ansvar)	road carrier (åkeri)	housing (boende)	works (bruk)
Number of texts	3747	2037	4083	2231
Number of words	3358	2483	2706	2137
ρ(non-smokers)	0.64	0.65	0.76	0.78
ρ(gender)	0.73 (men)	0.90 (men)	0.91 (women)	0.64 (men)

Table 3: Example text clusters from a clustering to 20 clusters of the occupation answers. The two top and two bottom clusters sorted in order of smoking purity. The words are the highest ranked in the corresponding word clusters and have been manually translated from Swedish. The sizes of the text and relative word clusters, as well as the smoking and gender purity are also displayed.

5. Experiment

We have applied our method on the questionnaire, described in the previous section, using the Infomat tool with the K-Means algorithm. The latter since it is fast, which makes the waiting times quite acceptable and the exploration pleasant even on an ordinary home computer.

We clustered the open answers regarding occupation several times to different numbers of clusters. Each time we also applied the relative clustering algorithm (see Figure 3) to the words. An example clustering is given in Figure 1. We also compared each clustering to the closed answer to identify interesting clusters as described in Section 3.2. The text clusters of Figure 1 are sorted in order of purity of smokers – the higher up in the picture the more smokers in the cluster.

We browsed the cluster contents as described in Section 3.1. In this particular example the cluster second from the bottom caught our eye: it has a low percentage of smokers, it is small and seemed to be coherent. In Figure 2 we have zoomed in on this cluster (and its relative cluster). After further browsing at this level we became convinced that a substantial part of the answers described occupations related to farming. Hence, we formulated a potential hypothesis, a relation between the open and closed answer: farmers smoke less than the average.

We repeated the steps of our method several times and observed the same relation in many of the iterations. Table 3 gives a textual presentation of another clustering, where *Cluster D* further supports this discovery.

After only a few hours[5] of exploration, concentrating on the most interesting clusters, we have formulated the following four hypotheses. They correspond well to the four clusters presented in textual form in Table 3.

A People working in leadership positions smoke more than the average.

B People working in the transportation industry smoke more than the average.

C Care workers smoke less than the average.

D Farmers smoke less than the average.

In the next section we try to assess hypothesis D, which was most consistent. The others may in part be explained by the gender distribution, see Tables 2 and 3, and should be studied further.

Studying the text clusters in Table 3, compared to gender regarded as a categorization, four other hypotheses could be formulated. We leave it to the reader to assess the quality of these.

6. Evaluation

With no prior knowledge of smoking habits in different occupation groups we have generated a hypothesis indicating a tendency that farmers smoke less than the average. In order to support or discard it thorough investigations and/or surveys should be performed. Lacking such possibilities, we have tried to find existing comparable surveys on smoking habits (after formulating the hypothesis).

Surveys differ in what they cover, both population sample and questionnaire formulation. The definition of a *smoker* may vary between surveys. Also, there exist many categorization systems for occupations, many of them differing in specificity and structure.

The questionnaire we have derived our hypothesis from is described in Section 4.. We have found the following comparable surveys:

- a Swedish survey by Statistics Sweden (SCB, 2006)
- two U.S.A. surveys (Lee et al., 2004; Lee et al., 2007)
- a European survey (McCurdy et al., 2003)
- an Australian survey (Smith and Leggat, 2007)

The most comparable survey is the one made by Statistics Sweden[6] (SCB), as it is conducted on the Swedish population. SCB is the central government authority for official statistics in Sweden. They provide general population statistics.

[5] Naturally, the amount of time can differ significantly depending on the questionnaire and the purpose of the investigation. The experiment demonstrates that interesting results can be obtained within a reasonable time.

[6] http://www.scb.se

The survey performed by SCB covers the years 1980 – 2005 and the ages 16 – 84. It is given almost every year and the statistics are presented from different aspects: household type, age groups, socio-economic group, etc. Here, smokers are defined as respondents who smoke daily. We focus on the years 1998 – 2003 (the time for the twin questionnaire) and the statistics for farmers as a socio-economic group.

The percentage of smokers overall in the SCB survey is smaller than in the questionnaire, as well as among farmers, see Table 4. However, the tendency that farmers smoke less than the average can also be seen here. Thus, the SCB survey supports our hypothesis.

	All workers	Farmers
SCB 1998 – 99	23.9%	8.7%
SCB 2000 – 01	24.6%	7.2%
SCB 2002 – 03	23.4%	8.9%
Questionnaire	29%	-
Cluster D	-	22%

Table 4: SCB: daily smokers in socio-economic groups in Sweden 1998 – 2003, ages 16 – 84. Questionnaire: smokers (according to definition in Section 4.3.) among twin respondents 1998 – 2002, born in or before 1958. Cluster D: one cluster from a clustering of the open answers in the questionnaire, see Table 3.

All surveys have different occupation categorization systems. The U.S.A. surveys, for instance, utilize a fine-grained categorization of farmers, and the portion of smokers differs between the subgroups. Also, the surveys cover different age groups. The European survey is focused on a younger population sample. Further, different smoker definitions are used. The Australian survey distinguishes *current, ex-,* and *never*-smoker groups. However, the tendency that farmers smoke less than the average is apparent in all surveys.

Considering all differences between the surveys and the twin questionnaire we can confirm our hypothesis, that farmers smoke less than the average. Thus our method is proven successful.

7. Conclusions and Future Work

We have presented a method for hypothesis generation from questionnaires through text clustering evaluation. Using the method we have generated the hypothesis that farmers smoke less than the average, which we have confirmed through literature studies. Normally, a new investigation would need to be performed.

We plan to apply the method on other questionnaires in different domains. Also, it could be applied on other types of data sets containing both textual data and data restricted to predefined values. One interesting example is electronic medical records.

Our method makes it feasible to explore and analyze open answers in large questionnaires, potentially containing hidden information. It provides a means for interactively revealing interesting parts of that information, reducing the manual work load significantly.

Acknowledgements

Our thanks go to VR (Vetenskapsrådet - the Swedish Research Council) for funding and to all participants in the project "Infomat" from KTH CSC (The School of Computer Science and Communication at The Royal Institute of Technology) and the department of Medical Epidemiology and Biostatistics (MEB) at Karolinska Institutet, both in Stockholm, Sweden.

We also thank VINNOVA (Swedish Governmental Agency for Innovation Systems) for funding and all members of the project KEA (Knowledge Extraction Agent) from DSV, KTH-Stockholm University and KTH CSC.

Furthermore, we would like to thank GSLT (The Swedish Graduate School of Language Technology).

The Swedish Twin Registry is supported by grants from VR, the Department of Higher Education, and NIH (grant AG-08724).

8. References

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*.

I. S. Dhillon. 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In *KDD '01: Proc. 7th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 269–274, New York, NY, USA. ACM.

A. Ekman and J. E. Litton. 2007. New times, new needs; e-epidemiology. *European Journal of Epidemiology*, 22:285–292(8).

A. Ekman, P. Dickman, Å. Klint, E. Weiderpass, and J. E. Litton. 2006. Feasibility of using web-based questionnaires in large population-based epidemiological studies. *European Journal of Epidemiology*, 21:103–111(9).

M. 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.

V. Kann, R. Domeij, J. Hollman, and M. Tillenius, 2001. *Text as a Linguistic Paradigm: Levels, Constituents, Constructs. Festschrift in honour of Ludek Hrebicek*, volume 60, chapter Implementation aspects and applications of a spelling correction algorithm.

D. Lee, W. LeBlanc, L. Fleming, O. Gómez-Marín, and T. Pitman. 2004. Trends in US smoking rates in occupational groups: the national health interview survey 1987-1994. *J. Occup. Environ Med.*, 46(6):538–48.

D. Lee, L. Fleming, K. Arheart, W. Leblanc, A. Caban, K. Chung-Bridges, S. Christ, K. McCollister, and T. Pitman. 2007. Smoking rate trends in U.S. occupational groups: The 1987 to 2004 national health interview survey. *J. Occup. Environ Med.*, 49(1):75–81.

H. Li and K. Yamanishi. 2001. Mining from open answers in questionnaire data. In *KDD '01: Proc. 7th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 443–449, New York, NY, USA. ACM.

P. Lichtenstein, U. De faire, B. Floderus, M. Svartengren, P. Svedberg, and N. L. Pedersen. 2002. The swedish twin registry: a unique resource for clinical, epidemiological and genetic studies. *Journal of Internal Medicine*, 252:184–205.

P. Lichtenstein, P. F. Sullivan, S. Cnattingius, M. Gatz, S. Johansson, E. Carlstrom, C. Bjork, M. Svartengren, A. Wolk, L. Klareskog, U. de Faire M. Schalling, J. Palmgren, and N. L. Pedersen. 2006. The swedish twin registry in the third millennium: An update. *Twin Research and Human Genetics*, 9(6):875–882.

S. A. McCurdy, J. Sunyer, J. Zock, J. M. Antó, and M. Kogevinas. 2003. Smoking and occupation from the European community respiratory health survey. *J. Occup. Environ Med.*, 60(9):643–8.

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. 2007. Infomat – a vector space visualization tool. In M. Sahlgren and O. Knutsson, editors, *Proc. of the Workshop Semantic Content Acquisition and Representation (SCAR) 2007*. Swedish Institute of Computer Science (SICS), Stockholm, Sweden. SICS Technical Report T2007-06, ISSN 1100-3154.

Statistics Sweden SCB. 2006. Undersökningarna av levnadsförhållanden (Living condition survey). http://www.scb.se/LE0101.

D. Smith and P. Leggat. 2007. Tobacco smoking by occupation in Australia: Results from the 2004 to 2005 national health survey. *J. Occup. Environ Med.*, 49(4):437–445.

D. R. Swanson and N. R. Smalheiser. 1997. An interactive system for finding complementary literatures: a stimulus to scientific discovery. *Artif. Intell.*, 91(2):183–203.

Paper VI

Part of Speech Tagging for Text Clustering in Swedish

Part of Speech Tagging for Text Clustering in Swedish

Magnus Rosell
KTH CSC
Stockholm, Sweden
rosell@csc.kth.se

Abstract

Text clustering could be very useful both as an intermediate step in a large natural language processing system and as a tool in its own right. The result of a clustering algorithm is dependent on the text representation that is used. Swedish has a fairly rich morphology and a large number of homographs. This possibly leads to problems in Information Retrieval in general. We investigate the impact on text clustering of adding the part-of-speech-tag to all words in the the common term-by-document matrix.

The experiments are carried out on a few different text sets. None of them give any evidence that part-of-speech tags improve results. However, to represent texts using only nouns and proper names gives a smaller representation without worsen results. In a few experiments this smaller representation gives better results.

We also investigate the effect of lemmatization and the use of a stoplist, both of which improves results significantly in some cases.

1 Introduction

Text clustering (see for instance Manning et al. (2008)) aims at dividing a set of texts into groups with coherent content without knowledge of any predefined categories. The result of a clustering could be useful in many different circumstances: it can be used as an intermediate step in a bigger system, or as a tool in its own right, to facilitate exploration of search engine results (Zamir et al., 1997) or for any text set (Cutting et al., 1992).

The result of clustering algorithms is dependent on a definition of a (dis)similarity between the objects. For text clustering the similarity is usually defined via a representation of the texts using some or all the words/tokens that appear in them. Two texts are typically defined as similar if they use the same words. Which words/tokens that are used and how they are preprocessed can have a great effect on the result.

Lemmatization or stemming allows us to treat several related tokens as the same, leading to an increased similarity between texts, using the different forms of a word. Part-of-speech (PoS) tagging can be used to achieve the opposite; separate homographs so that texts are not defined similar when they are using the different meanings of a token.

The rest of this paper is organized as follows. Sections 2 through 4 gives a background to the experiments that we have conducted and present in Section 5. Finally, in Section 6 we summarize and draw some conclusions.

2 Information Retrieval

In Information Retrieval (IR) texts are represented in the common vector space model, see any introductory text, for instance (Manning et al., 2008). Each element of a term-document-matrix is assigned a weight, modeling the importance of the corresponding term to the document. There are several weighting schemes; we use a tf*idf weighting scheme. The similarity between texts (in a search engine: a query and a text) is modeled by a measure that compare their corresponding columns in the matrix. We use the common cosine measure, the cosine of the angle between the vectors.

When building the representation a few preprocessing steps are usually applied after tokenization, depending on the application. Common terms are included in a stoplist and removed, as these usually not contribute to the similarity calculations, being present in many texts. Modern search engines do not use them at all since the

original motivation was to save storage space.

Token (or term) normalization, further, reduces classes of related terms to common representatives to increase similarity between texts that contain these. This includes a predetermined way to handle such things as capital letters, hyphenations, abbreviations, etc. From a linguistic point of view, the most interesting part of term normalization is the use of stemming or lemmatization to collapse morphological variants of a word. Stemming is a more ad hoc method that removes affixes and may reduce word derivations having different parts of speech into the same so called stem, while lemmatization refers to replacing each token with its proper lemma. The effect of using stemming on English texts for search engines is somewhat debated. Some studies have shown improvements, while others even a decrease in performance. There have been improvements reported when using stemming and/or lemmatization for several other languages.

In 2001 Hedlund et al. observed that Swedish was poorly known from an IR perspective. They identify a few properties of the Swedish language that are potential problems in IR (as compared to for instance English):

1. The rather rich morphology (inflectional and derivative).

2. The frequent formation of compounds, which appear as one token. (Of words remaining after the use of a stoplist 10 % are compounds, meaning that more than 20 % of the interesting morphemes are found in compounds.)

3. The high frequency of homographic words. (65% of words in running text)

To address these problems they suggest using natural language processing (NLP) tools: word normalization (stemming or lemmatization) for the morphological variation, compound splitting to extract the information in the parts, and part-of-speech tagging with gender for nouns to disambiguate homographic words. However, search queries are usually short and can be hard to part-of-speech tag correctly.

An IR system for Swedish has to take these issues into consideration. There has been a lot of work done on search engines for both mono and cross language retrieval in recent years. A big comparative study of several European languages is (Hollink et al., 2004). We feel a bit sceptic about the results for Swedish (and Finnish) since they report a substantial increase in performance when removing diacritic characters, indicating that the system does not handle the language very well. They also report substantial improvements using stemming and compound splitting for Swedish.

There are also a lot of studies in CLEF[1] (The Cross-Language Evaluation Forum) that include Swedish, several of which report improvements using morphological analysis.

Carlberger et el. (2001) saw an increase in search engine precision and recall on a newspaper text set when using stemming as compared to not using it. Ahlgren and Kekäläinen (2007) study several user scenarios on newspaper texts and report improvements for morphological analysis, word truncation, and compound splitting.

The results for search engines do not necessarily hold true for other IR methods, such as text clustering.

3 Text Clustering

The vector space model described in Section 2 can be used for text clustering. The reason for doing this is to define similarity between texts and/or groups of texts. Therefore it is not necessary to keep all tokens as in a search engine, where the goal is to be able to retrieve texts containing certain tokens. Hence, the results for search engines are not necessarily valid for text clustering.

Text clustering of Swedish texts has been investigated with respect to stemming and compound splitting (Rosell, 2003) and the use of nominal phrases in the representation (Rosell and Velupillai, 2005). Stemming seems to improve results, but the improvement is small. Compound splitting improves results, but the use of nominal phrases in the representation does not.

We use the K-Means clustering algorithm, see (Jain et al., 1999) for instance. It is fast and efficient and iteratively improves on k centroids (mean vectors) that represent k clusters. In each iteration each text is assigned to the group with the most similar centroid[2]. The algorithm stops when no text changes cluster between iterations. In the experiments presented here we stop after 20 itera-

[1] http://clef-campaign.org/

[2] We do not normalize the centroids when calculating similarity, leading to the average similarity between the text and all texts in the cluster.

tions, as the early iterations contribute more to the result.

In K-Means clustering each centroid contains all terms appearing in all texts of its cluster: terms with high weight in a centroid co-occur a lot in the cluster. If there is coherent content groups in the text set K-Means can find them or something related to them via centroids of cooccurring terms.

Homographs with several meanings may appear in several centroids and be disambiguated by the other terms. Synonyms will likely co-occur with the same words, and hence be present in the same centroid(s). In this work we investigate if these effects can be improved by separating homographs of different parts-of-speech.

4 Clustering Evaluation

Evaluation of text clustering can be either internal or external. Internal measures defines the quality of a clustering using the same information available to the clustering algorithm; the representation and/or similarity measure. As we evaluate different representations these are not appropriate here.

External evaluation can be performed by studying the effect of a clustering on a system that uses clustering as an intermediate step, by asking users for their opinions on the clustering result, or by comparing the result to a known categorization. The later is the easiest, fastest, and least expensive.

Among external measures based on comparisons of a clustering C with a known categorization K the mutual information (MI) is good since it compares the entire distribution of texts over the clusters to the entire distribution of texts over the categories (Strehl, 2002):

$$MI(C,K) = \sum_{i=1}^{\gamma}\sum_{j=1}^{\kappa}\frac{m_i^{(j)}}{n}\log(\frac{m_i^{(j)}n}{n_i n^{(j)}}),$$

where γ and κ are the numbers of clusters and categories, n the total number of texts, n_i the number of texts in cluster $c_i \in C$, $n^{(j)}$ the number of texts in cluster $k^{(j)} \in K$, and $m_i^{(j)}$ the number of texts in both cluster c_i and category $k^{(j)}$.

The normalized mutual information (NMI) takes distributions of texts over the clustering and the categorization into account (Strehl and Ghosh, 2003):

$$NMI(C,K) = \frac{2MI(C,K)}{\sqrt{H(C)H(K)}},$$

where $H(C) = -\sum_{i=1}^{\gamma}\frac{n_i}{n}\log\frac{n_i}{n}$ is the entropy for the distribution of texts over the clusters and $H(K)$ similarly. This makes comparison of evaluations of different clusterings compared to different categorizations theoretically possible. However, the mutual information can never take the inherent linguistic structure of different text sets into account; although comparable in both size of the entire set and distribution over categories, two text sets need not be similarly hard to cluster!

5 Experiments

We have clustered several text sets, see Section 5.1, with several different text representations described in Section 5.2 to a few different numbers of clusters (5, 10, 50) using the K-Means algorithm. All results presented here are average results over 20 runs with standard deviations in parenthesis.

5.1 Text Sets

We have used the following text sets:

KTH News Corpus (Hassel, 2001) is a set of downloaded news texts. The news are from different sources, some of which have a categorization. For the newspapers *Aftonbladet* and *Dagens Nyheter* the texts are categorized into five sections: Domestic/Sweden, Foreign/World, Economy, Culture/Entertainment, and Sports. We have extracted some small text sets from these:

- **A** is some of the texts with 20 or more words from Aftonbladet.
- **DN** is all of the texts with 20 or more words from Dagens Nyheter.

Occ comes from a questionnaire in The Swedish Twin Registry[3]. This text set is the free text answers from 1998 and 2002 to a question about occupation given to the twins born in and before 1958. All answers were categorized by Statistics Sweden[4] (SCB) according to two hierarchical occupation classification systems:

[3] The largest twin registry in the world, containing information about more than 140 000 twins. See (Lichtenstein et al., 2002; Lichtenstein et al., 2006) for a description of the contents and some findings that have come from it and http://www.meb.ki.se/twinreg/index_en.html for more information.

[4] http://www.scb.se

AMSYK is used by AMS (The Swedish National Labour Market Administration) and is based on ISCO88 (The International Standard Classification of Occupations).

YK80 was used in The Swedish Population and Housing Census 1980.

Table 1 gives the number of categories on each of the levels in the classification systems. For the evaluation of these experiments we have used the second level of both.

	L1	L2	L3	L4	L5
AMSYK	11	28	114	361	969
YK80	12	59	288		

Table 1: The Occupation Classification Systems (number of categories per level)

	Text Sets		
	A	DN	Occ
Texts	2424	6395	41949
Categories	5	5	28, 59
$H(K)/\log(\kappa)$	1.00	0.97	0.90, 0.83
Word Forms	12071	37725	17594
Forms/Text	52.29	97.41	15.60
Texts/Form	10.50	16.51	37.20
Lemmas	9050	26451	13873
Lemmas/Text	48.84	88.13	13.70
Texts/Lemma	13.08	21.31	41.29

Table 2: Text Set Statistics

We have used the grammar checking program `Granska`[5] (Domeij et al., 1999) for tokenization, lemmatization, and to tag each word with its part-of-speech. Table 2 gives some statistics for the text sets after preprocessing to word forms (including delimiters) and lemmas. The number of texts, tokens, and the average number of unique token per text and texts per unique token. We also give the number of categories and the "evenness" of the categorization: $H(K)/\log(\kappa)$, which is 1 for a categorization where all categories have equal size, and lower for other cases.

As can be seen there is a significant decrease in tokens when using lemmas instead of word forms. Even if this does not improve the results it improves the storage requirements for the representations.

5.2 Representation

We have evaluated several different representations, which we describe briefly here. In the next section (Section 5.3) we present the results.

`Granska` outputs among other things a tokenization that contains word forms, lemmas, the part-of-speech for each token, and some delimiters. The part-of-speech classes are given in Table 3 and is an adaption (Carlberger and Kann, 1999) of the the tag set in the Stockholm-Umeå Corpus (SUC) (Källgren and Eriksson, 1993).

We have used all the tokens in the representation we call *Full* with either word forms or lemmas (*Word Form* and *Lemma* in the tables). To reduce the Full representation one can use either a stoplist or only consider tokens that get a proper wordclass as their part of speech. The *All wordclasses* representation uses all tokens with these, except the delimiters.

For the *Stoplist* representation we removed tokens according to the Swedish stoplist of the snowball stemmer[6], plus all numbers, and words shorter than three characters and longer than 20.

To separate homographs by their part-of-speech we create new features by concatenating the lemma with its part-of-speech tag (*Lemma + PoS*), for instance: "och_kn", "spela_vb", "mittback_nn". We compare the results for this representation to the one using only the lemma. To separate even more homographs we use the gender for nouns as well (*Lemma + PoS + Gender*).

Most parts of speech in Table 3 contain only words that are usually in a stoplist. We have concentrated on the largest wordclasses, as these are also the ones that convey content in an obvious way. In the result tables we indicate which we have used by the abbreviations in Table 3.

When the representation is constructed we remove terms that appear in only one text as these do not contribute to the similarity calculations. We also remove texts that only contain one term.

5.3 Results

We present some results for text set DN in Table 4, and some of the results for text set Occ evaluated against the second level of the AMSYK categorization system in Table 5. The results for text set

[5] http://www.nada.kth.se/theory/projects/granska/

[6] http://snowball.tartarus.org/

Abbreviation	Part-of-Speech	Example
nn	noun	bil
pm	proper name	Lars
jj	adjective	grön
rg	number	12
ro	cardinal number	första
vb	verb	springa
ab	adverb	mycket
in	interjection	ja
ha	interrogative/relative adverb	när
dt	determiner	den
hd	interrogative/relative determiner	vilken
ps	possesive pronoun	hennes
hs	interrogative/relative possessive	vems
pn	pronoun	hon
hp	interrogative/relative pronoun	vem
sn	subordinating conjunction	om
kn	coordinating conjunction	och
pp	preposition	till
pc	participle	springande
pl	particle	om
uo	foreign word	the
an	abbreviation	d.v.s.
ie	verb base form marker	att
dl	delimiter	.

Table 3: Part-of-Speech Tags used in `Granska`

A are very similar to the ones for DN, and also the results on text set Occ evaluated against the YK80 categorization system (level 2) are very similar to the results evaluated to the AMSYK categorization.

The tables are divided into sections vertically for different numbers of clusters and horizontally for which features are used in the text representation: Word Form, Lemma, Lemma + PoS, or Lemma + PoS + Gender. Other aspects of the representation are presented as rows; which of the features are used in the representation.

The result of each experiment (20 K-Means clusterings of a particular representation) is presented with two values: the average NMI with standard deviations in parenthesis, and the number of features the representation gives rise to. As we remove texts that have one or fewer features some of the clustering are performed on fewer texts than are presented in Table 2. The number of texts that are removed are under on per cent in all cases.

Most differences are well within the standard deviations and should therefore not be considered significant. The representations are kept constant in the experiments; the varying results are due to the indeterministic K-Means algorithm.

5.4 Discussion

Our attempt to enhance the representation by introducing the part-of-speech tags (and gender) fails miserably. There are no interesting tendencies pointing to any improvements compared to using only lemmas, see Tables 4:b, 4:c, and 5:b. The effect of keeping only some parts-of-speech in the representation is not surprising: adjectives, verbs, and adverbs are not very good, while the nouns and proper names are as good on their own as all parts-of-speeches together. For five clusters on the Occ text set it is even better to only keep the large word classes than using them all (Table 5:b).

We have not tried a combination of the word form and the part-of-speech tag. This would have resulted in a representation with even more features, but might have given better results than the word forms on their own.

The lemmatization might address the homograph problem to some extent in addition to the morphological variants. An other explanation is that the cooccurence statistics gathered in the centroids is quite effective in separating homographs, and is not very dependent on which representation is used. Regardless of whether any of these two explanations are true, a representation extended with PoS tags does not improve results.

The comparison between word form and lemma representation in Tables 4:a and 5:a contains some interesting results. It is almost always beneficial to use lemmatization, and most times it improves results a lot. For text set DN it does not improve results significantly when clustering to five clus-

		Word Form		Lemma	
Clusters	Representation	NMI	Features	NMI	Features
5	Full	0.44 (0.05)	37725	0.52 (0.05)	26466
	Stoplist	0.52 (0.04)	35888	0.51 (0.04)	25604
	All wordclasses	0.47 (0.06)	37705	0.49 (0.04)	26451
50	Full	0.28 (0.01)	37725	0.35 (0.01)	26466
	Stoplist	0.28 (0.01)	35888	0.35 (0.01)	25604
	All wordclasses	0.28 (0.01)	37705	0.35 (0.01)	26451

a) Word Form vs. Lemma

		Lemma		Lemma + PoS	
Clusters	Representation	NMI	Features	NMI	Features
5	All wordclasses	0.49 (0.04)	26451	0.51 (0.04)	27532
	nn, pm, jj, vb, ab	0.50 (0.04)	25923	0.52 (0.05)	26767
	nn, pm	0.54 (0.04)	19507	0.55 (0.05)	19940
	jj, vb, ab	0.28 (0.02)	6729	0.29 (0.02)	6827
	jj, ab	0.20 (0.01)	4231	0.19 (0.01)	4285
	vb	0.27 (0.02)	2542	0.27 (0.02)	2542
50	All wordclasses	0.35 (0.01)	26451	0.34 (0.01)	27532
	nn, pm, jj, vb, ab	0.35 (0.01)	25923	0.34 (0.01)	26767
	nn, pm	0.37 (0.01)	19507	0.37 (0.01)	19940
	jj, vb, ab	0.24 (0.01)	6729	0.24 (0.01)	6827
	jj, ab	0.17 (0.01)	4231	0.17 (0.00)	4285
	vb	0.19 (0.00)	2542	0.19 (0.01)	2542

b) Lemma vs. Lemma + PoS

		Lemma + PoS + Gender	
Clusters	Representation	NMI	Features
5	All wordclasses	0.52 (0.04)	27612
	nn, pm, jj, vb, ab	0.50 (0.05)	26847
	nn, pm	0.51 (0.06)	20020
50	All wordclasses	0.34 (0.01)	27612
	nn, pm, jj, vb, ab	0.35 (0.01)	26847
	nn, pm	0.37 (0.01)	20020

c) Lemma + PoS + Gender

Table 4: Some Results for Text Set DN (about 6400 news articles)

ters, but it does not worsen results. The biggest improvement is for text set Occ clustered to five clusters, more than 50 % on average (standard deviation of about 20 %).

The stoplist improves results for text set Occ, but not for DN. It is particularly in combination with lemmatization, when clustering to few clusters that this can be seen. Perhaps the stop words obscure the representation more in the short texts of Occ. To use only the tokens that have proper wordclasses (All wordclasses) does not improve results. The Full representation does, however, not contain many other tokens in the first place.

Lemmatization effects all words/tokens in the representation. We expected that this global influence should be more obvious in results than the use of a stoplist, which is more local. However, the stop words adds noise; making all texts a bit similar, something which seems to be more important for short texts.

The clustering achieves better results when the number of clusters are roughly the same as the number of categories in the categorization used for the evaluation[7], regardless of the representation. It seems hard to improve results for this "optimal" number of clusters using the different representations we try here.

In these experiments we have used almost all words/tokens as features. It is possible to remove a lot of the features without getting worse results. We have tried a few versions were we remove words that appear in few documents. The general tendencies are still the same. Most notably there is nothing to be gained from using the part of speech tags.

Although results do not always improve with the use of lemmatization and a stoplist they never

[7]This is not surprising, considering the definition of NMI. For measures considering only the quality of any single cluster (not the entire clustering) the quality usually improves with more and smaller clusters.

		Word Form		Lemma	
Clusters	Representation	NMI	Features	NMI	Features
5	Full	0.10 (0.02)	17594	0.15 (0.02)	13916
	Stoplist	0.13 (0.02)	16378	0.25 (0.02)	13200
	All wordclasses	0.09 (0.01)	17546	0.15 (0.02)	13873
50	Full	0.25 (0.01)	17594	0.29 (0.01)	13916
	Stoplist	0.29 (0.01)	16378	0.33 (0.01)	13200
	All wordclasses	0.26 (0.01)	17546	0.30 (0.01)	13873

a) Word Form vs. Lemma

		Lemma		Lemma + PoS	
Clusters	Representation	NMI	Features	NMI	Features
5	All wordclasses	0.15 (0.02)	13873	0.15 (0.02)	14151
	nn, pm, jj, vb, ab	0.20 (0.02)	13565	0.20 (0.03)	13704
	nn, pm	0.23 (0.02)	10834	0.23 (0.01)	10841
50	All wordclasses	0.30 (0.01)	13873	0.30 (0.01)	14151
	nn, pm, jj, vb, ab	0.31 (0.01)	13565	0.31 (0.01)	13704
	nn, pm	0.31 (0.01)	10834	0.31 (0.01)	10841

b) Lemma vs. Lemma + PoS

Table 5: Some Results for Text Set Occ (about 42000 short texts)

deteriorate. On the other hand sometimes results improve a great deal. If a minimal representation is required one should consider using only nouns and proper names.

6 Conclusions and Further Work

We conclude that part of speech tagging does not improve results for text clustering of Swedish texts. However, to use only nouns and proper names in the representation often leads to results comparable to using all words, and may decrease the number of features significantly.

Lemmatization improves results a lot in several experiments. To use a stoplist improves results sometimes; in our experiments for short texts.

The cooccurence information in the K-Means centroids is obviously very good at handling homographs as no improvement in clustering results was achieved when introducing lemma-PoS-tag features.

As nouns seems to be very important for clustering, pronoun resolution could perhaps be interesting. However, it would just alter the weighting for the nouns and thus not affect the similarity between texts quite as radically as lemmatization and part of speech tagging.

References

P. Ahlgren and J. Kekäläinen. 2007. Indexing strategies for swedish full text retrieval under different user scenarios. *Inf. Process. Manage.*, 43(1):81–102.

J. Carlberger and V. Kann. 1999. Implementing an efficient part-of-speech tagger. *Softw. Pract. Exper.*, 29(9):815–832.

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*.

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*.

R. Domeij, O. Knutsson, J. Carlberger, and V. Kann. 1999. Granska – an efficient hybrid system for Swedish grammar checking. In *Proc. 12th Nordic Conf. on Comp. Ling. – NODALIDA '99*.

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

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.

V. Hollink, J. Kamps, C. Monz, and M. De Rijke. 2004. Monolingual document retrieval for european languages. *Inf. Retr.*, 7(1-2):33–52.

A. K. Jain, M. N. Murty, and P. J. Flynn. 1999. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323.

G. Källgren and G. Eriksson. 1993. The linguistic annotation system of the stockholm: Umeå corpus project. In *Proceedings of the sixth conference on*

European chapter of the Association for Computational Linguistics, pages 470–470, Morristown, NJ, USA. Association for Computational Linguistics.

P. Lichtenstein, U. De faire, B. Floderus, M. Svartengren, P. Svedberg, and N. L. Pedersen. 2002. The Swedish twin registry: a unique resource for clinical, epidemiological and genetic studies. *Journal of Internal Medicine*, 252:184–205.

P. Lichtenstein, P. F. Sullivan, S. Cnattingius, M. Gatz, S. Johansson, E. Carlstrom, C. Bjork, M. Svartengren, A. Wolk, L. Klareskog, U. de Faire M. Schalling, J. Palmgren, and N. L. Pedersen. 2006. The Swedish twin registry in the third millennium: An update. *Twin Research and Human Genetics*, 9(6):875–882.

C. D. Manning, P. Raghavan, and H. Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

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

A. Strehl and J. Ghosh. 2003. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617.

A. Strehl. 2002. *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*. Ph.D. thesis, The University of Texas at Austin.

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.

Paper VII

Global Evaluation of Random Indexing through Swedish Word Clustering Compared to the People's Dictionary of Synonyms

Global Evaluation of Random Indexing through Swedish Word Clustering Compared to the People's Dictionary of Synonyms

Magnus Rosell
KTH CSC
Stockholm, Sweden
rosell@csc.kth.se

Martin Hassel
DSV, KTH - Stockholm University
Kista, Sweden
xmartin@dsv.su.se

Viggo Kann
KTH CSC
Stockholm, Sweden
viggo@csc.kth.se

Abstract

Evaluation of word space models is usually local in the sense that it only considers words that are deemed very similar by the model. We propose a global evaluation scheme based on clustering of the words. A clustering of high quality in an external evaluation against a semantic resource, such as a dictionary of synonyms, indicates a word space model of high quality.

We use Random Indexing to create several different models and compare them by clustering evaluation against the People's Dictionary of Synonyms, a list of Swedish synonyms that are graded by the public. Most notably we get better results for models based on syntagmatic information (words that appear at the same time in the same context) than for models based on paradigmatic information (words that appear at different times in similar contexts). This is quite contrary to previous results that have been presented for local evaluation.

Also, the results are impressive on their own. Clusterings to ten clusters result in a recall of 83% for a syntagmatic model, compared to 34% for a comparable paradigmatic model, and 10% for a random partition.

1 Introduction

Word space models (see among others (Deerwester et al., 1990; Schütze, 1993; Landauer and Dumais, 1997; Kanerva et al., 2000; Sahlgren, 2006b)) map words to vectors in a multidimensional space by extracting statistics about the context they appear in from a large sample of text. Words that thus become represented by similar vectors (as measured by for instance a similarity measure such as the cosine measure) are considered related.

What this (meaning) relation could be referred to in ordinary (human) semantics is not obvious. It may capture something like synonymy, but may as well regard for instance antonyms, and a hyponym and its hyperonym as highly related.

Word space models have been evaluated using several different schemes (Sahlgren, 2006b). They are all *local* in that they only consider a small part of the words in the model. In Section 4 we discuss these further and introduce a new *global* evaluation scheme that takes all words in the model into consideration, using word clustering and a list of synonyms.

Relations between words based on their contexts can be divided into two categories (Sahlgren, 2006b):

- Two words have a *syntagmatic relation* if they appear together in the same contexts.
- Two words have a *paradigmatic relation* if they appear in similar contexts, but do not co-occur.

Word space models can be constructed in attempts to capture either of these two relations. In this work we use Random Indexing (see Section 2) to construct several different word space models.

The rest of the paper is organized as follows. Section 2 and 3 describe the background, the Random Indexing method for word space models and word clustering. We discuss evaluation of word space models in general and present our proposed global evaluation scheme in Section 4.

In Section 5 we describe and discuss the experiments we have made, the text set we have constructed the Random Indexes from (Section 5.1) and the list of Swedish synonyms graded by

the public, called the People's Dictionary of Synonyms (Section 5.2), that we have used for evaluation. Finally, we give some conclusions in Section 6.

2 Random Indexing

Random Indexing (RI) (Kanerva et al., 2000; Sahlgren, 2005) is an efficient and scalable implementation of the word space model idea. It can be used for attempts at capturing both syntagmatic and paradigmatic relations.

In the paradigmatic version RI assigns a unique extremely sparse *random vector* to each word, usually with a dimension of a few thousands, say n. The random vectors only contain $2t$ ($t \ll n$) non-zero elements, half of which are randomly assigned one (1), and half minus one (-1).

The random vectors are used to construct *context vectors* for all words. The method runs through the texts sequentially word by word focusing on a center word. A portion of the surrounding words are considered being in a *sliding window*. We have used symmetric windows with ω words to the left and right of the center word included. As the sliding window moves through the text the random vectors of all the surrounding words are added to the context vector of the the current center word. The addition may be either constant (all random vectors simply added to the context vector) or weighted depending on the distance between the center word and the particular surrounding word, d. We have used the constant weighting function (which we call *const*) and the commonly used exponential dampening: 2^{1-d} (which we use as a default method).

The resulting word vectors will be similar for words that appear in similar contexts. We measure the similarity/relatedness between two words by the cosine similarity of their corresponding context vectors (the dot product of the normalized vectors)[1].

In the syntagmatic version of RI random vectors are assigned to each text. When a word appears in a text the random vector of the text is added to the context vector of the word[2]. We define the similarity between two words as in the paradigmatic version. It now measures to what extent the words appear in the same texts.

Although, being reasonable approximations of syntagmatic and paradigmatic relations the two RI versions are closely related, as noted in (Sahlgren, 2006b). Consider the constant weighting function for the paradigmatic version. If we increase d until it covers whole texts each word in the text is updated with the sum of all the random vectors in the text (except the one associated with itself, which is a very small part of the sum for large enough texts). This sum serves as a "random vector" (albeit not sparse) for the text, which means that we have a method that is similar to the syntagmatic version[3]. A model, with an ω that does not cover the entire text, will become increasingly similar to the syntagmatic version with increasing ω.

The dense "random vectors" become similar if the texts share a lot of words. In such cases the paradigmatic model is prevented from being fully transformed into a syntagmatic one. However, if the syntagmatic model performs better than a corresponding paradigmatic one, we conjecture that the latter will gain from having its sliding window increased.

3 Word Clustering

We use the K-Means clustering algorithm (see for instance (Manning et al., 2008)) to cluster the words based on the word space model, which provides the necessary representation and similarity measure. K-Means improves on k centroids (component-wise average vectors), that represent k clusters, by iteratively assigning words to the cluster with the most similar centroid. We have set 20 iterations as maximum, as the quality of clustering usually improves most in the beginning of the process.

We use the dot product for similarity between the normalized word vectors and the centroids. Thus the similarity is the average cosine similarity between the word and all words in a cluster. In each iteration all words are compared to all centroids, meaning that when a word is assigned to a cluster all other words are taken into consideration. This is a very appealing property of the

[1] The method corresponds to a projection of the words represented in a space defined by the ordinary word-word-co-occurrence-matrix to a random subspace. When the original data matrix is sparse and the projection is constructed well the distortion in the similarities are small (Kaski, 1998).

[2] This results in a random matrix projection of the common term-by-document matrix used in search engines, see any introductory text, such as (Manning et al., 2008).

[3] For the paradigmatic RI version with a weighting function that decreases with the distance to the center word this relatedness is not as strong, but could perhaps be of some significance.

algorithm in its own right. It also makes it especially suitable for the word space model evaluation scheme we present in the next section.

4 Evaluation of Word Space Models

Word space models have been evaluated using several different resources and evaluation metrics (Sahlgren, 2006b). In (Sahlgren, 2006a) evaluation methods are divided into two categories: *indirect* schemes evaluate the word space model through an application and are therefore not concerned with the model per se, while *direct* schemes compare the models to some lexical resource, trying to judge its ability to model the information it contains.

The existing evaluation schemes are *local* – they only consider a small part of the words in the model. The most common direct evaluation scheme is to let the model perform a synonym test: for each question the model is considered successful if the similarity of the test word to the correct alternative is higher than to the other alternatives. Here, only the words in the synonym test are regarded. How they relate to the other words is not taken into consideration. In fact, it is only the words within the same question that are considered at the same time.

4.1 Global Evaluation

The *global* evaluation scheme we propose takes the relation between all words of the model into account. We cluster all words represented in a model; all words are assigned to one of several clusters by means of the similarity measure. In the assignment of each word all other words are considered via the clusters they appear in. This is true for most clustering algorithms, and in particular for the K-Means algorithm, see Section 3, that we use in the experiments.

> The global evaluation scheme considers a word space model to be of high quality if it leads to clusterings of high quality, and this quality reflects how all the words relate to each other.

When the clustering evaluation is performed using a lexical resource (such as a list of synonyms), we have a global and direct word space model evaluation.

In (Karlgren et al., 2008) it is argued that the most interesting information of the the word space model is found in the local structure, rather than in the global. This should not be confused with our global evaluation. It is the local relations (the similarities between words) that drives the division of all the words into a number of clusters; the clustering takes *all* local relations into consideration. Further, when the evaluation is made against a lexical resource, it concerns the local structure (there are very few synonyms to each word compared to the number of words in the model).

There are many measures of clustering quality that could be used to compare the models. The next section discusses word clustering evaluation and in particular the evaluation measures appropriate for the experiments of Section 5.

4.2 Word Clustering Evaluation

Clustering can be evaluated by internal and external measures. We are interested in how the underlying word space model relation compares to what words humans consider related; i.e. we want to compare the clustering result to an external resource through external evaluation. Depending on the resource this could be achieved in several ways.

In the following experiments (see Section 5) we compare the results to a synonym dictionary that consists of pairs of synonyms (Section 5.2). There are several measures (see for instance (Manning et al., 2008) and (Halkidi et al., 2001)), that compare a clustering to a known categorization based on pairs of words.

Each pair of words can be either in the same or in two different clusters, and in the same category or not. This gives us the four counts presented in the left part of Table 1: tp is for true positives, the number of pairs of words that appear in the same cluster *and* in the same category, fp, fn, and tn are for false positives, false negatives, and true negatives.

	Category		In
Cluster	Same	Different	dictionary
Same	tp	fp	tp
Different	fn	tn	fn

Table 1: Number of Pairs in the Same and Different Clusters Compared to a Categorization and a Dictionary

Using these several measures can be constructed, the most straightforward perhaps being

precision (p) and recall (r):

$$p = \frac{tp}{tp + fp}, \quad (1)$$

$$r = \frac{tp}{tp + fn}. \quad (2)$$

These measures all depend on that we know a full categorization, which is not the case in our experiments; pairs that are not in the synonym dictionary may still be synonymous or have another relation in some other way. We do not know what these relations might be, so we can not use the pairs not in the dictionary without the result becoming incomprehensible.

The only counts we can define using a dictionary of synonym are the ones in the right part of Table 1. Hence, the only measure we can define is recall, r. It denotes the part of the synonym pairs that appear in the same cluster. Although we find it likely, a high recall does not necessarily imply that all (or most of) the words in a cluster are related, only that the synonym pairs are not split between clusters.

To put the evaluation in perspective we will present the results for random partitions as well as the results for the clustering algorithm. The clustering result, of course, has to outperform the random partition in order to be considered any good at all.

In a random partition with k parts (clusters), for each word in a pair the probability for the other word of being in the same clusters is $1/k$. Thus the recall for the entire random partition is $1/k$.

4.3 Local Evaluation through Word Clustering

If we cluster just the words that also appear in the resource we compare the clustering to we make a *local* evaluation, which is much more similar to previously used schemes. It does, however, consider the relations between all the words in the resource. This is usually not the case for other local schemes, as described for the synonym test previously.

5 Experiments

We have clustered the words using several different RI models. To realize them we have used a freely available tool-kit called JavaSDM[4]. For all models we have used eight non-zero elements in the random vectors ($t = 4$). We will use the following notation to abbreviate differences between the representations, see Section 2:

"winω-n", or "text-n".

winω means that we have used a sliding window with ω words before and after the center word, text means that we have used texts as contexts, and n is the dimension of the vectors. In most experiments we have used the exponential dampening weighting function for the winω-n-methods. In those cases we have applied constant weighting we indicate that thus:

"winω-n-const".

As the K-Means algorithm is not deterministic we cluster the words ten times for each representation and calculate averages and standard deviations. We can only compare results for the same number of clusters and for two results to be considered different they, as a rule of thumb, have to not overlap with their standard deviations.

5.1 Text Set

The RI:s have been trained on a text set consisting of all texts from the Swedish Parole corpus (Gellerstam et al., 2000), 20 million words, the Stockholm-Umeå Corpus (Ejerhed et al., 1992), 1 million words, and the KTH News Corpus (Hassel, 2001), 18 million words. In all they contain 114 691 files/texts.

We tokenized and lemmatized all texts using GTA, the Granska Text Analyzer (Knutsson et al., 2003), removed stop words (function words and extremely frequent words) and all words that appeared less than four times.

5.2 The People's Dictionary of Synonyms

For the evaluation we have used the People's Dictionary of Synonyms (Kann and Rosell, 2005), a dictionary produced by the public. In 2005 a list of pairs of possible synonyms was created by translating all Swedish words in a Swedish-English dictionary to English and then back again to Swedish using an English-Swedish dictionary. The generated pairs contained lots of non-synonyms. The most inaccurate pairs were automatically removed using Random Indexing.

Every user of the popular dictionary Lexin online was given a randomly chosen pair from the list, and asked to judge it, for example:

[4] http://www.nada.kth.se/∼xmartin/java/JavaSDM/

Are 'spread' and 'lengthen' synonyms? Answer using a scale from 0 to 5 where 0 means *I don't agree* and 5 means *I do fully agree*, or answer *I do not know*.

Users of the dictionary could also propose pairs of synonyms, which subsequently were presented to other users for judgment.

All responses were analyzed and screened for spam. The good pairs were compiled into the People's Dictionary of Synonyms. Millions of small contributions have resulted in a constantly growing dictionary of more than 75 000 Swedish pairs of synonyms. Since the dictionary is constructed in a giant cooperative project on the web, it is a free downloadable language resource[5].

A unique feature of the People's Dictionary of Synonyms is that the synonymity of each pair is graded. It is the mean grading by the users who have judged the synonymity that defines this grade.

The available list contains 36 106 pairs that have a grading of 3.0 up to 5.0 in increments of 0.1. For the following evaluation we have only retained the 18 053 unique ones. Through the rest of the paper we refer to this part of the dictionary as *Synlex*. (See Table 5 and Figures 1:a and b for more information.)

5.3 Results

We report some clustering results in Tables 2 through 4. Where the standard deviation is 0.00 for the random partitions[6] we have not reported it. The best representation for each number of clusters is presented in bold face letters (for ties: both).

The results in Table 2 follow the global evaluation scheme of Section 4.1, while Table 4 uses the local scheme presented in Section 4.3. Table 3 reports the results for global evaluation for parts of Synlex with higher grade of synonymity.

We report some statistics for Synlex and the RI:s with representations 1800-win4 and 1800-text in Table 5 and Figure 1. The pairs in Synlex that are not in the RI are mostly multi-word tokens, words in non lemma form, and slang words that the public has wanted to include.

Figure 1 contains several different plots of the distribution of pairs over certain RI similarities and Synlex grades for both Synlex, the RI:s with representations 1800-text and 1800-win4, as well as combinations. In Figure 1:c and d we have calculated the RI similarities for a sample of 5000 words (not including the similarity of a word to itself).

[5] See lexin.nada.kth.se/synlex

[6] This is the case for large enough sets of words.

k	Representation dim-context(-const)	Recall (stdv) K-Means	Random
100	**1800-text**	0.56 (0.10)	0.01
100	1800-win4	0.15 (0.01)	0.01
5	500-text	0.48 (0.12)	0.20
5	**1000-text**	0.77 (0.07)	0.20
5	**1800-text**	0.83 (0.01)	0.20
10	**500-text**	0.77 (0.05)	0.10
10	**1000-text**	0.80 (0.05)	0.10
10	**1800-text**	0.83 (0.02)	0.10
5	500-win4	0.41 (0.02)	0.20
5	1000-win4	0.42 (0.02)	0.20
5	1800-win4	0.44 (0.03)	0.20
10	500-win4	0.32 (0.01)	0.10
10	1000-win4	0.31 (0.01)	0.10
10	1800-win4	0.34 (0.02)	0.10
5	500-win30	0.44 (0.03)	0.20
5	1000-win30	0.43 (0.03)	0.20
5	1800-win30	0.45 (0.03)	0.20
10	500-win30	0.34 (0.03)	0.10
10	1000-win30	0.34 (0.01)	0.10
10	1800-win30	0.33 (0.01)	0.10
5	500-win250	0.42 (0.02)	0.20
5	1000-win250	0.41 (0.03)	0.20
5	1800-win250	0.44 (0.02)	0.20
10	500-win250	0.33 (0.01)	0.10
10	1000-win250	0.34 (0.02)	0.10
10	1800-win250	0.33 (0.01)	0.10
5	500-win30-const	0.45 (0.03)	0.20
5	1000-win30-const	0.43 (0.02)	0.20
5	1800-win30-const	0.44 (0.03)	0.20
10	500-win30-const	0.34 (0.02)	0.10
10	1000-win30-const	0.34 (0.02)	0.10
10	1800-win30-const	0.34 (0.01)	0.10
5	**500-win250-const**	0.72 (0.07)	0.20
5	1000-win250-const	0.66 (0.04)	0.20
5	1800-win250-const	0.76 (0.09)	0.20
10	500-win250-const	0.58 (0.03)	0.10
10	1000-win250-const	0.56 (0.03)	0.10
10	1800-win250-const	0.60 (0.01)	0.10
5	**500-win1000-const**	0.67 (0.04)	0.20
5	1000-win1000-const	0.68 (0.05)	0.20
5	1800-win1000-const	0.69 (0.06)	0.20
10	500-win1000-const	0.58 (0.02)	0.10
10	1000-win1000-const	0.60 (0.03)	0.10
10	1800-win1000-const	0.60 (0.03)	0.10

Table 2: *Global Evaluation*. The Effect of Different Contexts. Recall for Word Clustering of All Words, RI in Table 5. (k – the number of clusters)

5.4 Discussion

Our major finding is that the syntagmatic RI versions perform much better than the paradigmatic versions in our global evaluation. This is apparent in Table 2, which contains the results for the syn-

		Representation	Recall (stdv)	
Grade(s)	k	dim-context	K-Means	Random
[4.0, 5.0]	**5**	**1800-text**	0.81 (0.01)	0.20 (0.01)
[4.0, 5.0]	5	1800-win4	0.45 (0.03)	0.20 (0.00)
[4.0, 5.0]	**10**	**1800-text**	0.81 (0.02)	0.10 (0.00)
[4.0, 5.0]	10	1800-win4	0.33 (0.02)	0.10 (0.00)
5.0	**5**	**1800-text**	0.78 (0.02)	0.20 (0.03)
5.0	5	1800-win4	0.43 (0.04)	0.20 (0.02)
5.0	**10**	**1800-text**	0.79 (0.03)	0.11 (0.01)
5.0	10	1800-win4	0.33 (0.02)	0.09 (0.02)

Table 3: *Global Evaluation for Synlex Grade Intervals*: Recall for Word Clustering of All Words, RI in Table 5. (k – the number of clusters)

	Representation	Recall (stdv)	
k	dim-context	K-Means	Random
5	500-text	0.22 (0.01)	0.20
5	1000-text	0.30 (0.03)	0.20
5	**1800-text**	0.45 (0.06)	0.20
10	500-text	0.11 (0.00)	0.10
10	1000-text	0.19 (0.04)	0.10
10	**1800-text**	0.31 (0.08)	0.10
5	**500-win4**	0.39 (0.00)	0.20
5	**1000-win4**	0.40 (0.01)	0.20
5	**1800-win4**	0.40 (0.00)	0.20
10	**500-win4**	0.27 (0.01)	0.10
10	**1000-win4**	0.27 (0.02)	0.10
10	**1800-win4**	0.28 (0.01)	0.10

Table 4: *Local Evaluation.* Recall for Word Clustering of Words Only in Synlex, Synlex*RI in Table 5. (k – the number of clusters)

tagmatic versions ("n-text") and several paradigmatic versions with different window sizes and the two weighting schemes. This result differ to local direct evaluations that have been performed against synonym resources, where paradigmatic versions have been more successful (Sahlgren, 2006b).

It is only for really large windows and the constant weighting scheme ("-const"), a paradigmatic version can compete. This is in line with the argument in Section 2 that a paradigmatic version with large windows and constant weighting scheme is closely related to the syntagmatic version.

The paradigmatic version with constant weighting scheme improves with increasing window size ($2 \cdot \omega$), but seems to be saturated at $\omega = 250$, since results do not improve for $\omega = 1000$. A window size of 500 covers most texts in their entire.

That the paradigmatic versions with exponential weighting (not "-const") does not improve with increasing window size is not surprising; the impact of words far away from the center word is limited.

The syntagmatic version perform better with increasing dimensionality (n). This suggests that they might benefit more from even larger dimensionality. The paradigmatic versions are uneffected by the dimensionality.

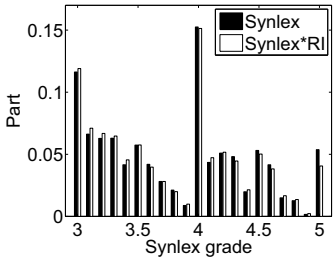
The graded evaluation in Table 3 does not differ from the *not* graded evaluation in Table 2 and thus confirms the results. This also suggests that the models can not separate the different gradings in Synlex.

The results for the local evaluation (see Section 4.3) in Table 4 gives a different view. The syntagmatic model performs much worse than in the global evaluation, while the paradigmatic performs similarily. Here, the paradigmatic model outperforms the syntagmatic, for low dimensionalities. In fact, the syntagmatic model performs as a random partition for $n = 500$. However, as in the global evaluation the syntagmatic version performs better with increasing dimensionality. For $n = 1800$ it performs comparable to the syntagmatic version.

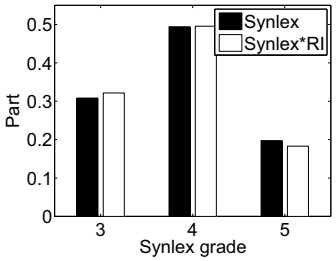
The syntagmatic model exploits the information in all of the words it contains and performs much better when it is allowed to use them (global vs. local evaluation). Then it outperforms the paradigmatic models. The results for the paradigmatic models are unaffected by wether they are allowed to consider all other words. Both versions obviously have their merits.

We observe that the best performing of the evaluated models is 1800-text, the syntagmatic model with a dimension of $n = 1800$. It performs superior to all paradigmatic models in the global evaluation and comparable in the local evaluation. In the global evaluation, for ten clusters, it achieves 83% recall, compared to 34% for the paradigmatic models with exponential dampening, and 10% for the random partitions.

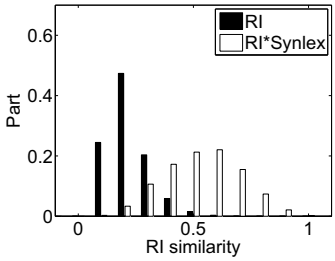
Figure 1 and Table 5 give us some more information about two of the RI models: the syntagmatic 1800-text, and the paradigmatic 1800-



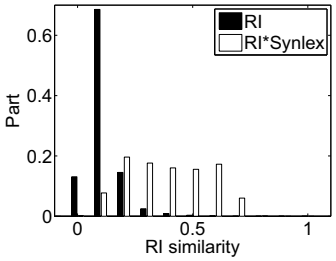
a) Synlex grades, $[3.0, 5.0]$



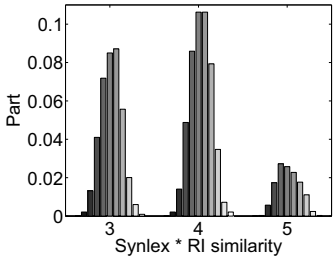
b) Synlex grades, rounded to integers.



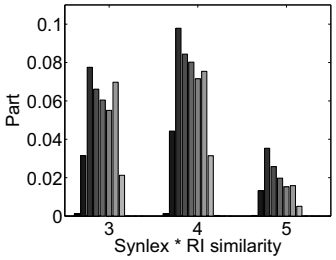
c) RI-similarity, 1800-win4



d) RI-similarity, 1800-text



e) Synlex Grades, RI-similarity, 1800-win4



f) Synlex Grades, RI-similarity, 1800-text

Figure 1: Distributions: the part of all pairs for certain similarities/grades. *Synlex*RI* and *RI*Synlex* means the pairs that appear in both Synlex and the RI. We use the intervals $(-0.1, 0.0]$, $(0.0, 0.1]$, $\ldots (0.9, 1.0]$ for the RI similarity. In Figures c, and d the part of the pairs in the similarity intervals are presented. For Figures e and f it is the part of pairs per Synlex grade. The gray-scale in Figures e and f corresponds to the RI similarity intervals. See Table 5 for total counts.

win4[7]. Both models assign higher similarity to pairs in Synlex than to those not included, see Figure 1:c and d. Model 1800-text is perhaps slightly more distinct in its separation, which might in part explain its better clustering result.

None of the models is able to separate the different Synlex gradings, see Figures 1:c and d. This is also supported by the evaluation with respect to Synlex grade intervals in Table 3.

6 Conclusions

We have presented and used a new *global* evaluation scheme for word space models. While local evaluation only considers a portion of the words in the model, global evaluation takes them all into consideration.

[7] In (Sahlgren, 2006b) it is recognized that a denser model leads to higher average similarities. This is the case for 1800-win4 compared to 1800-text (there are more words than texts).

	Pairs	Words	Possible Pairs
Synlex	18 053	15 296	$1.67 \cdot 10^8$
RI	$9.43 \cdot 10^9$	137 364	$9.43 \cdot 10^9$
Synlex*RI	14 051	11 173	$6.24 \cdot 10^7$

Table 5: Pairs and Words in Synlex and RI. *Synlex*RI* means pairs that appear in both Synlex and the RI.

We constructed word space models (realized using Random Indexing) on Swedish texts and used a list of synonyms called the The People's Dictionary of Synonyms for evaluation. In our global evaluation scheme models that attempt to capture syntagmatic relations between words performed better than models that attempt to capture paradigmatic relations. This result is contrary to previous results using local evaluation against synonym resources.

Another result worth mentioning is that the RI models are not able to separate the gradings of the synonym dictionary. They do, however, give higher similarity to synonyms in the dictionary than to other word pairs.

This work adresses the theoretic matter of how to evaluate word space models. Though we hope that the use of a combination of both local and global evaluation will promote the investigation of the nature of word space models and the word (meaning/similarity) relation they define, we conclude the paper with a more tangible question.

The syntagmatic models perform very well when they are allowed to take all words into consideration. How can this be exploited in applications?

References

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.

E. Ejerhed, G. Källgren, O. Wennstedt, and M. Åström. 1992. *SUC - The Stockholm-Umeå Corpus*, version 1.0 (suc 1.0). CD-ROM produced by the Dept of Linguistics, University of Stockholm and the Dept of Linguistics, University of Umeå. ISBN 91-7191-348-3.

M. Gellerstam, Y. Cederholm, and T. Rasmark. 2000. The bank of Swedish. In *Proc. of Second Int. Conf. on Language Resources and Evaluation. LREC-2000*, pages 329–333, Athens, Greece.

M. Halkidi, Y. Batistakis, and M. Vazirgiannis. 2001. On clustering validation techniques. *J. of Intelligent Information Systems*, 17(2-3):107–145.

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

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*.

V. Kann and M. Rosell. 2005. Free construction of a free swedish dictionary of synonyms. In *Proc. 15th Nordic Conf. on Comp. Ling. – NODALIDA '05*.

J. Karlgren, A. Holst, and M. Sahlgren. 2008. Filaments of meaning in word space. *Advances in Information Retrieval*, pages 531–538.

S. Kaski. 1998. Dimensionality reduction by random mapping: Fast similarity computation for clustering. In *Proceedings of IJCNN'98, International Joint Conference on Neural Networks*, volume 1, pages 413–418. IEEE Service Center, Piscataway, NJ.

O. Knutsson, J. Bigert, and V. Kann. 2003. A robust shallow parser for Swedish. In *Proc. 14th Nordic Conf. on Comp. Ling. – NODALIDA '03*.

T. K. Landauer and S. T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240.

C. D. Manning, P. Raghavan, and H. Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

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. Sahlgren. 2006a. Towards pertinent evaluation methodologies for word-space models. In *In Proceedings of the 5th International Conference on Language Resources and Evaluation*, Genoa, Italy.

M. Sahlgren. 2006b. *The Word-Space Model: using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Ph.D. thesis.

H. Schütze. 1993. Word space. In S. Hanson, J. Cowan, and C. Giles, editors, *Advances in Neural Information Processing Systems 5*. Morgan Kaufmann Publishers.

Paper VIII

Infomat –
Visualizing and Exploring Vector Space Model Data Matrixes

Infomat
Visualizing and Exploring Vector Space Model Data Matrixes

Magnus Rosell

Abstract `Infomat` is a vector space visualization tool aimed at Information Retrieval. It presents information stored in a matrix, such as the term-document-matrix, as a rectangular picture. The opacity of each pixel is proportional to the weight(s) of the corresponding matrix element(s).

Reordering the objects of the rows and columns makes different distributional patterns appear. These can be explored to understand the relations (similarities and differences) between the objects. `Infomat` allows the user to zoom in and out of the picture to obtain more detailed information, to remove objects and matrix elements, to reweight the matrix, and to cluster all, or a part of the objects. At the same time textual information is presented.

`Infomat` provides an overview of the content of the entire data and parts of it. In particular, text clustering results become easier to grasp, than when presented only in textual form.

Keywords Information Retrieval · Visualization · Vector Space Model · Exploration · Clustering · Text Clustering

1 Introduction

In Information Retrieval (IR) (see for instance Manning et al. [2008]) texts and/or words are often represented in a vector space model, for instance the word-text-matrix used by search engines and co-occurrence matrixes of the words in a text set. Each matrix element is assigned a weight that values the relation between the objects of the row and column. Such matrixes are often huge (and sparse) and therefore virtually impossible for humans to grasp.

M. Rosell
KTH CSC
100 44 Stockholm
Sweden
Tel.: +46-8-7906540
Fax.: +46-8-7900930
E-mail: rosell@csc.kth.se

Many existing visualization methods calculate the similarity between all objects and project this relationship down to two or three dimensions. Such methods have not proven to be superior to textual result presentation. However, there is a firm belief that there ought to be something to be gained from visualization, since pictures can communicate some kinds of information very fast. [Hearst, 1999]

We introduce `Infomat`[1], a tool that presents the representation matrix directly in two dimensions as a scatter plot and lets the user explore the similarity of the objects by reordering the rows and columns. If the objects are tangible entities such as for instance texts and words this leads to pictures that can be comprehended. The picture convey information from a bird's eye perspective that is otherwise hard to obtain. When earlier methods only inform the user *that* objects are similar (displayed near each other in the projection), `Infomat` reveals *why*.

The rest of the paper is organized as follows. The next section relates previous work in visualization and in text clustering. The later because Section 3 introduces `Infomat` via a text clustering example. Sections 4 through 9 goes through different aspects of the program in the context of the text clustering example. We give a short summary of a small user survey we have conducted in Section 10, and in Section 11 we describe some other possible applications. In Section 12 we discuss a few details in the implementation of the program. Finally, we draw some conclusions in Section 13.

2 Previous Work

`Infomat` can visualize any data matrix. Clustering is an integral part of the concept. Our main interest lies in Information Retrieval and text clustering. The following subsections introduce this particular domain and other related visualization methods.

2.1 Information Retrieval

Information Retrieval (IR) [Manning et al., 2008; Baeza-Yates and Ribeiro-Neto, 1999; Van Rijsbergen, 1979] is concerned with how to retrieve or extract information, usually in the form of texts, from a large data base of texts. The most well-known example of an IR application is the search engines that retrieve web documents that are thought to be relevant to a query.

Many IR applications have as their core data structure a sparse matrix that relates some concrete objects, such as texts and words. Search engines utilize the term-document-matrix, with values in the matrix elements that can be thought of as modelling how important the particular word (term) is for the text (document). The value, or weight $w_{i,j}$, in an element is usually calculated by multiplying a local weight like the term frequency $\mathrm{tf}_{i,j}$, that models how important a word is within a text, and a global weight like the inverse document frequency idf_j, that models how specific it is within the entire text set, for example:

$$w_{i,j} = \mathrm{tf}_{i,j} \cdot \mathrm{idf}_i \tag{1}$$

$$\mathrm{tf}_{i,j} = \frac{n_{i,j}}{\sum_i n_{i,j}}, \tag{2}$$

$$\mathrm{idf}_i = \log \frac{n}{n_{\mathrm{word}(i)}}, \tag{3}$$

[1] http://www.csc.kth.se/tcs/projects/infomat/infomat/

where $n_{i,j}$ is the number of times word i appears in document j, $n_{\text{word}(i)}$ is the number of documents that word i appears in, and n is the total number of texts.

The reason for building this representation matrix is to be able to define the similarity in content between texts (and queries). The similarity measure is the basic operator in several methods. There are many variants and the most common is the cosine measure, the cosine of the angle between two text vectors t_u and t_v:

$$\text{sim}(t_u, t_v) = \frac{t_u \circ t_v}{\|t_u\| \cdot \|t_j\|} = \frac{1}{\|t_u\| \cdot \|t_v\|} \sum_i w_{i,u} w_{i,v}. \tag{4}$$

If the text vectors are normalized beforehand we can use the dot product. As the texts are represented by vectors this model is known as the *vector space model*, and a lot of geometrical analogies are used.

2.2 Text Clustering

Clustering algorithms partition a set of objects into a number of clusters (parts) if provided a similarity (or dissimilarity) measure between them. Some also need a representation of the objects, usually provided in a object-by-feature matrix (as the term-document matrix). There are many clustering algorithms, see for instance [Jain et al., 1999] for a review. The most well-known is probably the fast and simple K-Means algorithm. In K-Means a cluster is represented by the centroid, the average vector of the representation vectors for the objects in the cluster.

In text clustering (see for instance [Manning et al., 2008; Frakes and Baeza-Yates, 1992; Van Rijsbergen, 1979]) the goal is to divide a set of texts into groups of similar texts. If we use the term-document-matrix and the cosine measure, described above, we expect to obtain groups of texts that are similar in content.

Text clustering by content has been proposed as a means to generate dynamic tables of content over text sets in order to explore them by Cutting et al. [1992]. Their clustering system Scatter/Gather presents clusterings of a few clusters in textual form. Each cluster is presented as a *cluster digest*. It consists of *typical titles* and *topical words*. The typical titles are the texts most similar to the cluster centroid, and the topical words are those with highest weight in the centroid. The topical words is an example of what is often called a *cluster label*, a short description of the cluster content. We prefer to use *cluster description* instead, as *label* may be confused with the cluster *name*, which could be any arbitrary string (like "Cluster 1" for instance)[2]. There are several other methods that could be used to extract a cluster description [Manning et al., 2008].

The Scatter/Gather system allows the user to choose any number of the presented clusters and re-cluster them in order to obtain a more detailed view of a subset of all texts. This interaction, we believe, is crucial for the user to understand the content. The large clusters that are produced to provide an overview of the set are not easily described by just a few words, regardless of how these were extracted.

[2] Also *labeling* could mean to assign texts to clusters, i.e. give each text the label of belonging to a certain cluster.

2.3 Visualization

Two-dimensional data can be conveniently visualized as a scatter plot. The task of visualizing multi-dimensional data is hard and there is a lot of work done on trying to do this. One example is the *grand tour* [Asimov, 1985], that presents a series of two-dimensional projections of the full data as scatter plots. The number of projections required to cover the data increases with the number of dimensions. Such a tool puts a lot of faith in the viewer's ability to combine them into an understanding of the full data. Dhillon et al. [2002] introduce *class-preserving* projections that retain as much as possible of the differences between classes of objects in the data. They call a sequence of such projections a *class tour*.

When the meanings of the dimensions are comprehensible the data matrix contains values (measurements) that represent the objects with respect to the dimensions/features. Such a matrix can be presented as a table with the names of the objects and features along rows and columns. The *reorderable matrix* [Bertin, 1981] lets the user reorder the objects and features so that similarities and differences between them can be seen. It requires that all the measurements (displayed as numbers or otherwise), objects and features can be seen (read). The *table lens* [Rao and Card, 1994] is a graphical interface that allows the user to browse a huge table with a lens that magnifies the rows and columns it is placed over.

The Hierarchical Clustering Explorer[3] (HCE) [Seo and Shneiderman, 2002] is a very impressive visualization tool, with many features, that is centered around hierarchical clusterings of both objects and features. The main view contains the dendogram for the objects, which at the leaves is aligned with a *color mosaic*. The color mosaic is a (compressed) representation of the object-feature-matrix in the form of a rectangular picture, where the colors of the pixels indicate the value in the corresponding matrix element(s). It is a direct presentation in two dimensions of a data matrix, just like the reorderable matrix. However, the matrix is compressed and the order of the objects and features are fixed and decided by the hierarchical clusterings.

In addition to the main view HCE features several other views of the data, such as scatter plots that are linked to the main view; selected objects in both views are highlighted so that the user can compare them. HCE has been applied to and evaluated in different domains (especially genomic micro-array data) with good results [Seo and Shneiderman, 2006]. The data matrixes are dense; the number of objects is up to 40 000 and the number of features between 10 and 150.

There is a lot of work done on visualization in information retrieval. Many methods for text clustering visualization project the similarity matrix (the similarity between all objects) to two (sometimes three) dimensions in some fashion [Hearst, 1999]. A projection to two dimensions is inherent in the clustering method *Self Organizing Maps* [Haykin, 1999], which has been applied to text collections [Lagus et al., 2004].

Unfortunately, so far, visualization systems for text clustering are harder to use for non-expert users than text based systems like Scatter/Gather. This is because the content of texts is best understood by reading them [Hearst, 1999]. However, users will become more accustomed to visualization and a combined approach might be proven more useful ...

[3] http://www.cs.umd.edu/hcil/hce/

3 A Text Clustering Example

`Infomat` handles any data matrix, with any objects or features along rows and columns. We will demonstrate it with a text-word-matrix in the context of text clustering. For this purpose we use a small part of the *20 Newsgroups* corpus[4], a collection of newsgroup texts in English, originally collected by Ken Lang [Lang, 1995]. We extracted texts from five of the newsgroups, see Table 1, removed common stop words and words that appear in few texts, and applied an implementation of the Porter stemmer [Porter, 1980] to normalize words. This resulted in 2590 texts with 9168 stems. Each text used on average 60 of these stems, and every stem appeared in about 17 texts. The file names are numbers and in the following we present these as an abbreviation for the entire texts[5].

We built a text-word-matrix for this text set, i.e. the transpose of the common word-text-matrix. Each matrix element corresponding to an appearance of a word in a text was assigned a weight by a tf*idf-scheme and the rows were normalized, see Section 2. We discuss weighting further in Section 6.1.

`Infomat` presents a matrix as a scatter plot in two dimensions. All elements in the matrix with a weight (usually in $[0, 1]$) greater than zero are plotted as points with opaqueness proportional to the weight. This picture would for most IR matrixes (and our example) be larger than any monitor. Hence we have to compress it. We do this by averaging – letting one pixel get the average weight of several neighboring matrix elements. See Section 4 for a longer discussion of how the picture is constructed.

Figure 1:a shows the compressed matrix for the text set. The rows represent the texts and the columns the words that appear in them. As the picture is compressed each horizontal pixel line represents several texts, and each vertical line several words.

This picture is not very informative! However, we know that there is information in IR matrixes. The positive outcome of several methods show that. To expose this information we consider the relation between the objects along both rows and columns. By sorting the row and/or column objects in different ways visual patterns that indicate relationships among them may occur. When the rows and columns represent actual objects the visual patterns are possible to comprehend.

If there is a natural order and/or grouping of any of the objects it is especially interesting. Figure 1:b gives such an example. The texts in our example have been clustered to five clusters (separated by horizontal black lines) using the K-Means algorithm. The clustering imposes an order on the texts; texts belong to clusters. By presenting the texts of clusters as adjacent rows good clusters will appear similar in the distribution of opaqueness over the words (columns). Dissimilar clusters will appear with contrasting distributions. How legible the similarities and differences in distributions are depends on several factors; the data and the screen resolution among others. The newsgroup postings usually contain rather many words, as the writers include the post(s) they respond to. It makes this picture rather dense. In Figure 2 we present similar pictures for a set of short Swedish newpaper articles.

Many clustering algorithms result in an ordering of the texts within the clusters. Such orderings may expose more structure. Here, for the K-Means algorithm, the order of the texts within each cluster (the ranking) is based on their similarity to the cluster

[4] Downloaded from `http://people.csail.mit.edu/jrennie/20Newsgroups/`

[5] Some texts have informative titles that could be used instead of the filename. The subject headings of the posts (see Figure 6) could have been used for this, but we have not.

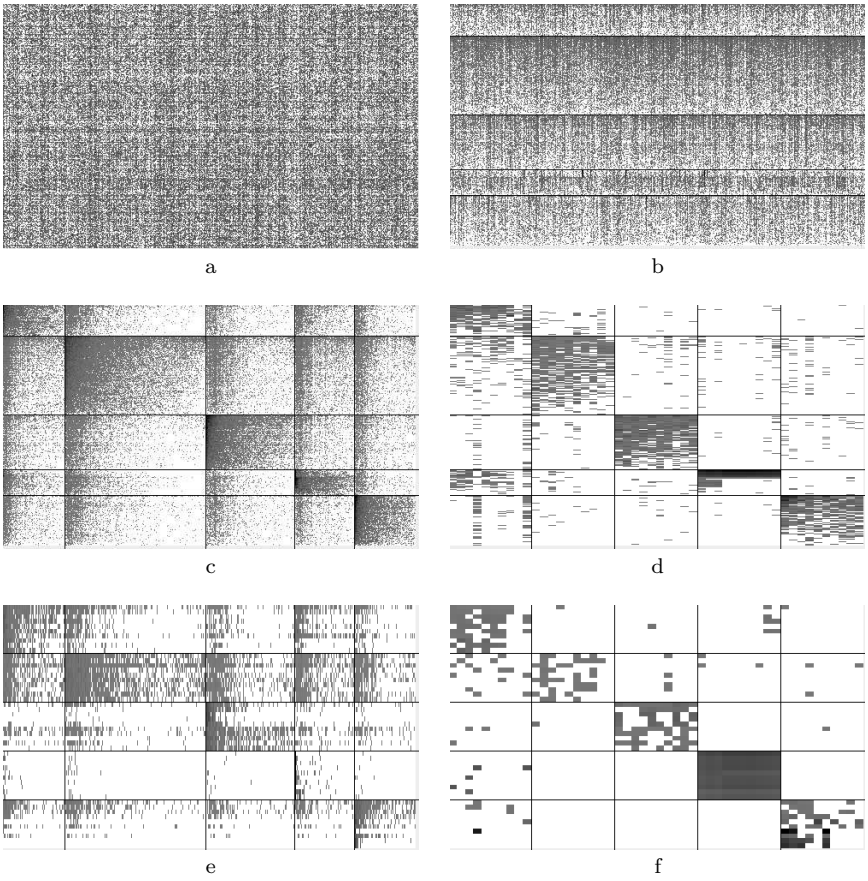


Fig. 1 Compressed scatter plots of the weights of a set of 2590 newsgroup posts/texts (rows), represented using 9168 different stems/words (columns). Solid black lines indicate cluster borders. See Section 3 for a longer description of Figures a through c. Figures d through f are described further in Section 8. *a)* Random order of texts and words. *b)* A K-Means clustering to five text clusters. Texts are ordered within clusters according to similarity to the centroid. The same random order of the words as in a. *c)* The text clustering plus a word clustering according to the algorithm in Figure 3. Words ordered according to weight within clusters. *d)* As in c, but only the top ten words for each word cluster. *e)* As in c, but only the top ten texts for each text cluster. *f)* The combination of d and e: only the top ten objects for each text and word cluster.

centroid (the component-wise average vector). We discuss orderings of different kinds further in Section 6.2.

There is a dualism between text clusters and word clusters, as noted by for instance Dhillon [2001]. In Figure 1:c we have constructed a word clustering based on the text clustering using the algorithm given in Figure 3.

The difference in word distribution between the clusters are quite obvious in Figure 1:c and can be discerned in Figure 1:b. This gives a superficial understanding of how separate and coherent the clusters are. Deeper inspection of these distributional patterns (as will be described) may give an understanding of how and why the clusters

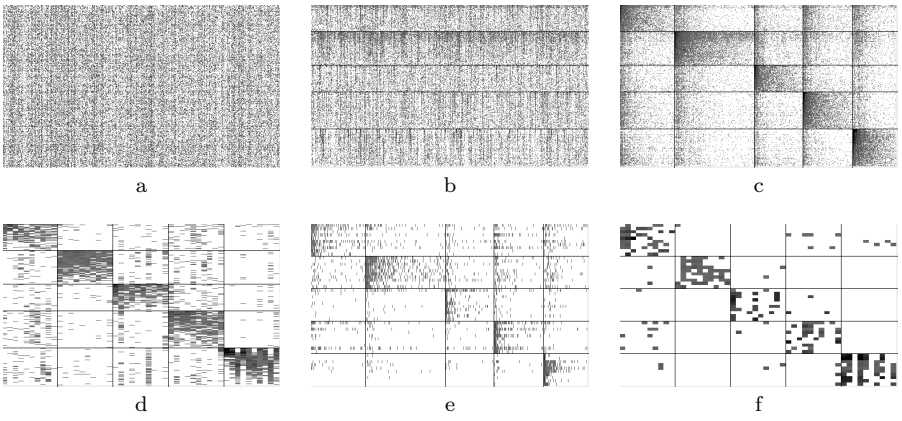


Fig. 2 Compressed scatter plots of the weights of a set of 2500 Swedish newpaper articles (rows), represented using 5520 different lemmas/words (columns). The articles contained on average 30 different words and each word appeared in on average 13 texts after preprocessing. The figures are constructed in the same way as in Figure 1.

Input: a text set $\mathfrak{T}$,
a set $\mathfrak{W}$ of all words appearing in $\mathfrak{T}$,
a clustering of the texts $\{T_i\}$.

- For each text cluster T_i:
  - calculate the centroid $\overline{T_i}$
  - construct an empty corresponding word cluster W_i
- For each word $w \in \mathfrak{W}$:
  - find $\overline{T_k}$ with maximal weight for w
  - put w in W_k, ordered by its weight in $\overline{T_k}$

Output: a clustering of the words $\{W_i\}$.

Fig. 3 *The Relative Clustering Algorithm* returns a word clustering corresponding to a text clustering.

were generated. Cluster content may be grasped by considering words that appear in all text clusters, in just some of them, or only in one. The word clusters of Figure 1:c can be considered extensive cluster descriptions for the text clusters. Cluster descriptions are discussed further in Section 8.

`Infomat` gives a direct visualization of the representation. For the idea to be applicable the objects probably need to be comprehensible entities, such as words, texts, etc. When compressing the matrix we loose the connection to single row and column objects. General trends are exposed at the cost of loosing the local details.

Through visual interaction the user is able to explore the results, zooming in and out to inspect interesting parts. Figure 4 shows the main window (to the right) zoomed in on a part of the picture from Figure 1:c. Which part is indicated in the overview window (above to the left). The pixel window (below to the left) shows the matrix elements for the pixel the mouse pointer is positioned at. In Section 5 we describe inspection of the picture further.

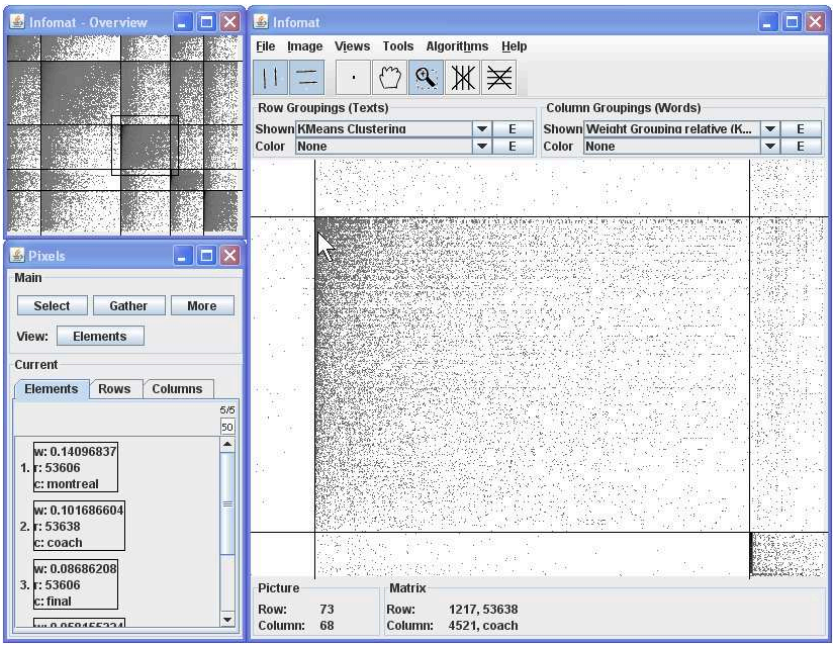


Fig. 4 The `Infomat` interface. The overview window (above left) shows the same picture as in Figure 1:c. A rectangle indicates the part which is displayed in the main window (to the right). The matrix elements represented by the pixel the mouse pointer is pointing to are listed in the pixel window (below left). The non zero elements are presented with their weight (w), and the row (r) and column (c) objects. In this example texts have numbers as names. The bottom part of the main window shows which pixel the mouse pointer is at and which is the first text and word that it represent in the current ordering (both the order number and the string). Pressing the "E-button" right of "KMeans Clustering" opens up the left window shown in Figure 6.

When processing an unknown data set, such as a text set, there are innumerable choices to be made and parameters to set. Further, what is a good clustering of a text set is conditional on the purpose for constructing it. Several partitions (or groupings) may be interesting and useful, providing different views on the same data. `Infomat` provides the user with tools to cluster and re-cluster parts of or the entire data. It is also possible to import and export any kind of *grouping*, by which we mean a set of one or more groups of objects.

4 Picture Construction

When the matrix is larger than the available picture size we have to present a compressed version. This could be achieved in several ways. It would be possible to do some kind of smoothing over neighboring matrix elements. We prefer to keep the link to individual objects and let each pixel represent the matrix elements of a corresponding (small) submatrix. Figure 5 summarizes how the opacities of the pixels are calculated from the weight matrix.

Input:

- Size of picture to be constructed:
  X pixel columns and Y pixel rows.
- Sparse weight matrix for a set of I row objects and a set of J column objects:
  $W = \{w_{i,j} > 0\}_{i \in I, j \in J}$.
- Subsets of the objects:
  $R \subset I$, $C \subset J$.
- Orderings of the subsets:
  $r(i) \in [0, |R| - 1]$ for all $i \in R$,
  $c(j) \in [0, |C| - 1]$ for all $j \in C$.

Construct sparse picture matrix with pixel opacity values: $P = \{p_{x,y} > 0\}_{x \in [0, X-1], y \in [0, Y-1]}$.

1. Let: $is = |R|/Y$, $js = |C|/X$
2. For all $w_{i,j} > 0$ and $i \in R$, $j \in C$:
   - Let $x = \lfloor c(j)/js \rfloor$, $y = \lfloor r(i)/is \rfloor$
   - $p_{x,y} \leftarrow p_{x,y} + w_{i,j}$
3. Normalize to $[0, 1]$: $p_{x,y} = p_{x,y} / \max(p_{x,y})$

Output: P

Fig. 5 *Picture Construction Algorithm* for weight matrixes with more elements than there are pixels (more matrix rows than picture pixel rows *and* more matrix columns than picture pixel columns). We use the *double hashmap* described in Section 12.1 to store both sparse matrixes. The picture is painted using the information in P.

Whether averaging or smoothing the compression is somewhat questionable – neighboring matrix elements (and rows and columns) are not necessarily related in a way that motivates them being presented together. This leads to a distorted image that can hide local differences. The zoom function, and the possibility to see what objects are represented by each matrix element can expose the local information. When there is an ordering and/or grouping of the rows and columns the compression may on the other hand reveal general trends.

In the compression we let the opacity of the pixels be proportional to the average weight of their submatrixes; higher opacity meaning higher average weight. The smaller picture we construct, the denser it will be. Compare the zoomed in part of Figure 4 to the same area in the overview window, which is presented using many more pixels. The full size scatter plot, on the other hand, would normally be very uninteresting, as most of it would have no opacity at all.

The opacity of a pixel has to lie in $[0, 1]$. To calculate the opacity for each pixel we use a simple linear scaling based on the weights of all the pixels (step 3 in Figure 5). It could, however, be achieved in several ways. A fully automatic solution would be to use a histogram equalization (see for instance Gonzalez and Woods [1992]).

In `Infomat` the user may set the minimum and maximum opacity for pixels that have any matrix elements with weights (see Figure 6). Alterations of the opacity of the pixels do not affect the weights of the matrix elements.

5 Picture Investigation/Exploration

The full compressed picture constitute a bird's-eye view of the matrix. It is always presented in the *overview window* of `Infomat`. Any part of it can be displayed in the *main window*, indicated in the overview window by a (small) rectangle. See Figure 4 for an example. The user can select what to show in the main window by selecting a rectangle in the main or overview window. This allows the user to zoom in and out of the picture to inspect interesting parts of the matrix at an appropriate level. It is possible to zoom in so that single objects along rows and/or columns are represented by several pixels.

In parallel with the visualization `Infomat` also provides full textual display of all groupings, groups, objects, and matrix elements. As the user moves the mouse pointer over the picture a list of the non zero matrix elements for the current pixel is displayed in the *pixel window*, see Figure 4. It can also present which matrix row and column objects the current pixel row or column represent. Using the pixel window the user can also collect objects and matrix elements from different parts of the matrix. These can be used for further processing, such as removing them. See Section 6 for more on how to manipulate the matrix.

It is also possible to view the groupings in textual form: a list of groups, each of which can be opened to display the list of objects in them, see Figure 6. Whenever an object in a list is a text and a location for it is specified, it can be opened in a viewer (the rightmost window in Figure 6). Lists of groups and objects can also be exported in xml-format and viewed in browsers, see Section 12.2.

6 Matrix Manipulation

`Infomat` is a descendant of the reorderable matrix, see Section 2. The rows and columns are displayed in the order of the current row and column groupings. The matrix and the groupings may be manipulated separately, leading to groupings that do not include all objects in the matrix, which means that the full picture (in the overview window) might not summarize the whole matrix. A *purge* function removes all objects that are not in the currently viewed groupings (and the corresponding matrix elements) from the matrix and all other groupings.

This section describes some of the possible manipulations that can be done to the matrix and the groupings. Several methods are dependent on a similarity definition between row objects and between column objects. There is a separate window where the user can choose which similarity measures are used, one for the rows and one for the columns.

6.1 Feature Selection and Weighting

Both rows and columns (as well as entire groups) may be deleted from the groupings or the matrix in several different ways. They can be deleted directly in the picture or via the pixel window (see Section 5), both being manual (visual) feature selection. There are also several automatic methods for object removal, such as a very flexible stoplist function and several other filtering methods for both row and column objects as well

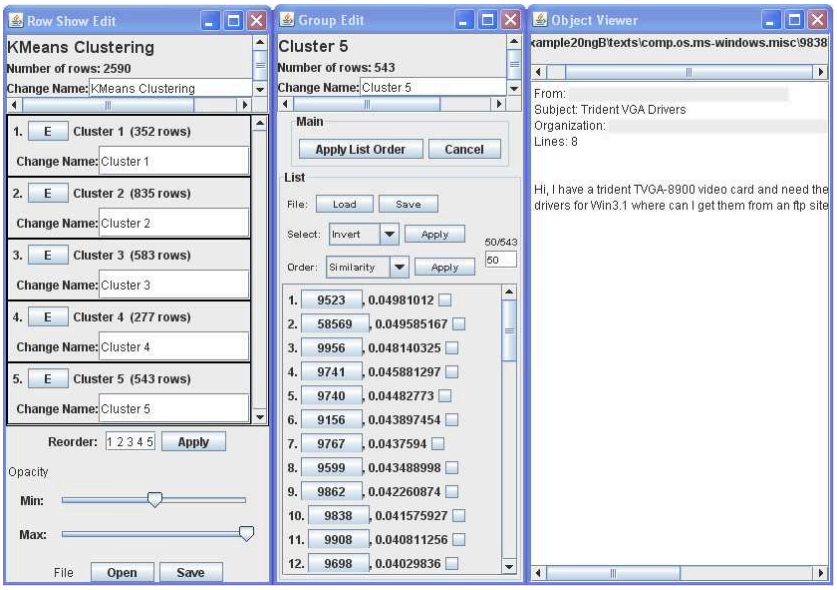


Fig. 6 Textual presentation. The left grouping window can be opened from the main window, see Figure 4. Pressing the "E-button" for any of the clusters in the list opens a group window as the middle one here. It shows a list of the objects in the group in their current order with order value. Pressing any object presented as a button opens up an object viewer, the right window. The top part indicate the location of the text file. We have concealed the identity of the person that wrote this particular post, which we chose since it was short. We also remove the head of the posts in the preprocessing. The bottom section of the left window allows the user to reorder the clusters, change the opacity of the picture and open/save whole groupings. The middle section of the middle window allows the user to reorder the list of objects in the group and apply this new order to the picture.

as matrix elements. One particular type of object removal will be described in Section 8.

Each matrix element contains the original count associated with the row-column-objects-relation. From these weights can be calculated. There are a few weighting schemes implemented which can be used to re-weight the matrix, or a new smaller one, after removal of row and column objects, for instance: an ordinary tf*idf scheme and a scheme that gives each element the original count as its weight.

6.2 Ordering

The order of the objects along the rows and columns of the matrix exposes similarities. Objects with similar distributions that are placed close to each other reinforce each other in the picture.

There are many possible ways to order a set of objects. `Infomat` presents objects in groups, and groups in groupings. Objects are ordered within groups, and groups within groupings. Lists of matrix elements, objects, and groups, as in Figures 4 and 6, can be resorted in different ways: in order of similarity to the group centroid or a single object, in alphabetical order, in random order, the reverse order of the current,

manually, etc, depending on what is sorted. Such lists can also be saved to files for further studies, and be imported again to be used for manipulations.

The order of groups in groupings and objects in groups can be applied to the picture, which may lead to new patterns and insights. We describe one particular ordering in Section 9.

Groupings are primarily created using clustering algorithms, of which a few basic are available, so far. The focus has been on fast methods to make the interface interactive. It is also possible to import/export any kind of grouping to `Infomat` through an xml-format[6].

7 Evaluation

This section describes the many evaluation and comparison possibilities in `Infomat`. These help the user to identify groups and/or groupings of higher quality.

7.1 Automatic Evaluation

Automatic clustering evaluation can be either intrinsic or extrinsic. Intrinsic evaluation uses the same information that were available to the clustering algorithm, i.e. the representation and the similarity measure. An example is the average similarity of pairs of texts within each cluster.

Extrinsic evaluation uses information that were not available to the clustering algorithm. The clustering result may be compared to another partition of the texts, like for instance a manual categorization. By counting the number of texts in both clusters and newsgroup categories we have constructed the confusion matrix in Table 1. An example of an extrinsic measure that is defined on the confusion matrix is the mutual information between a clustering and a categorization, see [Strehl, 2002].

In `Infomat` groupings can be evaluated using intrinsic measures and any pair of groupings along the rows (or columns) can be compared using extrinsic measures. The evaluation is performed in a separate window where you can choose which groupings to evaluate. The result is presented as a long list of measures, including overall measures for the entire grouping and measures for each group of the grouping(s).

7.2 Visual Evaluation

As previously said, the differences in word distribution between the clusters are quite obvious in Figure 1:c and discernible in b. This constitutes a visual intrinsic evaluation; the more apparent the diagonal pattern of Figure 1:c the better clustering. It is possible to see if a clustering or cluster contains similar texts, and which words are important for this similarity.

`Infomat` allows the user to visualize two row groupings at the same time by color coding. The color of the pixels is decided by the group membership of the row objects in the second grouping[7]. In Figure 7 the rows of Figure 1:c has been assigned colors

[6] These can also be viewed in a browser, as described in Section 12.2.

[7] That is: the average color of the row objects corresponding to the pixel.

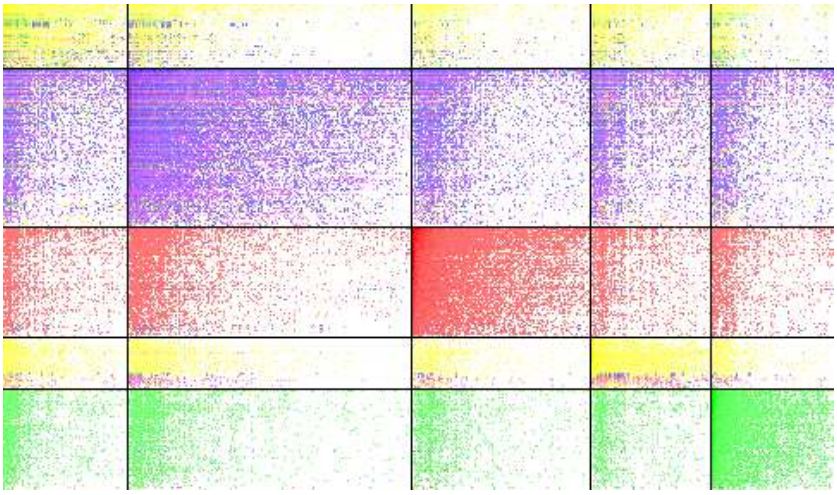


Fig. 7 Visual external evaluation. Same as Figure 1:c with coloring according to the newsgroups, see Table 1. The medical posts are divided between clusters one and four (see Section 8). Cluster two contains essentially all posts about politics and religion, while cluster three and five are rather pure with respect to the sports and computer posts.

	Text Clusters						Color in
Text Categories	1	2	3	4	5	Total	Figure 7
talk.politics.misc/	32	420	4	7	2	465	blue
sci.med/	276	62	2	240	5	585	yellow
rec.sport.hockey/	4	7	575	4	3	593	red
comp.os.ms-windows.misc/	28	6	1	4	531	570	green
talk.religion.misc/	12	340	1	22	2	377	magenta
Total	352	835	583	277	543	2590	

Table 1 Confusion matrix comparing the text clustering, from Figures 1:b and c, to the newsgroups. See Figure 7.

depending on which newsgroup they belong to. See Table 1 for which group is assigned which color. By studying the distribution of colors over the row clusters (from the K-Means clustering) a visual extrinsic evaluation is performed, comparing the clustering to the newsgroups.

If there are two column groupings as well the background of the picture could be drawn as vertical lines colored in the same manner. There is of course a limit when the number of colors just cause confusion!

Color coding can be used for other purposes as well. If we have a categorization (that can be constructed according to any external criteria) it can be interesting to find clusters (generated only from the information in the matrix) that have a high percentage of objects from any of the categories. These clusters indicate a relationship between the information in the matrix and the external categorization. See Section 11.

8 Cluster Descriptions

The word clusters of Figure 1:c constitute extensive text cluster descriptions that can be browsed and explored as previously described. The distribution of the matrix elements over the intersections of word and text clusters gives much more information of the "reasons" behind the clustering than a few words of a textual cluster description; the clustering algorithm uses the entire distribution.

Similar word clusters could be constructed using any text cluster description extraction method. (See Manning et al. [2008] for a discussion of some different methods.) The distributional patterns that these give rise to can provide further insights.

To help in the exploration of clusters `Infomat` has functions for presenting only the top objects of each group in a grouping. Figure 1:d shows the text clusters and the top ten words in each word cluster. In Table 2 these words are given. The picture allows us to study how these words are distributed over the texts, information which is not present in textual presentations such as the cluster digest.

There is a very heavy rectangle in the top part of the fourth text cluster for the fourth and relative word cluster[8]. These are all texts written by or answers to the same person in the medical newsgroup (sci.med). He used a signature that appears in all of them. (We have removed some of the words in our presentation to not reveal his identity.) Text cluster four seems to be formed around the posts by this person (but this is not necessarily true). As he wrote about medical matters the other texts in the cluster are similar. Recall from Table 1 that text cluster four is primarily from the medical newsgroup. This is also confirmed in the picture as there are some words in word cluster one that is common in both text cluster four and one, which is also about medical matters.

In a similar fashion Figure 1:e shows the top ten texts in all text clusters and the entire word cluster. Here we see that the top texts of cluster two, which contains texts from political and religious discussions, use a lot of different words. The sports discussions in the top ten texts of text cluster three use fewer and much more specific words.

It is important to note that there might be many texts that do not contain any of the words in either of the word clusters in Figure 1:d although it appears as if they all do. This is because of the weight averaging for pixels as described in Section 4. Similarly for the texts in Figure 1:e, and both texts and words in Figure 1:a through c. Only a close inspection can reveal local details.

Figure 1:f, finally, shows only the top ten objects from all text and word clusters. Here the signature of the person posting about medical matters is very apparent. Overall the picture looks very orderly; there seems to be a set of words that describe each text cluster fairly well. This corresponds to the cluster digest, as described in Section 2.2. The previous pictures have demonstrated that this does not give the whole picture, and might even be deceiving.

9 In Order of Appearance

There are many orderings of the texts and words that could be interesting to try. One natural order is the order in which the words are first are read. This order is available as `Infomat` assigns an id number to all objects when they are created.

[8] You may also be able to see this in the zoomed-in main view of Figure 4.

	Word Clusters				
Word no.	1	2	3	4	5
1	doctor	god	playoff	*	driver
2	diseas	govern	nhl	*	card
3	medi	law	season	*	ftp
4	info	moral	score	chasiti	version
5	patient	jesu	cup	*	font
6	treatment	clinton	fan	intellect	video
7	medicin	gai	goal	*	printer
8	pain	tax	detroit	surrend	disk
9	*	homosexu	wing	skeptic	soft
10	appreci	*	leagu	shame	instal
⋮	⋮	⋮	⋮	⋮	⋮
No. of words	1380	3122	1975	1326	1365

Table 2 Word clusters that can be considered text cluster descriptions. The top ten words for the word cluster of the word clustering relative to the text K-Means clustering, see Figure 1. Also compare with Table 1 and Figure 7. All strings here are stems, not real words. In cluster four all top ten words are from the signature of one poster in the medical newsgroup; it contained the sentence "Skepticism is the chastity of the intellect, and it is shameful to surrender it too soon." and some contact information. We have concealed (*) words that indicate his identity. Two more words are concealed. They are names of two other posters.

As we first loaded the small 20 newsgroups text set we obtained the picture in Figure 8:a. New words (columns) were found as new texts (rows) were read. However, a few texts seemed to have a huge amount of words that never appeared in the other texts (whole horizontal lines). After inspection we found that the preprocessing was not perfect: some posts included some kind of data (an image perhaps) that in our files were represented by a long list of space separated character sequences. All these were read as "words". After we had removed these texts (they included very little more) we got the picture in Figure 8:b as a result. In this way the overview of `Infomat` can give information of anomalies in the data.

The texts were read newsgroup category by category from separate catalogs. `Infomat` can extract a categorization from the file name paths. In Figure 8:b we have inserted separator lines between them. New, category specific, words is introduced by each category. It is possible to create word clusters that split the words by in which category they first appear (not shown).

For any text grouping we can create a pseudo ordering by considering which words appear in the texts in the order they have in the groups[9]. In Figure 8:c we have done this for the K-Means clustering used in the previous examples. We have also split the words into clusters accordingly. In Figure 8:d we only put words that appear in at maximum two text clusters in the corresponding word cluster. The rest are put in a rest cluster.

The word clusters in Figure 8:c and d are alternative text cluster descriptions that might reveal other interesting aspects.

[9] However, `Infomat` do note store the order of words within texts.

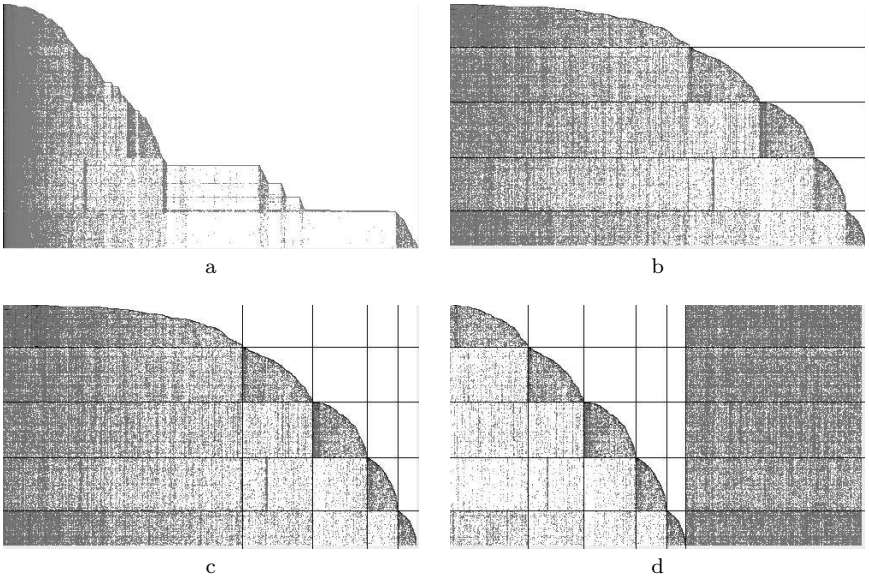


Fig. 8 The order in which words appear, see Section 9. Figures a and b show how new words are encountered as the texts are read one by one from catalogs corresponding to the newsgroups. Some words were uninteresting character strings representing some other kind of data (a picture?) (Figure a). In Figure b the corresponding texts are removed and separator lines inserted between newsgroups. Figure c shows the same K-Means clustering of the texts as in previous Figures and a pseudo order of appearance of the words based on it. In Figure d only words that appear in two or fewer text clusters are put in the corresponding word cluster, the rest in a rest cluster.

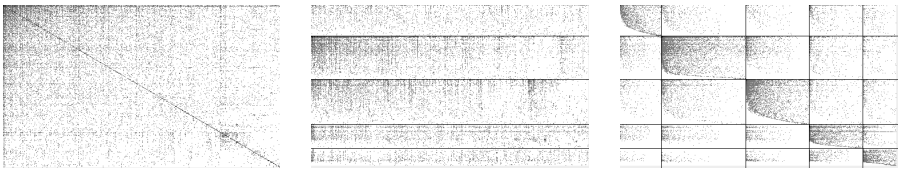
10 User Survey

In 2008 we used an earlier version of `Infomat` in a PhD course on text clustering with seven participants, all doing there work in natural language processing and proficient computer users. As part of the first assignment the students were asked to make themselves familiar with `Infomat` by studying the text set that has been used as an example here. They also answered a small questionnaire about the program. We present some of the results briefly here.

Although these students were quite familiar with the term-document-matrix they did not fully understand the picture until we explained it a second time. That a pixel line represents several texts and a pixel represents several matrix elements (see Section 3) is rather hard to grasp. However, when they understood they found the interface useful and intuitive.

They all agreed that the different patterns were interesting and useful: the word appearance in Figure 8:a (they did not see the other pictures in Figure 8), and especially the relative clustering, see Figure 1:c. Most of them found this enough to understand the content of the clusters by browsing and using the pixel window, see Figure 4, although all agreed that the possibility to read the actual texts were good.

They were also all positive to the visual evaluation as a means to extend the overview of the data, although most thought that if several groupings were to be compared ordinary measures would be better.



a b c

Fig. 9 Word-co-occurrence matrix for the 5272 words in the newsgroup text set that co-occur five or more times within a sliding window. The sliding window was defined as consisting of two words before and two words after the current word. a) The order of the words along rows and columns in which they were encountered. b) A K-Means clustering to five clusters along the rows and the initial order along the columns. c) The same K-Means clustering of the rows and a relative clustering of the columns.

There were many positive comments and suggestions of new features, some of which we intend to implement in time. We also got some problems in the program pointed out to us that we have corrected.

11 Some Other Applications

`Infomat` may be used in several areas other than Information Retrieval. As soon as you have a representation of objects in a matrix it may be visualized in the same way. The matrix is loaded in a general xml-format and small matrixes are handled as well.

Here we will describe some possible applications in information retrieval. The first one is of course what we have already been presenting; as a tool for general exploration of text set content following in the tradition of the Scatter/Gather textual clustering system [Cutting et al., 1992].

The number of possible external orderings and groupings of a set of texts is enormous. Texts are often associated with other data and it is often possible to order the texts based on these. It could be when the texts were produced, if they were a response to another text, who wrote them, who cited them, and all sorts of other data. All these kinds of orderings and groupings could be presented in `Infomat` and may give rise to new patterns.

A special case is a questionnaire, where free text questions are sometimes mixed with closed questions. In [Rosell and Velupillai, 2008] we used this to find relations between occupations (a clustering of a questionnaire text answer) and if the respondents are smokers or not (information extracted from a closed answer in the same questionnaire). Quite similar to questionnaires are, in this respect, medical records, that contain both texts and for instance measurements and medical ordinations.

Another possible application of `Infomat` would be to use paragraphs instead of texts. This could be used to study how the vocabulary changes over a longer text; topic shifts may become visual.

The other often studied matrix in information retrieval is the word-co-occurrence matrix. Figure 9 shows this on the example text set.

We also found `Infomat` to be a good educational tool, see Section 10. Students get to see the huge representation matrixes (term-document-matrix).

12 Implementation Aspects

`Infomat` consists of about 40 000 lines of Java[10] code and is freely available[11] following the GNU General Public License[12]. The program can be considered consisting of two parts: the graphical user interface, which is what is discussed in this paper, and command prompt usage classes that output textual results (see Section 12.2 for a brief discussion). There is a manual included in the download. `Infomat` has been tested on Unix and Windows.

The program runs in primary memory so there is limitations on how much data it can cope with. Also the calculations takes more time for more data. We have used a text set consisting of 42 000 texts with 6000 different words with quite reasonable waiting times on an ordinary laptop[13] (2008). The unprocessed word-co-occurrence matrix that we constructed in order to present Figure 9:a consisted of 33 600 words along both rows and columns. It was slow to process, but we removed most of the words using one filtering call.

Matrixes, groupings and groups are imported/exported in an xml-format, so any data could be handled. The picture construction is optimized for sparse matrixes, but smaller dense matrixes should work fine as well.

The following subsection discusses the main data structure used in `Infomat`, and the one after that another important data structure that is used for, among other things, setting up experiments outside of the graphical user interface.

12.1 Main Data Structure and Picture Construction

We implemented a *double hashmap*, which works as a hashmap with two indexes. It is used to store the sparse matrix with the objects as indexes and matrix elements as values, and allows for fast access to the non zero elements, as well as the set of non zero elements for any row or column. This makes it equally fast to retrieve and cluster both rows and columns.

The sparse matrix double hashmap (W in Figure 5) facilitates flexible construction of the picture based on any row and column ordering (grouping). In the picture construction another double hashmap (P in Figure 5) is created. For each pixel (x and y coordinates) it stores a pixel object, that in addition to the opacity keeps track of all the matrix elements associated with that pixel. The picture is painted using this double hashmap; only the non empty pixel objects result in plotted pixels. As the user moves the mouse pointer over the picture the double hashmap is used to look up which matrix elements the current pixel corresponds to.

When the rows and columns are divided into groups the picture is constructed considering each row and column group intersection as one separate picture. This allows the group separator lines to be inserted and removed fast. Such division of the picture into parts could also be used to speed up the construction by parallelization.

The time required to construct the picture is linear in the number of non zero matrix elements of the sparse weight matrix, plus linear in the number of non zero

[10] http://java.sun.com/

[11] http://www.csc.kth.se/tcs/projects/infomat/infomat/

[12] http://www.gnu.org/licenses/gpl.html

[13] Intel pentium processor 1.6 Ghz and 1 GB RAM. Windows XP.

pixels in the picture hashmap (see Figure 5). For sparse weight matrixes this is much faster than going through all matrix elements. However, it is still highly dependent on the graphics engine of the particular computer.

12.2 Properties and Experimentation

Many methods in `Infomat` have different parameters that can change their behavior, such as for instance how the initial grouping is constructed in K-Means. To keep track of these and to be able to present them and let the user alter them we implemented a *properties* class. Other classes with options use it and the parameters are presented to the user in a uniform way.

Properties can be saved and loaded as xml-files. By gathering such files in a catalog a clustering set up can be defined. A command prompt clustering class can run this clustering by specifying the catalog. The result is several other xml-files with the clustering. They can be viewed in a browser through a main page with a set of clusters with cluster digests. Each cluster is a hyper link to a page with more information about the cluster. The texts in the clusters are also links which leads to the actual texts if the program have this information. (Compare with Figure 6.)

It is also possible to set up a series of experiments consisting of several clusterings with different parameters by placing properties files in a catalog hierarchy. A special experimentation class can run these with evaluation result files as the output. There are also a class for summarizing the experiments that outputs tables in several different formats for presentation and processing in other programs (LaTex, MS Word, and Matlab). For more information see the manual[14].

13 Conclusions and Further Work

The graphical user interface of `Infomat` presents an information retrieval matrix directly instead of via calculation of the similarities between objects (text and/or words). Similarities between objects appear as distributional patterns as the matrix is reordered along rows and columns. Orderings of different kinds can reveal different relations between the objects. In this way, when earlier methods only inform the user *that* objects are similar (displayed near each other in a picture), `Infomat` reveals *why*.

Text clustering results are sometimes hard to grasp. `Infomat` allows the user to see how the words are used in the different clusters, something which is not possible using textual cluster descriptions. The distributional patterns give an intuition of the "reasons" behind the clustering.

The compressed picture reveals general trends, but at the same time hides local details. It is important to remember, that although parts of the picture might seem dense, the underlying matrix may be sparse. The zoom function lets the user explore the local details.

There are many possible functions that could be added to `Infomat`. We will implement a search tool that allows the formation of groups of objects that match a query. Among other clustering algorithms we are particularly interested in co-clustering algorithms. That is, algorithms that clusters both objects and features at the same time.

[14] See http://www.csc.kth.se/tcs/projects/infomat/infomat/

Also it could be beneficial to add some two-dimensional projection methods to provide alternate views.

Acknowledgements Thanks to all participants of the Infomat project[15] and the students who took our text clustering course.

References

D. Asimov. The grand tour: a tool for viewing multidimensional data. *SIAM J. Sci. Stat. Comput.*, 6(1):128–143, 1985. ISSN 0196-5204. doi: http://dx.doi.org/10.1137/0906011.

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

J. Bertin. *Graphics And Graphic Information-Processing*. Walter de Gruyter & Co., Berlin, 1981. ISBN 3-11-006901-6.

D. R. Cutting, J. O. Pedersen, D. Karger, and J. W. Tukey. 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*, 1992.

I. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *KDD '01: Proc. 7th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 269–274, New York, NY, USA, 2001. ACM.

I. Dhillon, D. Modha, and W. Spangler. Class visualization of high-dimensional data with applications. *Computational Statistics & Data Analysis*, 41:59–90, 2002. ISSN 0167-9473.

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

R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley, 1992. ISBN 0-201-50803-6.

S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall International, Upper Saddle River, NJ, USA, 1999. ISBN 0-13-908385-5.

M. Hearst. *Modern Information Retrieval*, chapter User interfaces and visualization, pages 247–323. Addison-Wesley, 1999. ISBN 0-201-39829-X.

A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

K. Lagus, S. Kaski, and T. Kohonen. Mining massive document collections by the websom method. *Inf. Sci.*, 163(1-3):135–156, 2004. ISSN 0020-0255.

K. Lang. Newsweeder: Learning to filter netnews. In *Proc. of the Twelfth Int. Conf. on Machine Learning*, pages 331–339, 1995.

C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. ISBN 978-0521865715.

M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

R. Rao and S. K. Card. The table lens: merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 318–322, New York, NY, USA, 1994. ACM Press.

[15] A project during which we started to implement the program `Infomat`, see http://www.csc.kth.se/tcs/projects/infomat/

M. Rosell and S. Velupillai. Revealing relations between open and closed answers in questionnaires through text clustering evaluation. In *Proc. of the 6th Int. Language Resources and Evaluation (LREC'08)*, 2008.

J. Seo and B. Shneiderman. Interactively exploring hierarchical clustering results. *Computer*, 35(7):80–86, 2002. ISSN 0018-9162.

J. Seo and B. Shneiderman. Knowledge discovery in high-dimensional data: Case studies and a user survey for the rank-by-feature framework. *IEEE Transactions on Visualization and Computer Graphics*, 12(3):311–322, 2006. ISSN 1077-2626.

A. Strehl. *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*. PhD thesis, The University of Texas at Austin, 2002.

C. J. Van Rijsbergen. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow, 1979.