Approaches to better context modeling and categorization

Madsen, Rasmus Elsborg; Hansen, Lars Kai; Larsen, Jan

Publication date:
2006

Document Version
Publisher's PDF, also known as Version of record

Approaches To Better Context Modeling And Categorization

Rasmus Elsborg Madsen

Kongens Lyngby 2005
IMM-PHD-2005-70

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

IMM-PHD: ISSN 0909-3192

Summary

This thesis investigates the role of three generally accepted assumptions that are made when mining text documents. The three assumptions are: that all words in a document are equally informative, that the order in which the words appear is non-informative, and finally the independence assumption that result in a non-burstiness assumption. The three assumptions simplifies many aspects of context-mining, but are never the less not true and result poor document modeling and categorization. Existing models has been improved by use of neural network sensitivities, natural language features, and a new probabilistic model that can adapt burstiness has been developed. These new approaches have generally resulted in better context modeling and ability to categorize documents better.

Resumé

Denne thesis undersøger indflydelsen af tre generelt accepterede antagelser, som anvendes ved analyse af text dokumenter. De tre antagelser er følgende: at alle ord i et dokument er lige informative, at den rækkefølge ordene optræder i et dokument, er ikke-informativ, samt en uafhængigheds antagelse der resulterer i en antagelse om at ord ikke forekommer i "bursts". Disse tre antagelser simplificerer mange aspekter ved emnet "context-mining", men er usande og resulterer i dårligere modellering og classificering af text-dokumenter. De eksisterende modeller er blevet forbedret ved brug af "neural network sensitivities", "natural language features", samt en ny sandsynlighedsmæssig model der kan modellere fænomenet at ord forekommer i "bursts". Disse nye metoder har resulteret i bedre modellering ad text samt bedre klassificering af texter.

Preface

This thesis was prepared at the department of Informatics Mathematical Modelling, at the Technical University of Denmark in partial fulfillment of the requirements for acquiring the Ph.D. degree in engineering.

The thesis deals with different aspects of mathematical modeling, in the area of modeling collections of text documents. The main focus is on generative probabilistic text models, but other machine learning approaches to context mining are considered.

The thesis consists of a summary report and a collection of seven research papers written during the period 2003–2005, and elsewhere published.

Lyngby, September 2005

Rasmus Elsborg Madsen

Papers included in the thesis

[A] David Kauchak, Rasmus Elsborg Madsen and Charles Elkan
Approximating the Dirichlet Compound Multinomial Distribution. *Technical Report* 2005

[B] Rasmus Elsborg Madsen, David Kauchak and Charles Elkan
Modeling Word Burstiness Using the Dirichlet Distribution. *To appear in: Proceedings of the International Conference on Machine Learning (ICML)* 2005

[C] Rasmus Elsborg Madsen, Sigurdur Sigurdsson, Lars Kai Hansen and Jan Larsen
Vocabulary Pruning for Improved Context Recognition. *Proceedings of the International Joint Conference on Neural Networks (IJCNN), pp. 80-85* 2004

[D] Rasmus Elsborg Madsen, Lars Kai Hansen and Jan Larsen
Part-of-Speech Enhanced Context Recognition. *Proceedings of IEEE Workshop on Machine Learning for Signal Processing (MLSP) XIV, pp. 635-644* 2004

[E] Rasmus Elsborg Madsen, Sigurdur Sigurdsson and Lars Kai Hansen
Enhanced Context Recognition by Sensitivity Pruned Vocabularies. *Proceedings of 17th International Conference on Pattern Recognition (ICPR), vol. 2, pp. 483-486* 2004

[F] Rasmus Elsborg Madsen
Modeling Text using State Space Models. *Technical Report* 2004

[G] Rasmus Elsborg Madsen
Multi-Subject fMRI Generalization with Independent Component Representation. *Technical Report* 2003

Acknowledgements

I want to thank my supervisors Lars Kai Hansen and Jan Larsen for our working relationship during the Ph.D. work. I would also like to thank Charles Elkan for inspiration during my research stay at the University of California, San Diego.

Contents

Chapter 1

Context Mining

1.1 Introduction

Back in times before we had computers, all of our external knowledge was organized in libraries, where librarians were the ones to organize all the knowledge and wisdom contained within the library. This organization of the library is crucial if the people who uses the library should be able to find what they were looking for. All the books in a library have therefore been manually categorized into whatever topic that was mainly considered in the book, and was given an indexing number that symbolized this category. For the topic "biology", the indexing number could e.g. be 58. After some time a topic could evolve so that subtopics would emerge. A subtopic of "biology" could be "micro biology", who would just be sub-indexed from it's super-topic, i.e. 58.34. Sometimes a new topic would emerge that was a combination of separate topics, that could be "bio chemistry" which is a combination of "biology" and "chemistry". While this combined topic could not logically be inherited by both its ancestors using the numbering system, it could become a sub-category under just one of it's category-ancestors or it might even get it's own main category. This combined category issue is of course inexpedient for the library categorization system. Since the growth of information was relatively small and topics evolved slowly, catalogs and thesaurus's could be made to let the library users know how the information was indexed. The evolution of knowledge did therefore not pose

a threat to the library way of structuring all the knowledge and wisdom. But then the Internet emerged...

1.2 Evolution of the Internet

In response to the USSR launch of Sputnik in 1957, the United States formed the Advanced Research Projects Agency (ARPA) to establish a US lead in science and technology for military purposes. But it wasn't until 1962 that ARPA started researching the internet, after commissioned to do so by the United States Air Force (USAF). The USAF wanted a military network wherefrom they could maintain command and control over their missiles and bombers in case of a nuclear strike. Seven years later in 1969 the physical ARPANET was created linking four host computers together, hereby conceiving the Internet.

But there would still be a long way ahead before the Internet opened up and became useful for the public. The e-mail system was invented in 1971 and quickly became a popular tool among researchers connected to the ARPANET. In 1982 the TCP/IP protocol was defined, a protocol defining how to send data-packages between computers connected to the Internet. At this time "the Internet" was also defined as all the networks connected by TCP/IP. The World Wide Web (WWW) was subsequently invented in 1991 allowing users to access files on the Internet by use of hypertext links. The WWW was created by Tim Berners-Lee who hereby created a better way of locating documents and other content files. The job was done at the European Particle Research Laboratory CERN, who's information databases were becoming overwhelmingly large and hard to navigate through. By 1993, 200 WWW servers were connected to the Internet, allowing mainly researchers all over the world to access the Internet content.

Falling prices on Internet access allowed the public in large numbers to begin accessing the internet in the late 1990's. A whole new basis for use of the Internet was hereby established. Applications and ideas for ways to use the internet have continued accelerating and today (July 2005) more than $67,500,000$ WWW hostnames are represented on the internet with a monthly gain of 2.76 million new hostnames (Bergman, 2001; Mauldin, 1995)[1].

[1] Web server surveys are uploaded monthly on www.netcraft.com

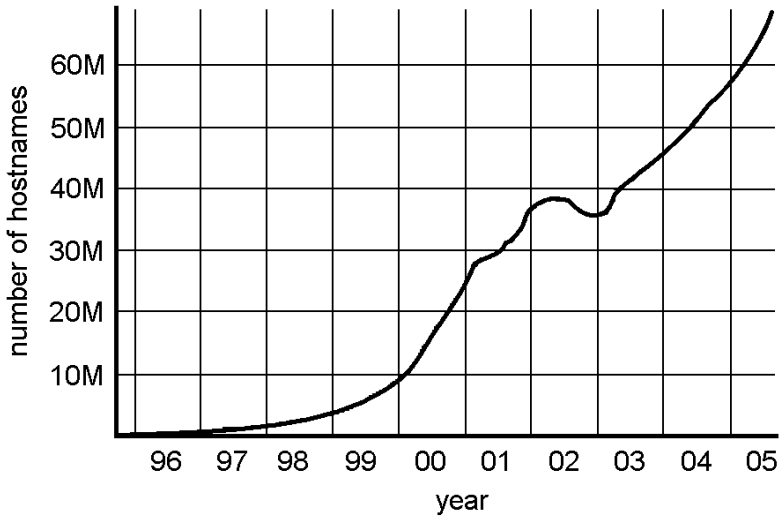


Figure 1.1: 10 years growth in the number of hostnames accessible on the internet. Except from a small fraction of time in 2002, the amount of hosts has grown rapidly for 10 years, counting more than 67 mill hosts today. The graph reflects the data from the web-site: www.netcraft.com.

1.3 Growth of Public Searchable Information

The growing number of Internet users, which today is more than $938,000,000$[2] users (14.6% of the worlds population), and WWW servers has resulted in an enormous continuously growing amount of information available to the public (Lyman & Varian, 2003). Google indexes more than 8 billion[3] web-pages today, and the number of indexed web pages keeps growing continuously.

Not only the amount of web servers but also the worldwide information production has increased a lot. Between 1999 and 2002 the information production increased by 30 percent each year, and a bigger fraction of the information is being published on the web (Lyman & Varian, 2003). An estimate of the amount of information generated in the year 2003 was determined by the School of Information Management and Systems at the University of California at Berkeley. Researchers found here that the world produces between 1 and 2 exabytes of unique information per year, which corresponds to 250 megabytes per person. One exabyte is a billion gigabytes or 10^{18} bytes. Only a tiny fraction of this information, approximately 0.003% or 30 terabytes, is written information which

[2] www.internetworldstats.com
[3] July 2005 at www.google.com

is by far the most information dense media.

Much of the generated information is published on other medias than the Internet, but the trend is that more and more information is being published on the Internet. The Internet has become a flexible and fast place to publish, allowing to reach much more people than with the old medias. This is especially true in the area of research, where it previously was hard and expensive to find relevant research articles. At Citeseer[4] they have found that articles freely available online have much higher citation rate than other articles (Lawrence, 2001) and are faster at impacting other scientific researchers.

The Internet has further resulted in generation and publication of much information that would never have been available without it. The phenomenon blogs[5] has given the public a way to reach others with political comments, travelling diaries or whatever the author might want to say to the public. Internet forums in all its subspecies, have further allowed to share and comment on all sorts of information. Many other examples exists of information that wouldn't have been created and published, if it wasn't for the Internet.

The total resulting growth of information is often referred to as the Information Explosion, which especially impacts the internet while a growing part of all the published information becomes available to us from the World Wide Web. Due to this information growth, it becomes more and more difficult to find relevant information on the web and other huge, complex and distributed databases[6]. The task of finding relevant information, getting an overview of some information or in general navigating through large amounts of information, is referred to as data mining, or in the case of working with texts information, simply text mining.

1.4 Text Mining

One side effect of the rapid evolution of the content on the Internet is that the world wide web has become an unstructured and fast growing database, making it hard to find the information needed. It would take almost an infinite amount of human effort to structure all this information manually into a logic database, making the information gathering problem uncomplicated. Though

[4] citeseer.ist.psu.edu

[5] A blog (shorthand for web log) is a frequently updated webpage consisting of dated entries arranged in reverse chronological order. By March 2005 more than 8.7 mill blogs existed (www.blogcount.com)

[6] This was also the problem CERN was facing when developing the WWW, just in a smaller scale.

search engines let web users find a lot of valuable information, many users are still left in frustration by the low precision and recall of these searches (Chakrabarti, 2000).

The amount of internet and e-mail users have made it feasible to market products using spam[7] e-mails. Due to the non-existent cost of sending an e-mail, many companies have turned to this strategy to boost product sales being a nuisance to many web users who have their e-mail in-boxes filled with spam daily. Though spam filters exist they far from catch all the spam mail and by using the filters one risks to loose important non-spam e-mails.

Text-mining covers a range of different approaches to e.g. search for, filter, structure and match unstructured information. It is widely believed that machine learning techniques will come to play an important role in these text-mining problems, and have already found it's way into many text-mining applications.

An example of a company using text-mining is Whizbang Inc., who produced the job service Flipdog.com, now part of the leading job search site Monster.com. Flipdog and Monster partly uses machine learning techniques to match CV's with job adds. Another example of a text-mining company is "Citeseer" who finds the reference structure associated with given a scientific document. IBM's WebFountain (web,), the WEBSOM project (Lagus et al., 2004) and the Stanford University semantic web platform TAP (Guha & McCool, 2003) are other examples of text-mining coming into play, making human information navigation easier.

Some other interesting text-mining tasks is the task of finding relevant products, based on customer reviews for new items and previously purchased items[8]. Another application is structuring or filtering of news articles from the media companies like Reuters, CNN, BBC etc, making it easier to find the relevant news and memes[9]. Structuring web-pages or other text content in a hierarchical manner would make internet searches more precise. Google and Yahoo are already structuring a smaller part of their indexed web-pages this way, hereby organizing the web in a kind of library system.

In this thesis, the focus is set on text categorization, the information retrieval aspect of text/web mining studies (Kosala & Blockeel, 2000; Sebastiani, 2002). A supervised approach to the text categorization task is considered, where an annotated corpus of documents is to be classified into a set of categories. While most of the methods considered are easily portable to the unsupervised text

[7] As of May 2003, 55 percent of all e-mails were spam (Lyman & Varian, 2003)

[8] The company Amazon is an example of a company with a huge database of product reviews. It is not unusual with more than 100 reviews for a single product

[9] Memes are new emerging topics

categorization task (clustering), the focus is kept on the supervised case while it allows for more accurate performance measures and better comparison with other methods.

1.5 Problems in Text Categorization

Although text is easily understandable to humans, it is still very difficult to have machines make sense of text sentences. One of the biggest problems is that text in it's true nature is almost infinite dimensional[10] for machines. The reason that humans can work with these high dimensional problems is our ability to grab the essential information from a text, and discard the rest of the information. From a sentence of ten words, the underlying meaning may be contained in five of the words and few relations among these words. The number of sentences that can be written in a ten word text is also far greater than the number of meanings that can be represented with a ten word text. We come to this conclusion since the same meaning can be written using different words in different order. The space of meaning is therefore much smaller than the space of text we use to represent the meaning. It is however difficult to make machines understand meaning, at first because we don't have any other form than text to represent it, and second because humans have a huge library of experience and common sense that is used to decipher text into meaning.

The nature of text is a combination of words and the order in which they appear. The order in which the words appear carry much less information than the words themselves, and humans seem to understand something about a text by seeing which words are contained in the text, while none of the context can be recovered from knowing the order in which some unknown words appear. The order in which words appear can be very important though, when trying to make sense out of a text. The two sentences "are you old" and "you are old" uses the same words, but the changes in word order, changes the meaning of the sentences from question to fact. If we look at the word order alone, we wouldn't be able to make much sense out of it. The representation of the semantics for the two sentences could be "1 2 3" and "2 1 3" which wouldn't make any sense at all. One reason that the word order carries little information about a text is that much word order information is contained in the grammatical rules of text, limiting the ways in which we can construct a sentence. Given the

[10] When a new sentence is written, the author can make use of all of the words in the Oxford English dictionary, which currently lists about 500,000 words. In the following example we assume that English grammar limits us to selecting only 2% of the words every time we add a new word to a sentence. Generating a sentence with 10 words will therefore allow us to create 10^{40} different kinds of sentences.

words “are”, “how” and “you”, the grammatical rules dictate that the sentence must be “how are you”. In special occasions the grammatical rules are broken, making the word order extra valuable. An example of this could be when Yoda is speaking in Star Wars. Having this word order information would definitely be a valuable feature for determining if the text is about Star Wars. Common sense is another factor that reduces the information in the semantic part of the text representation. Common sense allows us to discard various semantic constructions of texts if they make no sense. An example of a text that lacks common sense is “the baby carried the house to the mother”, while “The mother carried the baby to the house”, would make more sense.

Since the information about what words that occur in a text is much more valuable than the order in which they occur there is an easy way to transform text onto a much smaller space which can be used for computers and machine learning algorithms to make sense of text. By considering only the counts[11] of the words that appear in a text and not the order in which the words appear, text of any length is transformed from an infinite dimensional representation to a finite[12] dimensional representation, allowing statistical machine learning algorithms to come into play. The transformation is not only simple, but also contains most of the information from the natural text. Text categorization applications of today are therefore satisfied with this way of representing text.

Human language contains a lot[13] of -onyms, which are words with a particular property. Some of these properties make the accessibility of language difficult, while the full meaning behind the words are hidden, in some sense. A few of this -onyms are explained in the following. An acr-onym is a word formed from the initials of one or more words, such as NATO (North Atlantic Treaty Organization) and WWII (2’nd World War). A person who doesn’t know the acr-onym WWII will not have associations to Bismarck, even though the person knows that “Bismarck” was a famous ship in the second world war. Ant-onyms are word pairs that have opposite meaning, such as short and tall, small and large. The big difference of objects described with ant-onyms will not be captured by a person that does not know the words are acronyms. An ex-onym is the -onym for when the name of a group or a place is different, depending on whether you belong to the group or not, or if you live at this place or not. An example of a place is when “Cologne” is used instead of “Köln”. A heter-onym is a word that is spelled in the same way as another but have different meaning. An example is ’bow’ which can be “the bow of a ship” or “a bow used with arrows”. A hom-onym is a word which has different unrelated meanings, such as “fluke” which can be a fish, a part of a whale or a stroke of luck. Use of hom-onyms is

[11] Documents are then represented by histograms of word counts.

[12] Where the number of dimensions needed to represent a text is equal to the number of words in the vocabulary that is considered.

[13] More than 80 different kinds of -onyms exist.

often referred to as polysemy though there is a minor difference. A hyper-onym is a generic word that stands for a class or group of equally-ranked items, such as "car" which is a hyper-onym for the hyp-onyms "hatchback" and "sedan". A par-onym is a word that is related to another word and derives from the same root, such as dubious and doubtful. A syn-onym is words which is equivalent in meaning to another word, such as near and close. While the use of onyms makes the accessibility of language difficult, humans seem well suited to make sense of text content, in defiance of the use of onyms. The huge information database, we all carry in our brains, is probably the reason why we aren't bothered by onyms. In contrast to humans, machines don't have a huge information base which can be used to make sense of the onyms. Machine learning is therefore likely to be penalized when exposed to the onyms.

Document collections that are used to obtain statistical information about word occurrence frequencies for different document categories, are relatively limited in size. We therefore often don't get a sufficient amount of statistical information about how often a given word occurs in different categories. If a word only occurs a few times or doesn't occur at all in a document category, we can not be sure what the cause of this rarity is. It can be that the given word is rarely used in that category, or maybe the word is simply rare by nature. If we consider a document collections with the categories "sports cars" and "oil production", the word "bonnet" might occur once in the category "sports cars" and the word "overwhelming" might occur once in the category "oil production". If we were to categorize a new document based on the statistics from the known documents, a document containing the word "bonnet" would then correctly be categorized as a "sports cars" document. In the opposite case where the word "overwhelming" occurs in a new document, we might be wrong when categorizing it as a oil production document. Humans would be able to perform this task better while we carry a huge information bank in our head that tells us that the word "overwhelming" might not be too important in this categorization task, and should be considered as being noise. Because of the lack of documents and the sparseness of words in the documents, we can only obtain insufficient statistics for many of the words used in the corpora. Many word statistics are therefore very noisy, making it hard to create solid document models. The phenomenon burstiness can further make a statistic for a category look consistent, thereby fooling the document classifier. Finding the words that have consistent statistics that can be used for robust categorization is therefore hard.

The issues described here that make text categorization hard, are some of the most important issues for making machine learning for text categorization reach human performance. Solutions to these issues will be pursued in this thesis.

1.6 Vision for Future Text Mining Applications

With time the logical part of artificial intelligence (AI) will merge with the statistical part of AI in many applications, making AI applications that by far supersedes the single sided AI applications of today, where only the statistical approach is used. One company that pursues the text mining vision from a logic point of view is the company Cycorp[14]. Cycorp is in the process of creating a language ontology and a matching knowledge base and inference engine that can be used for various text mining applications, text understanding and reasoning. Compared with statistical approaches, the logical approach is very extensive and demands a lot of human effort. The Cycorp system is focusing on a lot of other aspects of text mining than text categorization, but the conclusion is still that the logical inference engine is not yet suited for text categorization.

The "Semantic Web" is another exciting vision (Berners-Lee et al., 2001) for approaches to make future text mining applications better at mining the internet. The basic idea with the Semantic Web is to add markup semantics[15] to all the content on web-pages, making it possible for search engines, agents, and other text mining applications to "understand" the content on the webpage. The DARPA[16] group and the "World Wide Web Consortium" (W3C) are some of the providers of standards and tools for semantic web markup. Not all problems with the Semantic Web have been solved yet, and especially agreement on a set of semantic ontologies is essential for further progress for the semantic web vision. Success for a semantic web is however also conditioned on the intensions of web content providers, to start using semantic markup on their web content.

We appraise that the Semantic Web and the logic approach to text categorization may not be for a long time yet. It is further unclear how much the two approaches will impact text categorization tasks, while they both are not directly aimed making this task easier. The Semantic Web and the logic approach to text mining will both contribute to making text categorization better than today though. The Semantic Web markup language layer will for instance contribute to the categorization by having classified the pieces of text, in a document, on micro level already. The logic approach will understand the document content on a higher level, whereby confusion from e.g. synonymy and polysemy will disappear. The statistical approach to text categorization is therefore likely to be useful combined with the Semantic Web and logic technology.

Much of the information contained in a logical database and information bank like the Cycorp database, could be ported into a huge matrix with one row for

[14] www.cyc.com
[15] Semantic markup is similar to HTML markup tags.
[16] www.daml.com

each word in the vocabulary. Each row in the matrix would then contain a set of weights that represent the words association to all the other words in the vocabulary. If two words are synonyms their associated weights should then be high and low if the words were antonyms. Words should also be weighted with respect to their all over discriminative ability. The matrix could then be multiplied onto document vectors adding information from the information bank. When a full matrix was created, it could be used for many different classification tasks. The task of creating an information bank matrix would demand lots of man hours, and is therefore out of the scope of this thesis.

The approaches to text categorization described in this section all demand a lot of manual labor and extensive knowledge, to get it up and running. In this thesis we are instead considering statistical machine learning approaches to text categorization which is far less extensive than the logical approaches described here.

1.7 Reading Guide/Overview

The rest of the content in this thesis is divided into 7 other chapters followed by the conclusion. The following two chapters consider discriminative text classifiers, while the remaining chapters deal with probabilistic text models. The chapters 5-8 all consider the analysis and development of a probabilistic model that can deal with the burstiness phenomenon for text.

- Not all words are equally well suited for discriminating texts. In chapter 2 we address the issue of finding the words that possess discriminative power, which can be useful for text categorization classifiers. The approach we suggest, uses neural network sensitivities to determine which words are best for discrimination, and prunes away words with little and inconsistent ability to be discriminative.

- The issue of removing information about the order in which words appear in text, is addressed in chapter 3. We experiment with fusing natural language features with the commonly used word count features, to capture some of this lost information.

- In chapter 4 we continue to pursue solutions, that can capture some of the information that is contained in the order that words appear in a text. Probabilistic state space models are used to model the joint word order information and word count information.

- The issue of word burstiness in documents is investigated in chapter 5. Commonly used generative probabilistic models do not take the burstiness phenomenon into account when modeling text, making the model probabilities inaccurate. We compare the abilities of the commonly used multinomial distribution and the Dirichlet distribution to model word burstiness.

- The Dirichlet distribution has some inexpediencies that make it poor at modeling sparse data. In chapter 6 we introduce the Dirichlet compound multinomial model, which can model the burstiness phenomenon and at the same time work with sparse data.

- The Dirichlet compound multinomial is good at modeling text, but the parameters are estimated slowly. In chapter 7 we address the issue of the slow convergence for the Dirichlet compound multinomial model. The model is approximated with an exponential model which can be estimated much faster.

- Many document categories often share latent topics among them. In chapter 8 we extend the approximated dirichlet compound multinomial model, allowing it to model shared latent topics.

Chapter 2

Pruning The Vocabulary Using Neural Network Sensitivities

2.1 Introduction

Language independent "bag-of-words" representations are surprisingly effective for text classification. Pattern recognition for text suffers much from the well-known curse of dimensionality though, since the number of input dimensions usually supersede the number of examples. This high dimensional representation is also containing many inconsistent words, possessing little or no generalizable discriminative power, and should therefore be regarded as noise. Using all the words in the vocabulary is therefore resulting in reduced generalization performance of subsequent classifiers, e.g., from ill-posed principal component transformations, using the *latent semantic indexing* dimensionality reduction procedure.

In this chapter we study the effect of reducing the least relevant and inconsistent words from the bag-of-words representation, reducing the high dimensional representation of text. Previous research has shown that big fractions of the vocabulary can be removed without penalizing the classification ability (Yang

& Pedersen, 1997). We consider a new approach using a set of neural network based sensitivity maps (Zurada et al., 1994) for determination of term relevancy, which is used for vocabulary pruning. The pruning of the vocabulary is based on the so called scaled sensitivity values. The scaled sensitivities are computed using the so-called NPAIRS split-half re-sampling procedure (Strother et al., 2002). The hypothesis is that sensitivity maps can determine which terms are consistently important, hence likely to be of general use for classification relative to terms that are of low or highly variable sensitivity. Scaled sensitivities has previously been successfully applied for dimensionality reduction of other ill-posed high dimensional pattern recognition challenges (Sigurdsson et al., 2004). With reduced vocabularies documents are classified using LSI representation and a probabilistic *artificial neural network* (ANN) classifier (Bishop, 1996).

2.2 Preprocessing

All the preprocessing steps are based on the so called bag-of-words representation, which does not take into account the sequence or order in which the words in a document appear. The bag-of-words representation considers only the amount of times each word appeared in a document, and is practically a word histogram representation. This form of representation makes lots of machine learning operations on text easy, especially dimension reduction methods like *independent component analysis* (ICA) (Bell & Sejnowski, 1995b; Bell & Sejnowski, 1995a; Molgedey & Schuster, 1994), *non-negative matrix factorization* (NMF) (Lee & Seung, 1999; Lee & Seung, 2001) and *singular value decomposition* (SVD) (Madsen et al., 2003). Using the bag-of-words representation however result in removal of semantical information that might be useful. In the following two chapters we will look deeper into the bag-of-words document representation, and how some of the semantical information can be exploited for better classification.

Using the generic bag-of-words approach for LSI documents are arranged in a document matrix $\mathbf{X}$, where each element in the matrix x_{dw} contains the number of times word w occurs in document d. The dimensionality of $\mathbf{X}$ is at first reduced by filtering and stemming. Stemming refers to a process in which words with different endings are merged, e.g., "train", "trained" and "training" are merged into the common stem "train". This example also indicates the main problem with stemming, namely that it introduces an artificial increased polysemy[1]. We have decided to "live with this problem" since without stemming vocabularies would grow prohibitively large. About 500 common

[1]Polysemy is when a word has more than one meaning.

non-discriminative stop-words, i.e. ("a", "i", "and", "an", "as", "at") has been filtered out of the document, while possessing little discriminative information. The common word filtering process reduces the document word count considerable, while only reducing the vocabulary marginally.

The term-document matrix can be normalized in various ways. In (Debole & Sebastiani, 2003) experiments with different term weighting schemes are carried out. The "term frequency-inverse document frequency" (TFIDF) weighting is consistently good among term weighting methods purposed, and is the method generally used (Huang, 2000; Sebastiani, 2002). After TFIDF normalization the resulting elements in $\mathbf{X}$ becomes,

$$x_{dw}^{\text{tfidf}} = x_{dw}^{\text{tf}} \log \frac{D}{F_w} \tag{2.1}$$

where D is the number of document in $\mathbf{X}$, W is the number of words in the vocabulary and F_w is the document frequency of word w and x_{dw}^{tf} is the log-normalized word frequency for a single document.

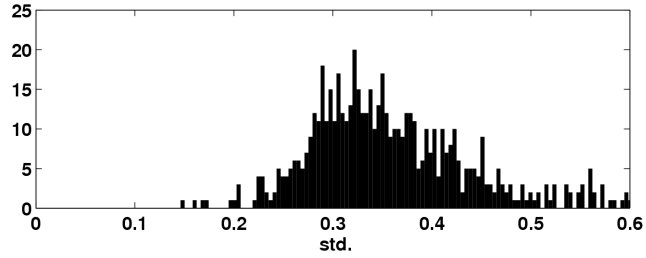
$$x_{dw}^{\text{tf}} = \begin{cases} 1 + \log(x_{dw}) & \text{if } x_{dw} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

The length of the documents is often a good prior for predicting the content within a small corpora. While document length might be a solid variable within the corpora, it not likely that it generally is a valid parameter. The length of the documents is usually normalized to prevent the influence that the document length might have. The Frobenius norm (Horn & Johnson, 1990) is used to length normalize each individual document vector to one.

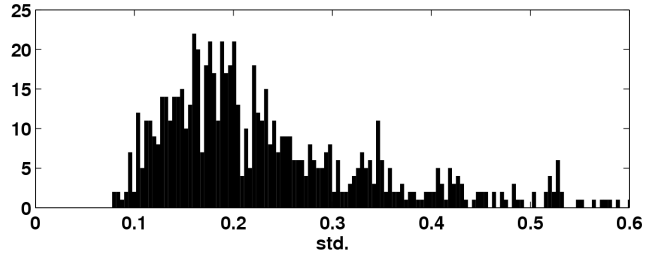
$$x_{dw}^{norm} = \frac{x_{dw}^{\text{tfidf}}}{\sqrt{\sum_{w'=1}^{W} x_{dw'}^{\text{tfidf}2}}}. \tag{2.3}$$

To emphasize the influence of document lengths, the distribution of the term standard deviations for the spam and non-spam documents, in the email data-set[2] (Nielsen, 2001), are illustrated in Figure 2.1.

[2]The email data-set contains 1400 emails, where approximately half of the emails are spam emails.



(a) non-spam



(b) spam

Figure 2.1: Distribution of the standard deviation for the email data-set. The distribution for the spam class and the non-spam class varies a lot. The standard deviation is a good discriminator, but probably not general outside this data-set. Using only the standard deviation for classification, the generalization error is 22%.

Using only the standard deviation measure for classification, 78% of the documents in a email spam data-set can be classified correctly (we here only discriminate between spam and non-spam emails). This clearly shows that document length is a good prior.

2.3 Latent Semantic Indexing

Latent semantic indexing (LSI) (Furnas et al., 1988; Deerwester et al., 1990) uses SVD to find the most varying directions in the semantic space for a set

of documents. Projecting the document vectors onto these directions of high variation, create a new low dimensional representation for the documents. LSI is furthermore believed to reduce problems synonymy[3] and polysemy (Deerwester et al., 1990; Larsen et al., 2002).

The preprocessed document matrix $\mathbf{X}_{\mathrm{p}}$ is factorized using SVD, carried out by an "economy size" SVD,

$$\mathbf{X}_{\mathrm{p}}^{\mathrm{T}} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^{T}. \tag{2.4}$$

where the orthogonal $W \times D$ matrix $\mathbf{U}$ contains the eigenvectors corresponding to the non-zero eigenvalues of the symmetric matrix $\mathbf{X}_{\mathrm{p}}^{T}\mathbf{X}_{\mathrm{p}}$. $\mathbf{\Lambda}$ is a $D \times D$ diagonal matrix of singular values ranked in decreasing order and the $D \times D$ matrix $\mathbf{V}^{T}$ contains eigenvectors of the symmetric matrix $\mathbf{X}_{\mathrm{p}}\mathbf{X}_{\mathrm{p}}^{T}$. The LSI representation of the documents $\mathbf{Z}$ is obtained by projecting document histograms on the basis vectors in $\mathbf{U}$,

$$\mathbf{Z}^{T} = \mathbf{U}^{T}\mathbf{X}_{\mathrm{p}}^{T} = \mathbf{\Lambda}\mathbf{V}^{T}. \tag{2.5}$$

Typically, the majority of the singular values are small and can be regarded as noise. Consequently, only a subset of K ($K << W$) features is retained as input to the classification algorithm which corresponds to using only the first few columns of $\mathbf{U}$ when document histograms $\mathbf{X}_{\mathrm{p}}^{T}$ are projected onto these. The representational potential of these LSI features is illustrated in Figure 2.2 and Figure 2.3,

where it is obvious that a low dimensional representation of the documents possesses much of the information needed for discrimination. After the preprocessing step, the data-set had a vocabulary of approximately 10.000 words, which can be reduced to about 100 principal directions.

2.4 Neural Network Sensitivities

The vocabulary pruning is based on a neural network sensitivity analysis that measures how much the neural network rely on each input. We use a two-layer feed-forward neural network structure with K inputs, where the estimated output $\tilde{y}_{dc}$ for class c is given by

[3]Synonymy is when multiple words have the same meaning.

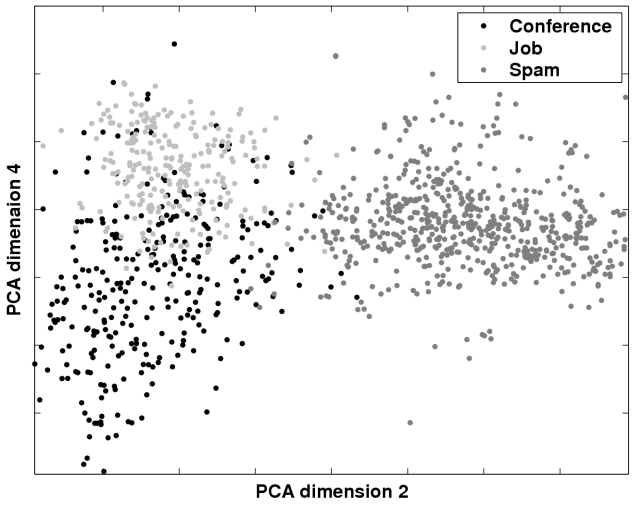


Figure 2.2: Illustration of the document distribution in the feature space. Here we show the Email corpus projected onto the 2nd and 4th principal directions. In this projection the "spam" class is well separated while the two other classes in the set ("conferences" and "jobs") show some overlap.

$$\tilde{y}_{dc} = \sum_{h=1}^{H} w_{ch} \tanh \left(\sum_{k=1}^{K} w_{hk} f_{dk} + w_{h0} \right) + w_{c0} \tag{2.6}$$

where w_{hk} is the input to hidden weights, w_{h0} is the input bias, w_{ch} is the hidden to output weights and w_{c0} is the hidden bias. The constant H is the number of hidden units and $\tilde{y}_{dc}$ is the outputs of the network. The network outputs are normalized using the softmax (Bridle, 1990) giving an estimate of the posterior probabilities,

$$\hat{P}(y_{dc} = 1|x_d) = softmax \left(\sum_{h=1}^{H} w_{ch} \tanh \left(\sum_{k=1}^{K} w_{hk} f_{dk} + w_{h0} \right) \right) + w_{c0} \tag{2.7}$$

where $\hat{P}(y_{dc} = 1|x_d)$ is an estimate of the probability that the document x_d belongs to class c, and y_d is a vector where element c is one if the document belongs to class c.

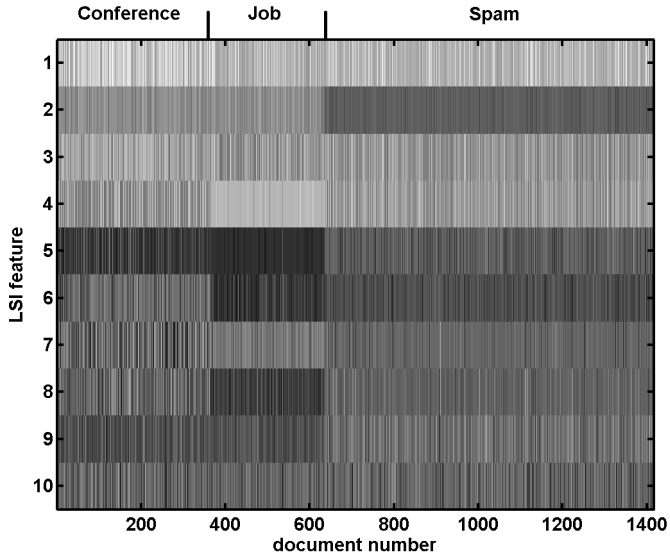


Figure 2.3: Illustration of the first 10 LSI features for the email data-set. The second feature is useful when discriminating the spam emails from the two other categories.

The sensitivities used here are the *absolute value average sensitivities* (Zurada et al., 1994) for class c. The sensitivities are the derivatives of the estimated posterior $\hat{P}(y_{dc} = 1|\mathbf{f}_d)$ of each class with respect to the inputs before the LSI projection.

$$\mathbf{s}_c = \frac{1}{D}\sum_{d=1}^{D}\left|\frac{d\hat{P}(y_{dc} = 1|\mathbf{f}_d)}{d\mathbf{x}_d}\right| \tag{2.8}$$

where $\mathbf{f}_d$ is the latent[4] document representation for document d and $\mathbf{s}_c$ is a vector of length W with the sensitivities for each of the words for class c. It is necessary to sum the absolute sensitivity values in equation 2.8 to avoid mutual cancellation of positive and negative sensitivities. The numerical calculations of the sensitivities can be found in (Sigurdsson et al., 2004).

The sensitivity vectors $\mathbf{s}_c$ are finally normalized to unit length, to ensure that

[4]we call the LSI representation, a "latent" representation, while the LSI finds a latent subspace within the space of the whole vocabulary

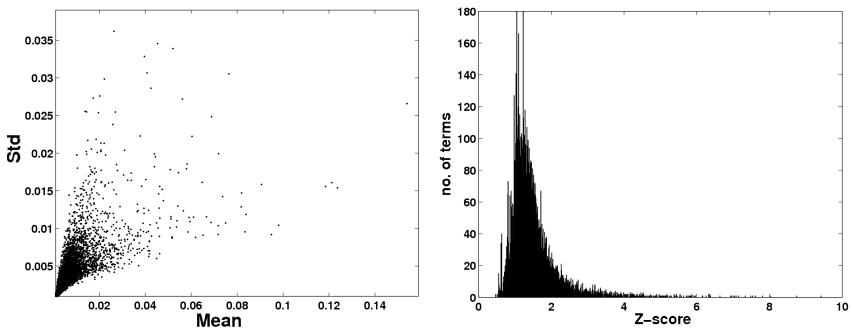
the non-uniqueness of the hidden-to-output weights does not result in different sensitivities.

$$\tilde{\mathbf{s}}_c = \frac{\tilde{\mathbf{s}}_c}{||\tilde{\mathbf{s}}_c||} \tag{2.9}$$

The reproducibility of the sensitivities for different data splits is essential when determining feature importance. Large sensitivities values that varies a lot from split to split are likely to be noise, from words used inconsistently in the corpora. Smaller sensitivity values that are reproducible are more likely to come from words with consistent discriminative power.

A split-half re-sampling procedure is invoked to determine the statistical significance of the sensitivity (Strother et al., 2002). Multiple splits are generated from the original training set and classifiers trained on each of the splits. For each classifier a sensitivity map is computed. Since the two maps obtained from a given split are exchangeable the mean map is an unbiased estimate of the 'true' sensitivity map, while the squared difference is a noisy, but unbiased estimate of the variance of the sensitivity map. By repeated re-sampling and averaging the sensitivity map and its variance are estimated. The vocabulary pruning is based on each individual words Z-score, where the Z-score is defined as the mean sensitivity divided by the sensitivity standard deviation $Z_w = \mu_w/\sigma_w$. Using Z-scores as indication of feature importance has previously been a robust measure in other applications (Sigurdsson et al., 2004). In Figure 2.4 the histogram of Z-scores for the words in the email vocabulary are shown. Only few of the words in the vocabulary have a high Z-score, indicating that many of the words can be regarded as being noise. Intensive pruning is therefore likely to make text classification better.

A wide variety of classification algorithms have been applied to the text categorization problem, see e.g., (Kosala & Blockeel, 2000). We have extensive experience with probabilistic neural network classifiers and a well tested ANN toolbox is available (Sigurdsson, 2002). Document classification approaches based on neural networks have a record of being successful (Sebastiani, 2002). The ANN toolbox adapts the network weights and tunes complexity by adaptive regularization and outlier detection using the Bayesian ML-II framework, hence, requires minimal user intervention (Sigurdsson et al., 2002). Pruning the vocabulary based neural network sensitivities, further makes classification with neural networks an obvious choice.



(a) Sensitivities (b) Z-scores

Figure 2.4: Standard deviation, mean and Z-scores for the sensitivities of the words in the email corpora. The values are calculated using 10 split-half re-sampled data sets. Few words have a high Z-score, while most of the words have a rather small Z-score.

2.5 Experiments

For the experiments we consider two data-sets, "Email" (Nielsen, 2001) and "WebKB" (CMU-WebKB, 1997) whom are used to illustrate and test the hypothesis. No less than ten split-half re-samples are used in all experiments. The Email data-set consists of texts from 1431 emails in three categories: conference (370), job (272) and spam (789). The WebKB set contains 8282 web-pages from US university computer science departments. Here we have used a subset (CMU-WebKB-2240, 1999) of 2240 pages from the WebKB earlier used in (Larsen et al., 2002). The WebKB categories are: project (353), faculty (483), course (553) and student (851). All html tags were removed from the data-set.

Preliminary experiments indicated that a reduced feature space of $K = 50$ projections and a neural network classifier with five hidden units is sufficient for the task. All results have been validated using 10 fold split half re-sampling cross validation. The neural network based term sensitivity is a function of the given training set. Terms for which the sensitivity is high but also highly variable are less likely to support generalizability compared to terms that have a consistent high or medium sensitivity. The empirical distribution of mean and standard deviations the terms sensitivities of the Email set are shown in Figure 2.4(a), where the consistently important words are those closest to the lower right corner. The Z-score measure is high for these words, and the pruning is

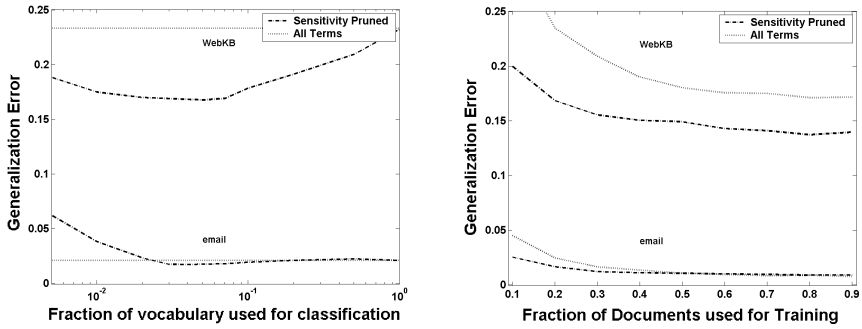
solely based on this measure.

Based on the scaled sensitivities, relevant keywords for the text categories have been extracted. For the Email data the five highest scores for the Conference category are (*Paper, Conference, Deadline, Neural, Topic*) and for the Job category (*Research, Position, Candidate, University, Edt*) and for the Spam category (*Money, Remove, Free, Thousand, Simply*). All these words are meaningful for the three categories, which is an indication of the validity of the sensitivity Z-score measure.

We start by classifying the documents in the two corpora using all the words that remain after the preprocessing, that is about 10.000 words. Using all the words, the generalization error rate is 23.3% in the WebKB and 2.1% in email data, when 20% of the documents are used for training the model. Removing respectively 97% and 95% of the vocabularies with the lowest sensitivity Z-score, the generalization error for the WebKB is reduced to 16.5% and to 1.5% for the Email data. Though the classification enhancement for the email data is modest, the relative generalization error is then reduced with 29% and 28% respectively for the two data-sets. The generalization error is generally lowered within a wide area of pruning fractions, see Figure 2.5(a).

We investigate the effectiveness of the sensitivity pruning from different training set sizes, by generating *learning curves* using full and pruned vocabularies. The learning curves are shown in Figure 2.5(b).The learning curves shows decreased generalization error for a range of training set sizes. For the WebKB, the pruning method shows consistently reduced generalization error of about 25% for the whole range of training set sizes. For the email data the pruning decreases the generalization error when using less than 40% of the data-set for training. When 40% or more of the data-set samples are used for training, the generalization error is not reduced further. Noise within the data might prevent any classification algorithm from further optimizing the generalization error for the email data.

We finally generate confusion matrices with the generalization error for each individual class. The rows in a confusion matrix show the true class information of the documents, while the columns marked with * show the estimated class information. The confusion matrices, when using the full vocabulary, are shown in Table 2.5. Comparing with the values of Table 2.5, the generalization error using the pruning approach is not only better but also more balanced, i.e. the values in the diagonal are more similar, and the column sums are closer to one.



(a) Pruning (b) Learning Curves

Figure 2.5: Generalization error using different pruning fraction (a) and learning curves with the optimal pruning settings (b). The best classification results are obtained when the vocabulary is reduced with 97% for the email data and 95% for the WebKB (a). The generalization error is then reduced with 29% and 28% respectively. Keeping the vocabulary reduction factor fixed at 95%, the learning curves show that the pruning has a positive effect over the whole range of training-set sizes.

	Conf*	**Job***	**Spam***
Conf	**.973**	**.016**	**.011**
Job	**.014**	**.969**	**.017**
Spam	**.006**	**.009**	**.985**

	Prj*	**Fac***	**Cou***	**Stu***
Prj	**.706**	**.114**	**.061**	**.119**
Fac	**.090**	**.613**	**.046**	**.251**
Cou	**.060**	**.065**	**.857**	**.018**
Stu	**.023**	**.130**	**.026**	**.821**

Table 2.1: Confusion matrix for email data and the WebKB, using the full vocabulary and 20% of the data for training.

2.6 Discussion

The results determined in the experiment section using the whole vocabulary, are similar to those classification result found in (Larsen et al., 2002). This makes the rest of the experimental results reliable.

The vocabulary pruning shows consistent enhancement of the generalization error for a range of different vocabulary fractions. This is a clear indication that many words are not carrying a sufficient amount of information, making

	Conf*	Job*	Spam*
Conf	**.983**	**.013**	**.004**
Job	**.012**	**.981**	**.007**
Spam	**.005**	**.008**	**.987**

	Prj*	Fac*	Cou*	Stu*
Prj	**.811**	**.077**	**.033**	**.079**
Fac	**.059**	**.758**	**.023**	**.160**
Cou	**.044**	**.052**	**.891**	**.013**
Stu	**.017**	**.109**	**.021**	**.853**

Table 2.2: Confusion matrix for email data and the WebKB, using the 5% of the vocabulary with the highest sensitivity Z-score. This approach produces more balanced confusion matrices.

them un-useful for classification of the documents. Surprisingly many of the words carry little information (the optimal pruning fraction is 95%) and should therefore not be considered valuable for the classifier. We can however not be certain, that the word ranking method considered here is the most optimal approach for the task. Other better ranking methods might suggest to use a greater or perhaps a smaller fraction of the vocabulary. Though better methods for ranking the words might exist, the sensitivity Z-score ranking has proved itself to some extend, by ranking relevant class keywords high. The sensitivity Z-score pruning method has also been shown to be relevant by being consistent over different training set sizes, though a bound was reached for the email data.

The enhancements that the pruning approach has caused on the generalization error, have also resulted in more balanced confusion matrices. This shows that probability mass in the classifier is not just moved to the bigger classes, gaining a better classification accuracy that way. The classification enhancement is actually gained by having a better model.

The downside of the sensitivity Z-score based pruning approach is that it is extremely slow. While the Z-scores are based on many re-samplings and repeatedly re-training of the neural networks, the process of estimating the Z-scores is very time-consuming. Another way of estimating the word relevances should therefore be considered, where simple linear correlation methods as canonical correlation analysis (Mardia et al., 1979) might be a usable approach.

2.7 Summary

Neural network sensitivities were introduced in a LSI based context recognition framework. The scaled sensitivities were normalized to a Z-score for each word in the corpora. Based on these Z-scores the vocabularies were pruned considerably resulting in consistently lower generalization errors when new documents were to

be classified. The experiments were carried out on two mid-size data-sets and showed consistency in enhancing the relative classification by approximately 25% over a range of training set-sizes. The results suggest that effective ways of determining the discriminative words can reduce the dimensionality of text pattern recognition systems considerably.

Chapter 3

Fusion With Natural Language Features

3.1 Introduction

The document vector space model (Salton et al., 1975), the bag-of-words model and its varieties are effective document simplifications, that make machine learning approaches to text modeling and classification simple. The two document representations has resulted in the development of many different algorithms (Deerwester et al., 1990; Hofmann, 1999; Sebastiani, 2002; Blei et al., 2003) who are effective for text classification. The models that use these representations however loose a big fraction of the information contained in the documents, by considering only the counts of how many times words appear in a given document. The other part of information contained in documents is the information about the order in which the words appear. Though the major part of document information is contained in the knowledge about which word occur, some important information might be captured from the word appearance order, that could make document classification accuracy better. We here consider the use of natural language processing (NLP) features, for capturing parts of the word order information.

Researchers have previously used NLP features to enhance classification accu-

racy in a number of studies. One example is the use of the so-called WordNet[1] (wor,) system. The WordNet synonymy features were used to expand term-lists for each text category (De Buenaga Rodríguez et al., 1997). This strategy enhanced the accuracy of the text classifier significantly. Limited improvements were obtained by invoking semantic features from WordNet's lexical database (Kehagias et al., 2003). In (Basili et al., 2001) and (Basili & Moschitti, 2001) enhanced classification ability was reported by the use of POS-tagged terms, hereby avoiding the confusion from polysemy. In (Aizawa, 2001) a POS-tagger was used to extract more than $3 \cdot 10^6$ compound terms in a database. A classifier based on the extended term list showed improved classification rates.

It is easy to extract the word appearance information from a document and form it into some meaningful representation that can be used for machine learning, i.e. vectors or histograms. The word order information is however harder to extract to some simple low dimensional representation, which is easily portable to a machine learning algorithms. The word order information is also likely to be valueless if considered alone at first, for later fusion with the probabilities from the vector space model. Instead of using the word order information directly, parts of the information can be captured in some other form. We here consider word tag features estimated by a so-called part-of-speech tagger, that is a tag value for each word in the document. The tag value is describing the associated word's grammatical status in the context, i.e. the word's part of speech.

In this Chapter our aim is to elucidate the synergy between these so called part-of-speech tag features and the standard bag-of-words language model features. The feature sets are combined in an early fusion design with an optimized fusion coefficient that allows weighting of the relative variance contributions of the participating feature sets. With the combined features documents are classified using the LSI representation and a probabilistic neural network classifier similar to the one used in chapter 2 section 2.3 on page 16.

3.2 Part-of-speech Tagging

A part-of-speech (POS) tagger (Manning & Schütze, 1999) is an algorithm that reads text and for each token in the text returns the text-tokens part-of-speech, e.g. noun, verb or punctuation. A POS tagger therefore converts a strings of text-tokens into strings of POS tag tokens of similar length. Most POS taggers

[1] WordNet is an online lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. The package also contains a set of relational links for the synonym sets.

are based on statistical methods, like e.g. hidden Markov models, and trained on large corpora with examples of texts annotated with POS tags. An example of a large corpus is the Penn treebank (Marcus et al., 1994), a corpus consisting of more than 4.5 million words of American English text with annotated tags.

We here consider a probabilistic POS tagger, QTAG (Mason, 2003; Tufis & Mason, 1998), which uses the commonly used Brown/Penn-style tag-set, shown in Table 3.2.

Most statistical taggers of today are fairly robust, tagging approximately 97% (Schmid, 1994) of the words in a text with the correct tag, depending on the tag-set used. We have chosen a rather small tag-set (Brown/Penn), which uses "only" 70 different tags, see Table 3.2. The taggers with smaller tag-sets have slightly better accuracy than the taggers with larger tag-sets. The taggers with smaller tag-sets, are therefore more likely to generate tags that are more generalizeable.

The QTAG tagger used here is among the best taggers when it comes to high accuracy, which is an important feature. The high accuracy means that the tags can be regarded as a true un-noisy feature. In Table 3.2 the QTAG has been used to extract the POS tags of a simple sentence.

The representation we here use for the POS-tags tokens is similar to the one used for normal text tokens in chapter 2. The POS-tag information is therefore captured in a bag-of-POS-tags, where the ordering information is discarded. The remaining information is therefore at histogram of POS-tag tokens for each document. Histograms of POS-tags can be interpreted as the authors style of writing, i.e. a fingerprint that tells how the author constructs his sentences. Some authors might construct grammatically different sentences from others. This grammatical difference might not be captured when only word histograms are considered. An example how a sentence could be constructed with basically the same meaning but different writing style is shown in Table 3.2.

The two sentences in Table 3.2 are basically the same when looking at their word histograms after stemming and stop-word removal. This is also expected since they carry the same meaning. The difference in writing style however could be important for some applications, like author detection tasks and spam detection filters[2] (Androutsopoulos et al., 2000; Sakkis et al., 2001). The use of nonsense sentences could be detected from the POS-tag histograms. The writing style might also be an important feature when classifying documents,

[2]Some spam emails have a lots of information carrying words attached to the button of the email to suppress the fraction of spam related words within the email. These words are just concatenated in some nonsense way. This suppression technique confuses some spam filters, making spam emails penetrate the filters.

POS	description	POS	description
BE	be	PN	pronoun, indefinite
BEDR	were	POS	possessive particle
BEDZ	was	PP	pronoun, personal
BEG	being	PP$	pronoun, possessive
BEM	am	PPX	pronoun, reflexive
BEN	been	RB	adverb, general
BER	are	RBR	adverb, comparative
BEZ	is	RBS	adverb, superlative
CC	conjunction, coordinating	RP	adverbial particle
CD	number, cardinal	SYM	symbol or formula
CS	conjunction, subordinating	TO	infinitive marker
DO	do	UH	interjection
DOD	did	VB	verb, base
DOG	doing	VBD	verb, past tense
DON	done	VBG	verb, -ing
DOZ	does	VBN	verb, past participle
DT	determiner, general	WBZ	verb, -s
EX	existential there	WDT	det, wh-
FW	foreign word	WP	pronoun,
HV	have	WP$	pronoun, possessive
HVD	had	WRB	adv, wh-
HVG	having	XNOT	negative marker
HVN	had	!	exclamation mark
HVZ	has	”	quotation mark
IN	preposition	’	apostrophe
JJ	adjective, general	(	parenthesis begin
JJR	adjective, comparative	)	parenthesis end
JJS	adjective, superlative	,	comma
MD	modal auxiliary	-	dash
NN	noun, common singular	.	point
NNS	noun, common plural	...	...
NP	noun, proper singular	:	colon
NPS	noun, proper plural	;	semi-colon
OD	number, ordinal	?	question mark
PDT	determiner, pre-	???	undefined

Table 3.1: POS tags used by the Q-TAG part-of-speech tagger. The tag-set is variant of the common Brown/Penn-style tag-sets, and has generally been used for tagger evaluation.

while some document classes usually have a special group of authors associated with it. These authors unconsciously may agree on a specific writing style.

Another situation where important information is removed during the conver-

Tag-set	number of tags
Brown	**170**
Brown/Penn	**70**
CLAWS1	**132**
CLAWS2	**166**
CLAWS5	**65**
London-Lund	**197**
Penn	**45**

Table 3.2: The sizes of different POS-tag sets, differ greatly in the number of distinctions they make. The tag-set used here is the Brown/Penn tag-set.

The	**mechanic**	**put**	**the**	**hammer**	**on**	**the**	**table**
DT	**NN**	**VB**	**DT**	**NN**	**IN**	**DT**	**NN**

Table 3.3: Example of a tagged sentence.

The	**prisoner**	**has**	**inmates**	**who**	**behaves**	**badly**
DT	**NN**	**HVZ**	**NNS**	**WP**	**VBZ**	**RB**
,	**so**	**he**	**feels**	**frustration**	**.**	
,	**VBN**	**PP**	**VBZ**	**NN**	**.**	
The	**prisoner**	**feels**	**frustrated**	**with**	**his**	**badly**
DT	**NN**	**VBZ**	**VBN**	**IN**	**PP$**	**RB**
behaved	**inmates**	**.**				
VBN	**NNS**	**.**				

Table 3.4: Two sentences with similar meaning, written with two different writing styles. The first sentence is constructed in a simpler manner than the second one, which let the words flow more easily.

sion from text to word histograms, is when the same word can have more meanings (polysemy). In Table 3.2 are two sentences with different meaning but almost same word usage after stemming and stop-word filtering. The differences in the two sentences are again captured by the POS-tag histograms.

We notice that we might discard valuable information by disregarding the order in which the POS-tags appear, by considering POS-tag histograms instead of sequences. A hidden Markov model might capture more information from the sequences of POS-tags, than the LSI model can capture from the histograms of POS-tags. The fusion of the text and POS features however becomes much simpler when using the histogram representation. The histogram representation

I	usually	want	to	train	late	in
NN	RB	VB	IN	VB	JJ	IN
the	night	with	the	others	.	
DT	NN	IN	DT	NNS	.	
I	was	later	for	the	train	the
NN	BEDZ	RBR	CS	DT	NN	DT
other	night	than	usually	.		
JJ	NN	CS	RB	.		

Table 3.5: Two sentences with different meaning, written with use of almost identical words. After stemming and stop-word removal, the word usage is the same.

will therefore be used in the following sections.

3.3 Feature Fusion

Feature fusion is the process of combining different feature-sets into one set of features that can be used for classification. This is in analogy with the ability of the human brain to combine multiple inputs for enhanced pattern recognition capability. As mentioned earlier the two feature sets to be fused are both on same form, that is histograms of counts. The fusion can be achieved at different levels using either early fusion or late fusion.

Early fusion is when the features are combined before making the LSI transformation on the combined set of word tokens and POS tokens. Using early fusion, the histograms for the two feature sets are simply concatenated, resulting in a larger histogram. The LSI transformation will then find the low dimensional representation, that will contain the combined paradigms of the two feature sets.

When late fusion is considered, the two feature-sets are combined after the LSI subspace approximation. We are then left with two feature-sets to feed to the classifier algorithm. Using this fusion technique we might end up with features following the exact same paradigm in each of the two sets of features. Late fusion might result in an even worse incident, that is that the paradigms covered in much noise might not be captured in any of the feature subspace projections. The full synergy effect from the two feature-sets might therefore not be used, while the features are separated. We therefore choose to use the early fusion scheme.

Early fusion, of feature sets with different statistics, based on variance decomposition requires determination of the relative weights of the participating feature sets. One possibility would be to use variance decomposition based on factor analysis which is insensitive to relative scaling of variables. For simplicity, we have chosen to introduce a single fusion coefficient α which can be tuned for each corpus separately.

$$\mathbf{X} = \begin{bmatrix} \alpha \mathbf{X}_{words} \\ (1-\alpha)\mathbf{X}_{tags} \end{bmatrix} \tag{3.1}$$

The histograms of word counts are contained in $\mathbf{X}_{words}$ and the histograms with POS-tags in $\mathbf{X}_{tags}$. If the fusion parameter is close to zero $\alpha \approx 0$, the variance is dominated by tag features while when close to one $\alpha \approx 1$, term features dominate. The fusion coefficient is determined from the training-set using cross validation re-sampling.

3.4 Experiments

For the experiments we consider the same two data-sets, "Email" and "WebKB", that was considered in chapter 2. A third data-set "multimedia" (Kolenda et al., 2002; Kolenda, 2002) will also be used for the experiments. The multimedia corpus consists of texts and images from 1200 web pages, where only the text part is considered here. The categories in the multimedia corpus are: Sports (400), aviation (400) and paintball (400).

The preprocessing, LSI subspace projection and neural network classification set-up is similar to the approach used in chapter 2, except for the vocabulary pruning based on sensitivities which is omitted here.

We performed at first two kinds of experiments, first using the POS-tags alone and secondly using the word counts alone. We split the corpora in 20% for training and 80% for testing (the role of the split ratio is discussed below). The POS-tags features alone are surprisingly potent: We found that 89.7% of the email data-set is classified correctly using the 70 POS-tag features. This should be compared to 96.6% classification accuracy obtained using the word count features. For the multimedia data, using the POS-tag and term features separately resulted in accuracies of 74.6% and 94.2% respectively. The WebKB data is somewhat harder to classify. Here the POS-tag and term features lead to accuracies of 57.2% and 76.1% respectively.

The potential synergy of words and POS-tags is illustrated in Figure 3.1. The figure shows the performance correlation between the classifiers trained on the individual feature sets. The bars labelled "independent" indicate the performance of the classifiers where the feature-sets are assumed independent. These data are obtained from their basic performance and assuming independence of their decisions. In bars labelled "real" we show the actually observed rates. Note that there is a high potential synergy, since the observed performances are close to those predicted by independence.

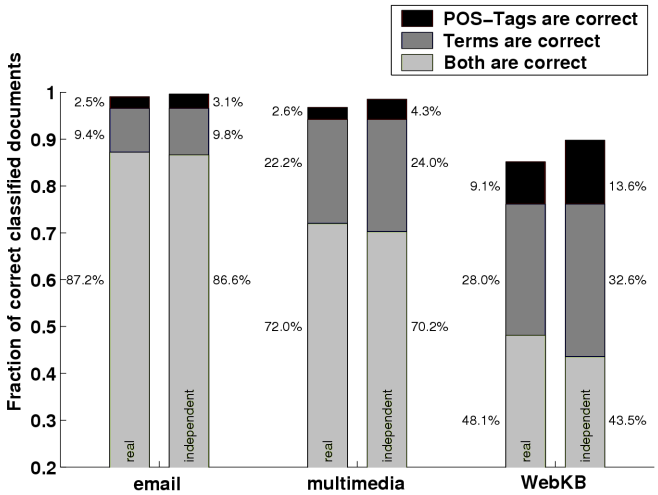
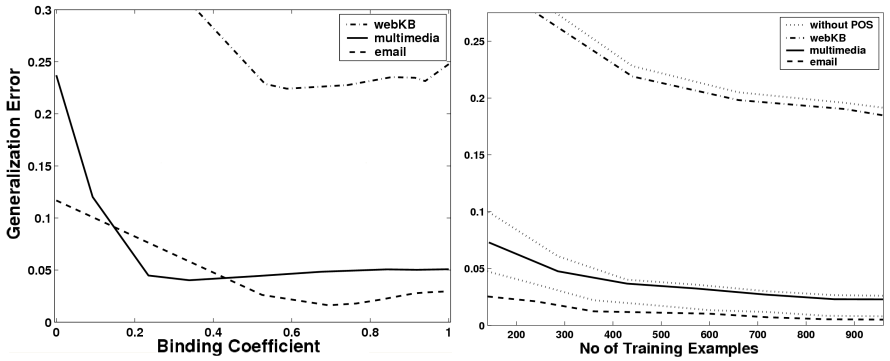


Figure 3.1: Fraction of correct classified documents for the POS-tag and term representations. The bars labelled 'real' indicate observed rates of events where the two feature sets lead to correct decision and one correct/one incorrect respectively. This is compared with rates estimated from assumed independence of errors (bars labelled 'independent'). The figure indicates that the errors made by classifiers based on POS-tags and the term features sets are relatively independent, hence, that there is a potential synergy to be gained from fusion of the feature sets.

We next turn the attention to the combined feature set. In Figure 3.2(a) we illustrate the role of the fusion coefficient α, c.f., (3.1). The classification test set error rates (an unbiased estimate of the generalization error defined as the probability of misclassification of a random test datum) were obtained by ten-fold cross-validation. We observed significant synergy: The performance of the term features ($\alpha = 1$) is indeed improved by adding POS-tag feature information. The effect is relatively high for the email data-set (reducing the error by almost 30%), while the effect is smaller for the harder WebKB set (the error is reduced by about 8%).



(a) Fusion (b) Learning Curves

Figure 3.2: (a) Generalization error obtained by fusion of POS-tags and word counts with a variable fusion coefficient. $\alpha = \mathbf{1}$ corresponds to POS-tag features only. Optimal fusion results in reduction of the error rate by 30%, 22% and 8% in the email, multimedia and WebKB corpora respectively Results obtained by ten-fold cross-validation using a 20/80 train/test set split ratio. (b) To the right we show learning curves with and without POS-tag fusion. The fusion of natural language and conventional term features improves performance for all the training set sizes investigated.

	Conf*	Job*	Spam*
Conf	**.974**	**.017**	**.009**
Job	**.015**	**.968**	**.017**
Spam	**.007**	**.008**	**.985**

	Conf*	Job*	Spam*
Conf	**.979**	**.015**	**.006**
Job	**.017**	**.977**	**.006**
Spam	**.004**	**.006**	**.990**

Table 3.6: Confusion matrix for email data using the full vocabulary and 20% of the data for training. The table to the left is when conventional term features are used alone. The table to the right is for fusion of conventional term features and natural language features. The use of natural language features has an extended positive influence on the ability to discriminate spam emails.

The synergistic advantage is likely to depend on the size of the database. For further investigation of this, we have estimated "learning curves". The results are provided in figure 3.2(b). In these ten-fold cross-validation experiments we used the 'optimal' fusion coefficients found in figure 3.2(a). In these relatively limited data sets there is a positive, albeit diminishing, synergy to be obtained for all training set sizes.

We suggested earlier that the POS-tags might be valuable for detecting spam emails, while spam emails often has a special writing style. In Table 3.6 the confusion matrix for email data-set is shown when using conventional term features alone and when they are fused with natural language features. The use of natural language features has especially made the classifier able to discriminate between spam and non-spam emails. Some of the POS-tag features in Figure 3.3 show that spam emails are easy to discriminate from the two other categories.

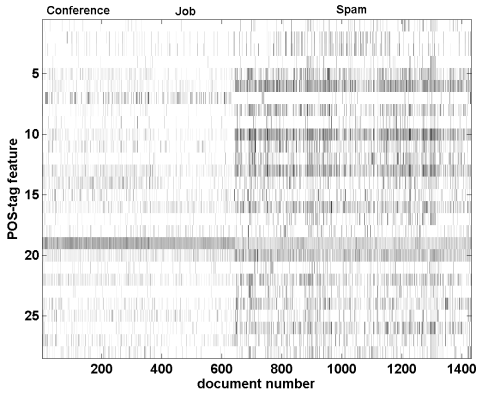


Figure 3.3: Some of the most valuable POS-tag features, for discriminating spam emails from the two other categories. The first 642 emails are conference and job emails, and the remaining 789 are spam emails.

3.5 Discussion

Classification of documents based on POS-tag features has shown to be surprisingly effective. The 70 features actually captures so much information about the documents, that using them alone provides a decent classifier.

The fusion of conventional text features and POS-tag features shows consistent enhancement of the generalization error for a range of different vocabulary fractions. The enhancement is most distinct when only few documents are available for training the model. When a sufficient amount of training examples are present the model is probably capturing enough information from the conventional text features, and the natural language features are therefore not useful any longer.

The POS-tag features have especially been valuable when the classifier needs to

discriminate between classes of documents where the author's writing style is a distinct factor. This is true for the email-data set, and the use of POS-tags has therefore had an extended positive influence on the ability to discriminate spam emails.

3.6 Summary

Natural language features in the form of part-of-speech (POS) tags were introduced to supplement conventional term features. The features were combined in an early fusion scheme, making it possible to use neural network classifier on a LSI subspace projection of the combined features. The addition of natural language features has resulted in consistency in enhancing the relative classification over a range of training-set sizes, where the approach has been most impressive when only few training examples are available. Though valid for three data-sets, the combined features have especially increased the ability to discriminate spam emails.

Chapter 4

State Space Models

4.1 Introduction

In the previous chapter we addressed the problem of the bag-of-words document representation not capturing the information from the order in which the words appear. The some of the information contained in the order in which the words appear, was captured by natural language features. We here take a different approach to make the word order information useful for document classification. The model we consider here is acting on both parts of information at the same time, that is the information about what words appear and in what order they appear.

State-space models have the ability to capture information from the order in which the words appear, and combine it with the word appearance probabilities. The state-space models should therefore conceptually super-seed vector-space models in ability to model documents correctly. State space models have previously been used for language modeling, e.g. in context of predicting the next word in handwritten text recognition systems (Zimmermann & Bunke, 2004), and has been successful so. It is therefore further likely that the state-space model can capture valuable information that can be used for text classification.

We here consider two state-space based approaches, both based on an underlying

Markov state space model. Both approaches suggest a method to overcome the dimensionality problem of text, which otherwise makes the state-space models extremely slow. The first approach suggested here generates a new lower dimensional vocabulary, which is later used in a hidden Markov model. Using the second approach, the state part of a hidden Markov model is used in conjunction with LSI emission probabilities.

4.2 Discrete Markov Process

The discrete Markov process (Rabiner & Juang, 1986) is a state space model that can model and generate sequences of discrete symbols. The discrete Markov process considers a system with K states s_k, where for each time-step t the process changes state, where the new state can be the same as the previous state. The actual state at time t is denoted q_t, which can be interpreted as the discrete symbol generated at time t. The probability of changing state to a new state $q_{t+1} = s_j$ from the state $q_t = s_i$ is determined by the transition probabilities $a_{s_i,s_j} = P(q_{t+1} = s_j | q_t = s_i)$, where $\sum_{j=1}^{K} a_{s_i,s_j} = 1$ and $a_{s_i,s_j} \geq 0$. The transition probabilities are therefore only dependent on the current state of the process and not the time t or previous states $q_{t-t'}$. A tutorial on Markov processes can be found in (Rabiner, 1989).

The discrete Markov process assembles an urn scheme where there is one urn for each state in the Markov process. When the time-step changes, a new urn is selected according to the transition probabilities, and a ball from that urn is drawn, and the color noted, whereafter the ball is returned into the urn. Each urn contains only balls with the same color.

The urn model analogy to text modeling is straight forward. Instead of balls, each urn is filled with words, again only one kind of words for each urn. When a document is generated, we start out with one particular urn and draw a word from it, and continue to another urn and draw a new word here. The transition probabilities determine what words are likely to appear after the present one. The Markov process will therefore be able to model parts of the semantics of the language model, by the transition probabilities. These semantics are not modeled at all when only word appearances alone are considered, i.e. using the vector space model representation.

Different kinds of documents might contain the same kinds of words, where the order of the appearances of the words, can change the meaning of the content. The word “train” could for example be used in documents about transportation or in documents about exercising in the gym. The words appearing around the

word train, will therefore change the meaning of that particular word. The difference in meaning could therefore be captured by the Markov model. Another example of when the order of the words appearances can change the meaning of a sentence, is when the word "not" is used. Yet another example where transition probabilities could be useful is in spam email detection systems. This was discussed further in the previous chapter.

The drawback of the Markov model is that it models a huge probability space, since it considers all the possible word-pairs in the vocabulary. Since most document collection vocabularies consider about $100,000$ words, the model must consider $10,000,000,000$ possible transition probabilities. The transition probabilities would therefore consume to much memory for holding this data representation. By use of a grammar, many of the transition probabilities could be pruned away, while many word pairs can't be used in grammatically correct sentences. Though the pruning approach would reduce the amount of modeled probabilities tremendously, the amount of memory used to represent the model would still be very large. On top of the memory consumption, the model would also need a lot of data to be able to estimate all the transition probabilities. For existing document collections, the amount of data is far too limited to estimate the probabilities, making a huge need for smoothing, which usually result in bad modeling performance. Human brains can probably work with some variety of this modeling approach, while we can generalize many probabilities in the model by use of grammar and can therefore easily prune away the unlikely Markov model transition probabilities.

4.3 Hidden Markov Model

The hidden Markov model (HMM) (Rabiner & Juang, 1986; Rabiner, 1989) extends the discrete Markov process by adding an additional emission parameter to each state. The emission parameters control the output that is generated from each state, i.e. a discrete symbol. For the HMM, each state therefore has the potential to generate all the symbols in the vocabulary of symbols. For each time-step t the HMM still changes state according to the transition probabilities a_{s_i,s_j}, but the symbol is now generated using the emission probabilities $b_{s_j,v_m} = P(x_t = v_m|q_t = s_j)$, where x_t is the symbol generated at time t and v_m is symbol number m from the vocabulary of M symbols.

The HMM assembles an urn scheme that is similar to the Markov process urn scheme. A new urn is still selected at each time-step according to the transition probabilities, and a ball from the new urn is drawn. The color of the ball is noted whereafter the ball is returned into the urn. Using the HMM each urn

now contains a distribution of balls that each has one of M different colors.

Since the number of symbols that can be generated M is independent of the number of states K, the memory consumption of the model can be reduced remarkably when the vocabulary is huge. If we consider a vocabulary of about $100,000$ words and use a state-space of 100 states, the amount of probabilities used to describe the model is approximately $10,000,000$, which is only $1/1000$ of the amount of memory needed to describe the Markov process for the same vocabulary.

The HMM parameters can be estimated using the expectation maximization (EM) algorithm (Dempster et al., 1977), resulting in an iterative update procedure that estimates the model parameters using the so called forward-backward approach (Rabiner, 1989),

$$\pi_{s_i} = \gamma_{1,s_i} \tag{4.1}$$

$$a_{s_i,s_j} = \frac{\sum_{t=1}^{T-1} \xi_{t,s_i,s_j}}{\sum_{t=1}^{T-1} \gamma_{t,s_i}} \tag{4.2}$$

$$b_{s_j,v_m} = \frac{\sum_{t=1}^{T} (O_t = v_m) \xi_{t,s_i,s_j}}{\sum_{t=1}^{T} \gamma_{t,s_j}} \tag{4.3}$$

where π_{s_i} is the probability of starting in state s_i and $\gamma_{t,s_i} = \sum_{j=1}^{K} \xi_{t,s_i,s_j}$ and $\xi_{t,i,j}$ is the probability of being in state s_i at time t and in state s_j at time $t+1$ and $(O_t = v_m)$ is 1 if the observation at time t is symbol v_m, and zero otherwise. The full description of the learning rules can be found in (Rabiner, 1989).

4.4 HMM with LSI GMM Vocabulary

The HMM approach reduces the memory needs, comparing it with a Markov process with a similar vocabulary size, making it possible to represent the model in a standard computer of today. The HMM model is however still fairly large and the EM updates that estimate the parameters are very demanding, computationally. In the approach described here, the vocabulary is therefore projected to a lower dimensional representation using latent semantic indexing (LSI) (Deerwester et al., 1990) with a SVD basis (Madsen et al., 2003) and gaussian mixture models (GMM). In Figure 4.1, the lower dimensional representation of the vocabulary is shown.

The procedure of transforming the vocabulary to a lower dimensional representation, takes place in the following way:

1. Documents are cut into substrings of length L, with 50% overlap.
2. A common LSI representation for the substrings in all the documents is estimated using SVD.
3. The substrings are clustered using GMM on the first H dimensions of the LSI representation.
4. The clusters are now forming a new and much smaller vocabulary for the substrings. Each substring is transformed to an index associated with the closest cluster.
5. A HMM is trained for each class of documents using the new vocabulary.
6. New documents are classified using the HMM forward backward classification algorithm.

The classification algorithm is using the forward-backward approach which is also used to estimate the parameters.

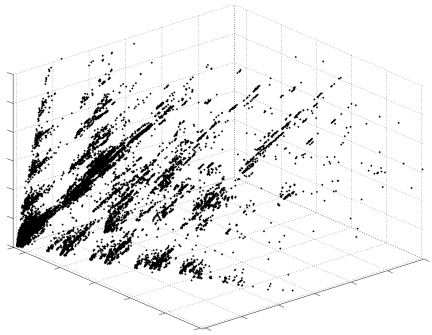


Figure 4.1: Space for the new vocabulary.

4.5 HMM with LSI emission probabilities

In the section about the hidden Markov model, we reject the model for use on text directly, while the high number of parameters for the model would make it

converge slowly, due to size of the vocabulary. It is further undesirable to use the HMM directly on each single class while the classes won't be able to share the emission probabilities. It is desirable to share the emission probabilities for all the classes while they can be thought of as latent topics, where there is a latent topic for each single state in the HMM. This idea is conceptually similar to the ideas from latent semantic indexing and it's varieties (Furnas et al., 1988; Deerwester et al., 1990; Hofmann, 1999; Kolenda et al., 2002; Blei et al., 2002; Blei et al., 2003).

The problems of shared latent topic emissions could be overcome by redefining the HMM to be a model with more state space transition models, but only one single state emission model. This model would be likely to inherit the slow convergence property of the normal HMM. We therefore reject the model here, knowing that it probably would be the best modeling approach to the problem.

The alternative to a redefined HMM, is to estimate the emission probabilities b_{s_j,v_m} using another algorithm and keeping them fixed when first estimated. Using this approach it would only be necessary to estimate the state transition parameters a_{s_i,s_j} and initial state probabilities π_{s_i} for each separate class. This estimation procedure would further not need to run in an iterative EM-loop where the one set of parameters are estimated based on an estimate of the other set of parameters. The transition parameters would therefore only need one or very few iterations to converge.

There are more alternative ways to determine a set of shared latent topic emission parameters. Three possible approaches are *independent component analysis* (ICA) (Bell & Sejnowski, 1995b; Bell & Sejnowski, 1995a; Molgedey & Schuster, 1994), *singular value decomposition* (SVD) (Madsen et al., 2003) and *non-negative matrix factorization* (NMF) (Lee & Seung, 1999; Lee & Seung, 2001). The latter approach has the advantage of estimating non-negative values when factorizing the data, which is valuable since these values reflect emission probabilities, i.e. they have to be positive and sum to zero. NMF has also shown valuable for text clustering previously (Xu et al., 2003). In practise however the NMF does not work well with the sparse structure of the text data, resulting in very few active words in each NMF latent topic. When only few words are active it is necessary to either use a lot of smoothing or use many latent topics. Neither of these fixes are likely to give us a good model or classifier, so we turn to SVD approach instead. The latent topics estimated by the SVD all have many active words. The problem of probabilities being negative is solved by simply setting negative values equal to zero, and then normalize the distribution.

The procedure of using the HMM state space model with LSI estimated emission probabilities, takes place in the following way:

1. A common set of HMM emission parameters are estimated using the LSI approach on the documents using the histogram representation.
2. A set of HMM state space parameters are estimated for each class using the word sequences for each document.
3. New documents (sequences of words) are classified using the HMM forward backward classification algorithm.

4.6 Experiments

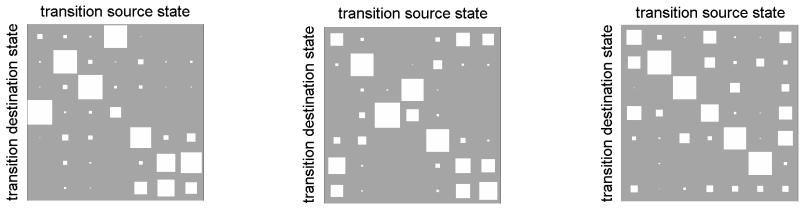
Like the previous chapters, we are here working with the three corpora: email, WebKB and multimedia. The number of words in the three corpora are reduced by use of stemming and stop-word removal. Though we here only show results for the email-data, similar results where gained by use of the two other data-sets. The TF-IDF transformation has been applied to the document collections, when performing experiments using the HMM with LSI-GMM generated vocabulary. In the experiments where using the HMM with LSI emission probabilities, the TF-IDF transformation has not been applied. The reason is that the HMM works on sequences where each unit in the sequence must be unity. A weighting scheme could be applied to the HMM, where the TF-IDF coefficients could be applied as weights. At first we are interested in investigating if the model works conceptually, and have therefore skipped the transformation step.

We start by training the HMM with LSI-GMM generated vocabulary (HLG) using the email-data. The largest class in the email data-set (spam) accounts for 0.55% of the emails. A naive classifier should therefore have a classification accuracy of about 0.55. The two models considered should therefore have generalization error below 0.45%.

We find that the HLG approach works best when a LSI subspace of 4 dimensions is used to form the new HLG generated vocabulary. A set of 100 gaussians is used to cover the 4-dimensional space forming an new vocabulary of 100 words. The first three dimensions of the subspace are shown in Figure 4.1, where the structure of the data are much different from the structure found by the generic LSI representation Figure 2.2. Each cluster that is put in the space in Figure 4.1 now represents a word in the new vocabulary.

Estimating the HMM for the new sequences, the transition probabilities for the three classes in the email-set, show us whether there is a sequential difference between the three classes that is captured by the model. In Figure 4.2, a graphical illustration of the transition probabilities is shown. Seven states is used in

the HMM to best model the new sequences.



(a) Job (b) Conference (c) Spam

Figure 4.2: Graphical illustration of the transition probabilities for the HLG model. For the Job category (a), state 1 and 4 are paired and almost isolated from the other states making them a semantic chain for the category. Similarly state 5, 6 and 7 form a state group that is likely to generate long sequences of words. The Conference category (b) has a similar group formation where the states 3 and 4 form a group, and state 1,7 and 8 form a group. The spam category (c) does not have the same strong group formation as the two other categories, but is instead less symmetric. There is however weak group formation between the states 1, 2 and four, and the states 3 and 5. The illustration of the transition probabilities reveal that there is a sequential pattern that is captured by the model.

The illustration in Figure 4.2 shows clearly that there is information in the order in which the words appear in a document, and that this information can be captured by the HMM.

There are more settings that determine the optimal HLG model, i.e. number of LSI dimensions, number of states in the HMM, the number of gaussian mixtures and the length of the substrings used to form the vocabulary. In Figure 5.1 the classification accuracy for the HLG model as function of the substring length is plotted, where the settings for the remaining parameters are close to optimal.

The HLG model has an accuracy that is lower than the accuracy of the LSI model, for all possible substring-length values.

We next turn to the HMM model with LSI estimated emission probabilities. We again take a look at the transition probabilities for the three classes in the email-set, for discovering whether there is a sequential difference between the three classes that are captured by the model. In Figure 4.4 an illustration of the transition probabilities is shown. The Illustration shows transition probabilities

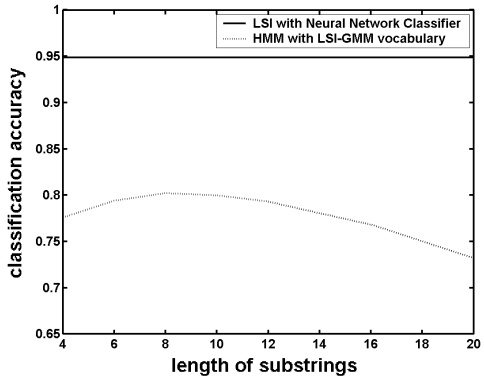
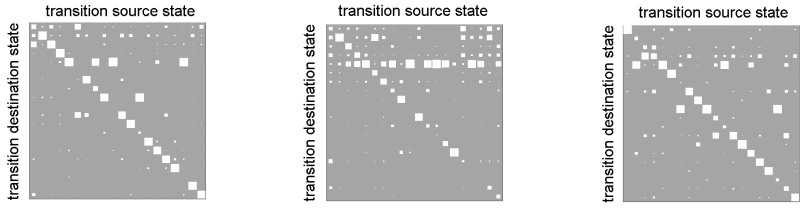


Figure 4.3: Classification accuracy using the HMM with LSI-GMM reduced vocabulary, as function of the substring length. The classification results are compared with a neural network classifier using a LSI subspace on TFIDF normalized data. The data used are email data where 20% of the data are used for training.

for a model with 20 states, where the best model instead uses about 120 states. The smaller model is shown since it is easier to survey.



(a) Job (b) Conference (c) Spam

Figure 4.4: Graphical illustration of the transition probabilities for the LSI-HMM model. The group formations for the LSI-HMM state space is harder to discover than those for the HLG model. There is, however small groupings like state 1 and 3 for the Job category (a).

The formation of state groups is not as obvious as it was for the HLG model. It is therefore less obvious whether or not, the LSI-HMM model has captured much sequential information about the documents.

In Figure 4.5 we show the learning curves for the LSI-HMM compared with the generic LSI model. The LSI model performs slightly better than the LSI-HMM model when used for classification. The performance of the two models follow each other, for the whole range of training set sizes.

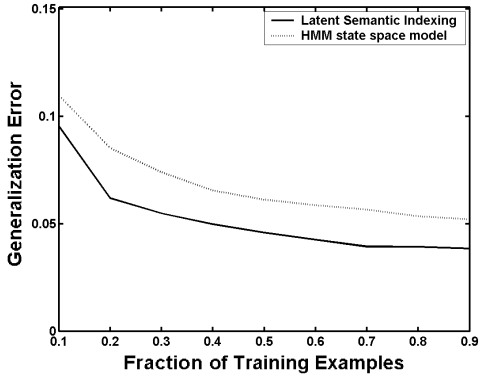


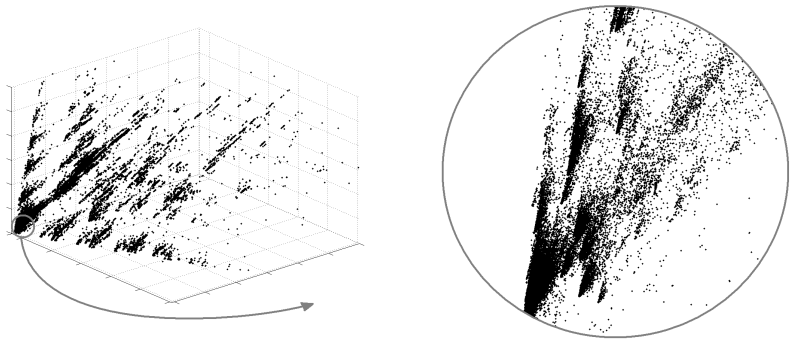
Figure 4.5: Learning curves for the LSI HMM model. The LSI-HMM model is slightly worse at classifying documents correctly than the generic LSI model.

4.7 Discussion

The first approach to capture information from the word sequences in documents, the HLG model did not perform well at the classification task. The state transition probabilities however showed that word order information was captured in the model. The reason for the lack of classification performance is therefore not to be found in the use of the state space model, but rather in the transformation of the vocabulary to a lower dimensional LSI-GMM vocabulary.

Previous experiments have shown that the 50 or more LSI components are needed to create an efficient classifier. It is therefore likely that valuable information is lost when we only use 4 LSI principal component directions here. The reason for only using few LSI components is that the use of many components makes it hard for the GMM to model the semantic space correctly. As illustrated in Figure 4.6, the density in the LSI subspace is very high in some areas, and the clusters are not very gaussian in shape. A very high amount of gaussian mixtures is therefore needed if they should cover a higher dimensional subspace. In practise the gaussian mixtures are poor at modeling the new LSI subspace, while they tend to cluster around high density areas when too many LSI dimensions are used. This gives a bad fit to many of the outer data-points,

resulting in poor classification performance.



(a) full space (b) zoom

Figure 4.6: Zooming in on the LSI space of the document substrings. Some of the spaces are very dense on data, making the areas very attractive for the gaussian mixtures. When using high dimensional representations of the LSI substring space, the outer data points therefore tend to be badly modeled. Since much variation exists for the data in non dense areas, lack of modeling in these areas are likely to result in loss of information.

The HMM model with LSI estimated emission probabilities was much better at classifying documents correctly than the HLG model. The state transition probabilities did however not seem to capture any valuable information about the differences in word sequences for the three classes. It is likely that a true EM estimate of emission probabilities would have resulted in different transition probabilities that would capture more of the word order information, leading to better classification. A true EM estimate of shared latent topic emissions will however require that the a new HMM must be redefined and update rules determined.

4.8 Summary

We have used two state space model approaches to capture the information, that is contained in the order in which words appear in documents. The first approach involved a transformation of the document vocabulary into a smaller LSI vocabulary, whereon a HMM could be trained. This approach lacked in

classification ability but was conceptual successful at capturing word order information.

The second approach involved making an estimate of latent topic emission probabilities for at HMM using LSI. This approach had less success at capturing word order information, but was better at the classification task. We have hope that the HMM approach will have greater success by the development of a HMM with shared latent topic emission probabilities.

Chapter 5

Dirichlet

5.1 Introduction

In the previous chapters we have used discriminative classifiers to categorize text documents. In this chapter and the following chapters the focus is set on categorization based on generative probabilistic models. Generative approaches to classification are popular since they are relatively easy to interpret and can be trained quickly. With these generative approaches, the key problem is to develop a probabilistic model that represents the data well. Unfortunately, for text classification too little attention has been devoted to this task. Instead, the generic multinomial model is typically used. Recent work (Rennie et al., 2003) has pointed out a number of deficiencies of the commonly used multinomial model, and suggested heuristics to improve its performance.

The central problem with the multinomial model is that it assumes that documents are created from a static process, i.e. the multinomial model does not change it's emission probabilities during the process of generating a document. This assumption is opposed to the true nature of text documents, where a word at a given place in a document is dependent on the previous words in the document and especially dependent on whether the word itself has appeared previously. If a given word has appeared previously in a document, it is much more likely to appear again later in that document, i.e. the word appears in

bursts. The fact that words tend to appear in bursts, as opposed to being emitted independently has been investigated previously in (Church & Gale, 1995; Katz, 1996).

The burstiness issue has previously been addressed a couple of times. Rennie et al. (2003) address this issue by log-normalizing counts, hereby reducing the impact of burstiness on the likelihood of a document. Teevan and Karger (2003) empirically search for a model that fits documents well within an exponential family of models, while Jansche (2003) proposes a zero-inflated mixture model.

In this chapter we go further. We show that the multinomial model is appropriate for common words but not for other words. The distributions of counts produced by multinomial distributions are fundamentally different from the count distributions of natural text. Zipf's law (Zipf, 1949) states that the probability p_i of the occurrence of an event follows a power law $p_i \approx i^{-a}$, where i is the rank of the event and a is a parameter. The most famous example of Zipf's law is that the frequency of an English word, as a function of the word's rank, follows a power law with exponent close to minus one.

We subsequently present a different probabilistic model that, without any heuristic changes, is far better suited for modeling the burstiness phenomenon. We propose to model a collection of documents using the Dirichlet distribution (Minka, 2003) as an alternative to the multinomial. The dirichlet distribution has one additional degree of freedom, which we shall see allows it to capture burstiness. The dirichlet distribution can be thought of as a bag-of-scaled-documents where the multinomial distribution is a bag-of-words model.

In the following we continue to represent an individual document as a histogram of word counts (Salton et al., 1975). We further assume that word emissions are independent given the document category, i.e. the naive Bayes property holds. This property is not valid (Lewis, 1998), but naive Bayes models remain popular (McCallum & Nigam, 1998; Sebastiani, 2002) because they are fast and easy to implement, they can be fit even with limited training data, and they do yield accurate classification when heuristics are applied (Jones, 1972; Rennie et al., 2003).

Dirichlet distributions have been used previously to model text, but our approach is fundamentally different. In the LDA approach (Blei et al., 2003) the Dirichlet is a distribution over topics, while each topic is modeled in the usual way as a multinomial distribution over words. In our approach, each topic, i.e. each class of documents, is modeled in a novel way by a Dirichlet distribution instead of by a multinomial. Our approach is therefore complementary to the LDA and related approaches. This will be explained more deeply in one of the following sections.

5.2 Multinomial modeling of text

When using a multinomial distribution for text modeling, the multinomial specifies the probability of observing a given vector of word counts, where the probability θ_w for the emission of word w is subject to the constraints $\sum_w \theta_w = 1$ and $\theta_w > 0$. The probability of a document x represented as a vector of word counts x_w is

$$p(x|\theta) = \frac{n!}{\prod_{w=1}^{W} x_w!} \prod_{w=1}^{W} \theta_w^{x_w} \tag{5.1}$$

where x_w is the number of times word w appears in the document, θ_w is the probability of emitting word w, W is the size of the vocabulary, and $n = \sum x_w$. When looking at the multinomial distribution in eq. (5.1), the distribution consists of two parts. The first part is the scaling of the distribution, accounting for all the possible ways a vector of length $|\mathbf{x}|$ can be selected. The scaling penalizes vectors where most words occur rarely. The second part of the distribution determines the probability of a given vector of words based on how likely each word is to occur.

The multinomial distribution is different for changing document lengths n. This is not a problem when learning the parameters since it is possible to generalize over documents with different lengths. The maximum likelihood parameter estimates $\hat{\theta}$ are

$$\hat{\theta}_w = \frac{\sum_{d=1}^{D} x_{dw}}{\sum_{w'=1}^{W} \sum_{d=1}^{D} x_{dw'}} \tag{5.2}$$

where d is the document number and D is the number of documents. These estimates depend only on the fraction of times a given word appears in the entire corpus.

When the multinomial model is used to generate a document, the distribution of the number of emissions (marginal distribution) of an individual word is the binomial:

$$p(x_w|\theta) = \binom{n}{x_w} \theta_w^{x_w} (1 - \theta_w)^{n - x_w}. \tag{5.3}$$

The multinomial distribution can as the hidden Markov model be understood as an urn scheme. The multinomial assembles a simple urn scheme, where a single urn contains an infinite number of colored balls. The multinomial parameters θ_w determine the fraction of the balls in the urn that have the color w. When a sequence of balls are to be generated, a new ball is drawn from the urn and the color is noted whereafter the ball is returned into the urn. This procedure is repeated until the desired sequence length is obtained.

5.3 The burstiness phenomenon

The term "burstiness" (Church & Gale, 1995; Katz, 1996) describes the behavior of a rare word appearing many times in a single document. Because of the large number of possible words, most words do not appear in a given document. However, if a word does appear once, it is much more likely to appear again, i.e. words appear in bursts. To illustrate this behavior, the probability that a given word occurs in a document exactly x times is shown in Figure 5.1 for the industry sector corpus. Words have been split into three categories based on how often they appear in the corpus. The categories are "common", "average", and "rare". The common words are the 500 most frequent words; they represent 1% of the words in the vocabulary and 71% of the emissions. The average words are the next 5000 most common words; they represent 10% of the vocabulary and 21% of the emissions. The rare words are the rest of the vocabulary (50,030 words) and account for 8% of the emissions.

A few things should be noted about Figure 5.1. Not surprisingly, common words are more probable than average words which are more probable than rare words. Interestingly, though, the curves for the three categories of words are close to parallel and have similar decay rates. Even though average and rare words are less likely to appear, once a word has appeared, the probability that it will occur multiple times is similar across all words.

Equation (5.3) shows that it is unlikely under the multinomial model for a word to occur many times in a document, because the single word count distribution decays exponentially. Figure 5.2 shows the average word count probabilities from ten synthetic corpora generated from a multinomial model trained on the industry sector corpus. Each synthetic corpus was generated so its documents have the same length distribution as documents in the industry sector corpus.

The multinomial captures the burstiness of common words, but the burstiness of average and rare words is not modeled correctly. This is a major deficiency in the multinomial model since rare and average words represent 99% of the

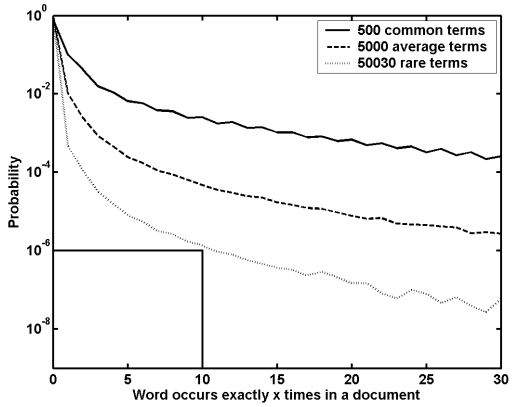


Figure 5.1: Count probabilities of common, average and rare words in the industry sector corpus. The figure shows, for example, that the probability a given rare word occurs exactly 10 times in a document is 10^{-6}. The ripple effect seen in common words occurs because no vocabulary pruning is done, so certain HTML keywords such as "font" or "table" occur an even number of times in beginning and ending tags. The three curves are almost parallel showing that when a word has occurred in a document, the likelihood of it occurring again is almost independent of how frequent the word usually appears in documents.

vocabulary and 29% of emissions and, more importantly, these words are key features for classification. An explanation for this behavior is that the common words are more likely to satisfy the independence assumption, since many of the common words are non-content, function words. The rare and average words are information-carrying words, making them more likely to appear if they have already appeared in a document.

Figure 5.3 shows the simplex of possible count vectors for three words when a multinomial model is used and the sum of the counts is $n = 50$. All the probability mass is close to the most likely count vector, so burstiness is not likely. If bursts were likely, then the probability mass would be on the edges and corners of the simplex.

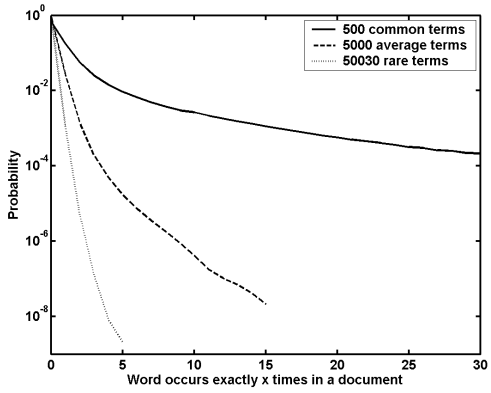


Figure 5.2: Count probabilities for a maximum likelihood multinomial model, trained with the industry sector corpus. The curves are not parallel, showing that burstiness is not an equally likely phenomenon for the different word-groups in the vocabulary, when the multinomial model is considered.

5.4 Dirichlet modeling of text

The Dirichlet distribution is a probability density function over distributions. It is defined as

$$p(\theta|\alpha) = \frac{\Gamma\left(\sum_{w=1}^{W} \alpha_w\right)}{\prod_{w=1}^{W} \Gamma(\alpha_w)} \prod_{w=1}^{W} \theta_w^{\alpha_w - 1} \tag{5.4}$$

where θ is a vector in the W-dimensional probability simplex, i.e. $\sum_w \theta_w = 1$. The α vector entries are the parameters of the Dirichlet.

When modeling text, the θ vector represents a document, making the model have the form $data^{parameter}$. This form makes Dirichlet models qualitatively similar to Zipf distributions, where the parameter is an exponent. In contrast, the multinomial model has the form $parameter^{data}$. By rewriting the Dirichlet distribution in the exponential family form,

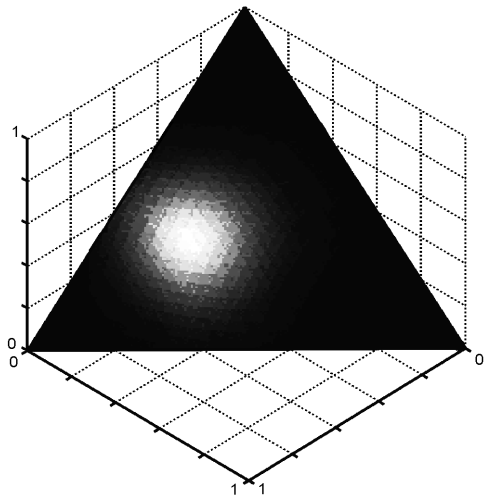


Figure 5.3: Probability simplex of possible count vectors using the multinomial bag-of-words model with parameters $\theta = \{0.44, 0.25, 0.31\}$ and $n = 50$. The simplex is defined by the plane stretching from the three corners (n,0,0), (0,n,0) and (0,0,n), where n is set to 1 for simplicity. The simplex defines the probability of all possible outcomes when drawing from an urn. In reality the simplex is not a plane, since only a discrete set of outcomes are possible. The simplex shows that is it unlikely to draw a set with bursty counts, while all of the probability mass is located near the center of the simplex.

$$\begin{aligned} \log p(\theta|\alpha) &= \sum_{w=1}^{W} (\alpha_w - 1) \log \theta_w \\ &+ \log \Gamma(\sum_{w=1}^{W} \alpha_w) - \sum_{w=1}^{W} \log \Gamma(\alpha_w) \end{aligned} \tag{5.5}$$

we see that the log transform of the data is naturally considered. This is again in contrast to the multinomial in exponential form. In (Rennie et al., 2003) it has been shown that carrying out a log-transformation on the data enhances the multinomial models ability to perform document categorization.

In the bag-of-words representation, documents are vectors of word counts. The Dirichlet distribution is a distribution not over count vectors but over probability vectors. The obvious choice is to let θ be a scaled version of the document vector. We can view this approach as drawing a scaled bag of words (representing one

document) from the Dirichlet bag-of-scaled-documents.

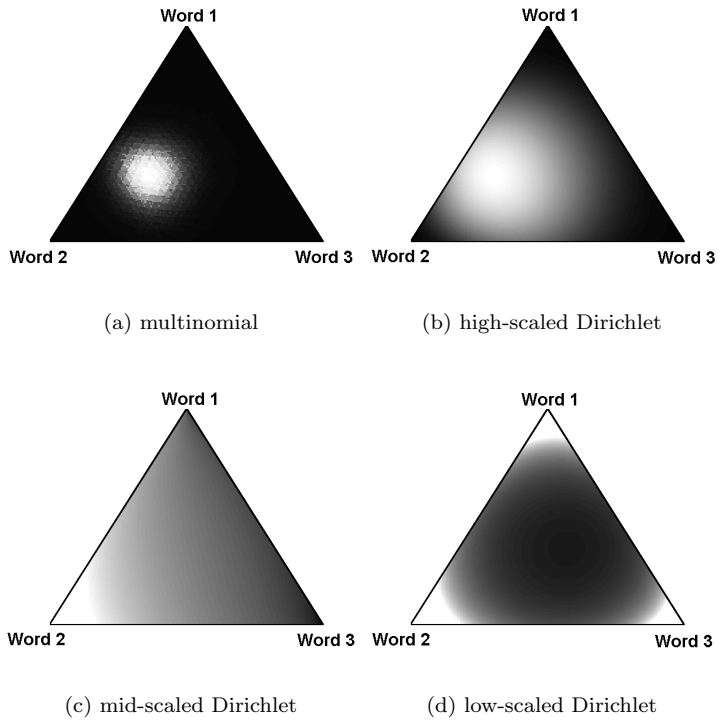
Comparing the Dirichlet distribution with the multinomial distribution, the Dirichlet actually is a distribution of multinomial distributions. Both distributions have the same number of parameters but the Dirichlet has an extra degree of freedom, by not having the constraint that the parameters have to sum to one. This extra degree of freedom, the scaling of the parameters allows the dirichlet to model various degrees of burstiness. In Figure 5.4 the probability simplex for a multinomial distribution is compared with similar simplices for the Dirichlet distribution where the Dirichlet parameters for each simplex use different scaling. The figure shows that by low-scaling the parameters, the Dirichlet allows for bursty behavior, while high-scaling allows for non-bursty behavior. Since a Dirichlet has only a single extra degree of freedom compared to a multinomial, it cannot model the individual burstiness of each word. This inflexibility is a trade-off between the expressiveness of the model and its learnability with limited training data.

Figure 6.2 shows the average word count probabilities from ten synthetic corpora. The corpora were generated using a Dirichlet model which was trained in a similar way as the multinomial in Figure 5.2. The Dirichlet generated documents have been scaled up to the correct document length, where a random process has determined which word fractions that were rounded up and down. The Dirichlet generated corpus has a degree of burstiness that is similar to that of the original corpus.

The Dirichlet distribution also has an analogy in the collection of urn models. The Dirichlet distribution models an urn containing an infinite amount of sub-urns, where each sub-urn contains an infinite amount of balls, i.e. each sub-urn is a multinomial urn. The scaling of the Dirichlet parameters tells whether the distribution of balls in the sub-urns are likely to be similar or not. The unscaled Dirichlet parameters tell what the distribution in an average urn is likely to be.

5.5 Discussion

We do not present any experimental result in this chapter, while the Dirichlet model does not perform well in the categorization task, when comparing with the multinomial. The difficulty with this approach is that document vectors are sparse in nature, i.e. each document tends to contain only a small subset of the vocabulary, resulting in many of the entries being zero. Since the Dirichlet likelihood for a probability vector is zero if the vector contains any zeros, smoothing of the training data is required before a Dirichlet distribution can be



(a) multinomial (b) high-scaled Dirichlet

(c) mid-scaled Dirichlet (d) low-scaled Dirichlet

Figure 5.4: Word probability simplex with the multinomial parameters (a) 0.44, 0.25, 0.31. and three word probability simplices with Dirichlet parameters (b) 3.94, 2.25, 2.81, (c) 1.32, 0.75, 0.93 and (d) 0.44, 0.25, 0.31. By changing the scaling of the Dirichlet parameters, the probability mass is moved around the simplex. When the parameters are high-scaled, the probability mass is located near the center of the simplex, making non-bursty behavior likely. When the parameters are low-scaled, the probability mass is located near the corners, making bursty behavior a more likely phenomenon.

estimated. In practice, this results in an over-smoothed Dirichlet distribution, where all the rare words have about the same probability of appearing in all the classes. Since the rare words contain most of the discriminative information, this model is not useful for document classification.

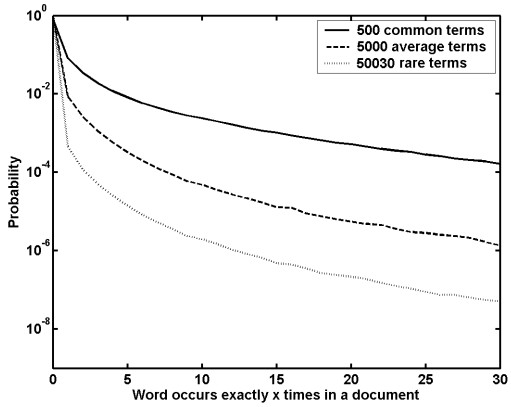


Figure 5.5: Count probabilities for a maximum likelihood Dirichlet model, trained with the industry sector corpus. Like in the original corpus, burstiness is almost equally likely for the three categories of words.

5.6 Summary

The burstiness phenomenon in text was introduced and an analysis of the generative probabilistic multinomial model's ability to model burstiness was carried out. The analysis showed that the multinomial does not have the degree of freedom to correctly model word burstiness, resulting in poor modeling of this phenomenon for 99% of the words in the vocabulary. The Dirichlet model has an extra degree of freedom, and has shown an ability to use this freedom to correctly model burstiness in text. Smoothing problems have however prevented the Dirichlet model in being a successful at categorizing documents.

Chapter 6

Dirichlet Compound Multinomial

6.1 Introduction

In the previous chapter 5, we looked at the word burstiness phenomenon for text documents and showed that the commonly used multinomial couldn't model this phenomenon correctly. The Dirichlet distribution was able to model word burstiness correctly but was unsuccessful when categorizing document, while the distribution got over smoothed when used with sparse data.

Text is sparse by nature so we must consider another model that can handle data sparseness while at the same time being able to model burstiness. Since the multinomial is handling sparseness well and the Dirichlet is handling burstiness well, an obvious approach is to combine the two distributions in a compound model, the Dirichlet compound multinomial (Minka, 2003). The Dirichlet compound multinomial can be thought of as a bag-of-bags-of-words model. Compared with the multinomial, the DCM has an additional degree of freedom in its parameters, which is similar to the degree of freedom in the Dirichlet model, allowing the two models to capture the burstiness in text.

6.2 Dirichlet Compound Multinomial Modeling of Text

The Dirichlet compound multinomial (DCM) is a compound distribution defined by the Dirichlet distribution and the multinomial distribution. By compounding the Dirichlet with the multinomial we inherit the multinomial distributions ability to model sparse data. The DCM defines a probability distribution over histograms of counts x_d with length n_d, similarly to the multinomial distribution. The parameterization of the DCM is similar to that of Dirichlet distribution, i.e. parameterized by $\alpha \geq 0$.

The generative process of the DCM is hierarchical. To generate a document using the DCM, a sample is first drawn from a Dirichlet. This sample is a probability distribution which can be used as parameters for the multinomial distribution. From the multinomial distribution words are iteratively drawn for the document, until the desired document length is acquired.

The likelihood for a count histogram is specified by the sum of weighted count histogram probabilities found by the multinomial distribution. The weighting is based on the parameterization of the multinomial, defined by the Dirichlet distribution. The likelihood $p(x|\alpha)$ for the DCM is defined in equation 6.1. Although we use the parameters θ, the only genuine parameters for a DCM are the α parameter entries. The likelihood of a document is an integral over θ vectors weighted by a Dirichlet distribution:

$$
\begin{aligned}
p(x|\alpha) &= \int_\theta p(x|\theta)p(\theta|\alpha)d\theta \\
&= \int_\theta \frac{n!}{\prod_{w=1}^{W} x_w!} \left(\prod_{w=1}^{W} \theta_w^{x_w} \right) \frac{\Gamma\left(\sum_{w=1}^{W} \alpha_w \right)}{\prod_{w=1}^{W} \Gamma(\alpha_w)} \prod_{w=1}^{W} \theta_w^{\alpha_w - 1} d\theta \\
&= \frac{n!}{\prod_{w=1}^{W} x_w!} \frac{\Gamma\left(\sum_{w=1}^{W} \alpha_w \right)}{\prod_{w=1}^{W} \Gamma(\alpha_w)} \int_\theta \prod_{w=1}^{W} \theta_w^{\alpha_w + x_w - 1} d\theta \\
&= \frac{n!}{\prod_{w}^{W} x_w! \, \Gamma\left(\sum_{w=1}^{W} x_w + \alpha_w \right)} \frac{\Gamma\left(\sum_{w=1}^{W} \alpha_w \right)}{} \prod_{w=1}^{W} \frac{\Gamma\left(x_w + \alpha_w \right)}{\Gamma(\alpha_w)}.
\end{aligned}
\tag{6.1}
$$

The last step of equation (6.1) is obtained by noticing that $\prod_{w=1}^{W} \theta_w^{x_w}$ combined with $\prod_{w=1}^{W} \theta_w^{\alpha_w - 1}$ is the un-normalized version of the Dirichlet distribution $p(\theta|\alpha + x)$, and using the fact that $\int p(\theta|\alpha)d\theta = 1$.

There exists no closed-form solution for the maximum likelihood parameter values for the DCM model. An iterative gradient descent optimization method can be used to estimate the α vector by computing the gradient of the DCM log likelihood. Two bound inequations are used with the gradient, leading to the update,

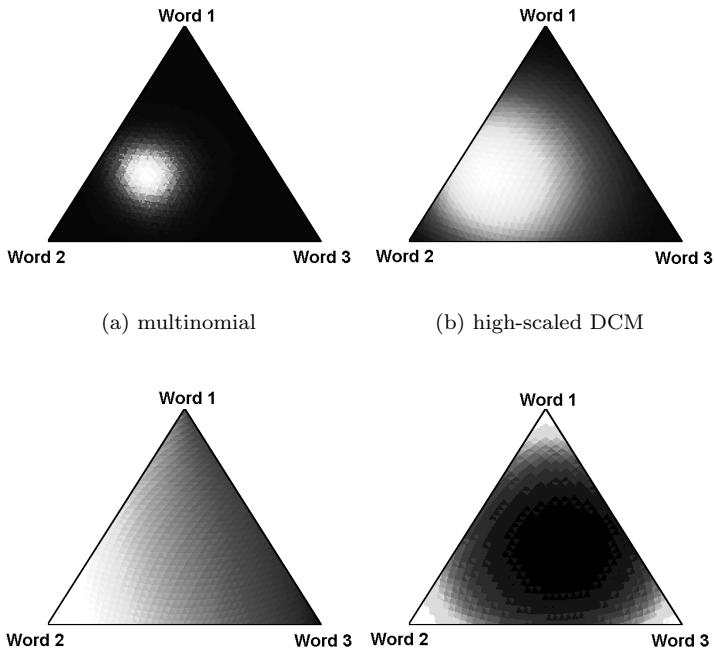
$$\alpha_w^{new} = \alpha_w \frac{\sum_{d=1}^{D} \Psi(x_{dw} + \alpha_w) - \Psi(\alpha_w)}{\sum_{d=1}^{D} \Psi(x_{dw} + \sum_{w'=1}^{W} \alpha_{w'}) - \Psi(\sum_{w'=1}^{W} \alpha_{w'})} \tag{6.2}$$

where the digamma function Ψ is defined as $\Psi(\alpha) = \frac{d}{d\alpha} \log \Gamma(\alpha)$. For more information see Minka (2003).

In Chapter 5 we saw that the Dirichlet distribution could use the extra degree of freedom to capture the word burstiness. In Figure 7.2 it is shown that the DCM can use its flexibility in a similar way. The figure shows the simplex of count probabilities given by equation (6.1) for three words with $n = 50$, for different α vectors. When the parameters are small, most probability mass is located near the corners of the simplex. When the parameters are large, most mass is near the center of the simplex, modeling word counts that are not bursty. As the parameters tend to infinity, the DCM model approaches equivalence with a multinomial model. The surfaces of the DCM simplexes are not smooth, while the DCM only is defined for integer counts and not fractional counts.

Figure 5.2 showed that the multinomial was unable to correctly model the burstiness of natural text, and Figure 6.2 showed that the Dirichlet Distribution was able to model this phenomenon. Figure 6.2 shows the probability of a term appearing multiple times in a document under the DCM model. The experimental design is similar to those of the previous chapter. Like the Dirichlet, the DCM can model burstiness for all word types. The curves for the three categories of words are again close to being parallel.

Figure 7.2 and 6.2, conclude that the DCM has the ability to model word burstiness. By having compounded with the multinomial the DCM inherits the ability to model sparse data. Before looking at the experiments with the DCM we will take a short look deeper into the DCM model and compare it with the LDA model.



(a) multinomial (b) high-scaled DCM

(c) mid-scaled DCM (d) low-scaled DCM

Figure 6.1: Word probability simplex with the multinomial parameters (a) 0.44, 0.25, 0.31. and three word probability simplices with DCM parameters (b) 3.94, 2.25, 2.81, (c) 1.32, 0.75, 0.93 and (d) 0.44, 0.25, 0.31. By changing the scaling of the DCM parameters, we see that the probability mass is moved around the simplex, in a similar manner as for the Dirichlet Distribution. When the scaling of the DCM moves towards infinity, the DCM becomes the multinomial.

6.3 DCM marginal distribution

The marginal distribution for the DCM model, determines the emission distribution for a single word in a document. The marginal distribution $P(x_w|\alpha, n)$ can be found integrating over the marginal distribution for the multinomial $P(x_w|\theta_w, n)$ multiplied with the probability $P(\theta_w|\alpha)$ of a single multinomial parameter θ_w, given the dirichlet parameters α and the length of the document. The marginal distribution for the multinomial is the binomial distribution.

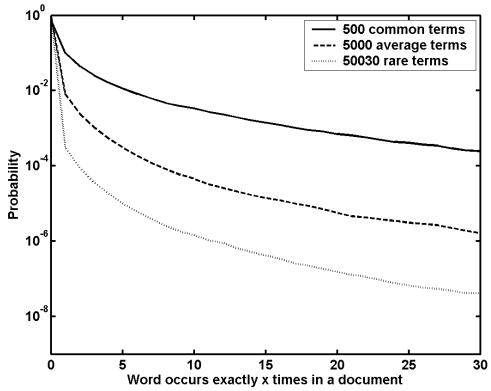


Figure 6.2: Count probabilities for a maximum likelihood DCM model, trained with the industry sector corpus. The DCM models burstiness in a way that is similar to burstiness in the real data.

$$P(x_w|\alpha, n) = \int_0^1 P(x_w|\theta_w, n)P(\theta_w|\alpha)\partial\theta_w \tag{6.3}$$

In the case where only two-dimensions exist, i.e. there is only two words in the vocabulary, the marginal probability $P(\theta_w|\alpha)$ of a single multinomial parameter θ_w, is easily determinable from the dirichlet distribution in equation 5.4. In the two-dimensional case, the marginal distribution is found by integrating out the other parameter,

$$P(\theta_w|\alpha) = \int P(\theta|\alpha)\delta\theta_\bullet = \frac{\Gamma\left(\sum_{w=1}^{W}\alpha_w\right)}{\prod_{w=1}^{W}\Gamma(\alpha_w)}\theta_w^{\alpha_w-1}(1-\theta_w)^{\alpha_\bullet-1} \tag{6.4}$$

where $\theta_\bullet = \theta \setminus \{\theta_w\}$ is the set of multinomial parameters excluding θ_w and similarly $\alpha_\bullet = \alpha \setminus \{\alpha_w\}$ is the set of dirichlet parameters excluding α_w. In the two-dimensional case $\theta_\bullet$ and $\alpha_\bullet$ are the parameters for the other word. This makes equation 6.4 simple while $\theta_\bullet = 1 - \theta_w$.

When the number of dimensions rises above two, the integral in equation 6.4 becomes hard. Fortunately the integral can be simplified to the two-dimensional case where $\alpha_\bullet = \sum_{w'}\alpha_{w'} \setminus \{\alpha_w\}$ and $\theta_\bullet = 1 - \theta_w$. The marginal distribution

for the DCM therefore becomes the beta-binomial, see also (Ishii & Hayakawa, 1960; Johnson et al., 1997).

$$
\begin{aligned}
P(x_w|\alpha, n) &= \int_0^1 P(x_w|\theta_w, n) \int_0^{1-\theta_w} p(\theta|\alpha)\partial\theta_\bullet \partial\theta_w \\
&= \int_0^1 \binom{n}{x_w} \theta_w^{x_w} \left(1-\theta_w\right)^{n-x_w} \\
&\quad \frac{\Gamma\left(\sum_{w=1}^{W} \alpha_w\right)}{\prod_{w=1}^{W} \Gamma(\alpha_w)} \theta_w^{\alpha_w - 1} (1-\theta_w)^{\alpha_\bullet - 1} \partial\theta_w \qquad (6.5) \\
&= \binom{n}{x_w} \frac{\beta(x_w + \alpha_w - 1, n - x_w + \alpha_\bullet - 1)}{\frac{\Gamma(\alpha_w)\Gamma(\alpha_\bullet)}{\Gamma(\alpha_w + \alpha_\bullet)}} \\
&= \binom{n}{x_w} \frac{\beta(x_w + \alpha_w - 1, n - x_w + \alpha_\bullet - 1)}{\beta(\alpha_w, \alpha_\bullet)}
\end{aligned}
$$

where $\beta()$ is the Beta function and $\Gamma()$ is the Gamma function.

Using the polya urn model representation instead of the DCM, it is easily revealed that the marginal distribution can be determined by simplifying to the two-dimensional case. The polya urn drawing distribution is dependent on the previously drawn balls and the initial set of balls, i.e. the previous counts and the α parameters. All the balls with another color than the one we are interested in can therefore be joined into one ball color $\alpha_\bullet$, leaving us with the two-dimensional case.

6.4 The polya urn model

Like the multinomial, the DCM model can be thought of as an urn model, containing balls with different colors. Depending on the policy associated with the urn, the sequences of balls drawn follow different distributions. The urn associated with the multinomial distribution contains the same distribution of balls during a whole drawing experiment, i.e. every time a ball has been drawn from the urn, the color is noted whereafter the ball is returned into the urn. In a sense the multinomial urn is therefore static, and the distribution drawn from the urn, will converge towards a scaled version of the distribution in the urn. For the same reason, different drawing distributions from the urn will also

tend to be homogenous, while there are no dynamics changing the urn, making the urn process stationary. The θ parameters of the multinomial are directly dependent on the fraction of balls with a given color in the urn,

$$\theta_c = \frac{\#\ balls\ with\ color\ c}{\sum_{c'} \#\ balls\ with\ color\ c} \tag{6.6}$$

where θ_c the multinomial parameter associated with balls of color c.

The urn scheme associated with the DCM is named the polya urn. When drawing a ball from the polya urn, the ball color is noted, the ball is returned into the urn and a ball with similar color is also put into the urn. This urn policy makes it more likely to draw a ball with the same color as the previously drawn ball. This policy also results in the fact that the urn will contain a lot of balls with a single color, after drawing many more balls than there initially were in the urn. The ball distributions generated from the polya urn are therefore in-homogenous and the generating process is non-stationary, in contrast to the multinomial generated distributions.

From the polya urn policy it follows that the urn distribution by the end of the experiment is equal to the initial urn distribution plus the distribution drawn from the urn. When the number of balls initially in the urn is going towards infinity, the balls added to the urn distribution result in insignificant changes to the urn distribution. In the infinite case, the polya urn therefore becomes similar to the urn associated with the multinomial. When the urn contains few balls at the beginning of the experiment, the added balls result in a significant change to the distribution.

The parameters of the DCM is directly related to the initial number of balls in the polya urn, $\alpha_c = \#\ balls\ with\ color\ c$, where the initial number of balls can be fractional.

Due to the fact that extra balls are added to the polya urn, when a ball is drawn, the drawing distribution of the polya urn scheme is constrained to be more diverse than distributions drawn from the multinomial. If more concentrated distributions are desirable the urn scheme should be changed, so balls drawn from the urn are not returned. This however would mean that the urn could run out of balls, which might result in mathematical complications.

6.5 Comparing with Latent Dirichlet Allocation

Dirichlet distributions have been used previously to model text, but our approach is fundamentally different from previous approaches. One approach that might at first seem similar is Latent Dirichlet Allocation (LDA) (Blei et al., 2002; Blei et al., 2003; Girolami & Kaban, 2003). In the LDA approach the Dirichlet is a distribution over topics, while each topic is modeled in the usual way as a multinomial distribution over words. In our approach, each topic, i.e. each class of documents, is modeled in a novel way by a Dirichlet distribution instead of by a multinomial. Our approach is therefore complementary to the LDA and related approaches.

While the LDA and the DCM models both can be understood as processes that generate documents from distributions of multinomials, they might at first seem similar. The two models never the less address two different challenges associated with text modeling and are fundamentally different. We first take a look at the graphical model representation in Figure 6.5 and 6.5 of the two models, which reveals the major differences.

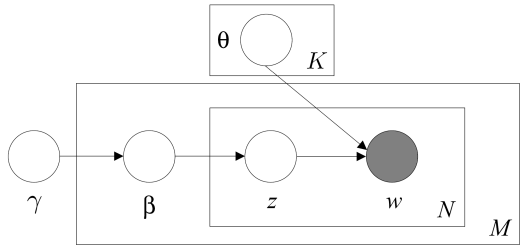


Figure 6.3: Graphical illustration of LDA, modeling a corpus.

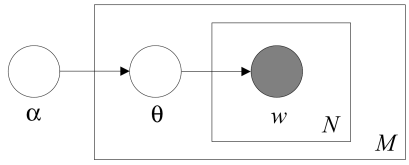


Figure 6.4: Graphical illustration of DCM, modeling one single class.

LDA is a mixture model for a set of K topics, where each individual topic is modeled by one multinomial. Each of the M documents are generated by first selecting a latent topic distribution β, from the distribution of topic distributions γ. For each word of the N words in the document, a topic from β is selected

and the multinomial θ associated with that topic is generating a word. Since the LDA model represents each individual topic as a multinomial, it inherits the issue of incorrect modeling of word burstiness. The LDA model also explicitly allows documents from different classes to share latent topics.

In contrast to the LDA, the DCM is a model for a single topic or class, that is an alternative to the multinomial. Using the DCM approach, each of the M documents are generated by first sampling the class distribution α to generate an unique multinomial model θ for the document. This document model is repeatedly sampled N times to generate the document. The DCM approach allows to generate a variety of multinomial distributions, including those who can generate documents with word bursts.

Since the DCM is an alternative to the multinomial, it could simply be plugged into the LDA, substituting the θ in the graphical model representation in Figure 6.5.

Later in chapter 8, an approximation of the DCM will be integrated in a latent topic framework, similar to the LDA.

6.6 Experiments

For the experiments we use three standard corpora: the so-called industry sector, 20 newsgroups and Reuters-21578 document collections. We compare the DCM method against the multinomial model as well as against recent heuristically improved versions of the multinomial method, which perform as well as discriminative methods (Rennie et al., 2003). We have made every effort to reproduce previous results in order to ensure that a fair comparison is made.

Documents are preprocessed and count vectors are extracted using the Rainbow toolbox (McCallum, 1996). The 500 most common words are removed from the vocabulary to ensure that our results are comparable with previous results. The Dirichlet toolbox (Minka, 2003) is used to estimate the parameters of the DCM model.

When using DCM or multinomial models for classification, we apply Bayes' rule $p(y|x) = p(x|y)p(y)/p(x)$ with a uniform prior $p(y)$ over the classes, following (Rennie et al., 2003). The class y with the highest probability $p(y|x)$ is the predicted label.

6.6.1 Heuristics to improve multinomial models

Various heuristics have been applied to the multinomial and related models to enhance classification performance (Rennie et al., 2003). We briefly review them here for completeness. The first heuristic is the log-transformation of the data, which has been shown to mitigate problems caused by burstiness. In the tables of results below, L is shorthand for the transformation

$$x_{dw}^{\log} = \log(1 + x_{dw}) \tag{6.7}$$

where all logarithms are natural, i.e. base e. One traditional information retrieval heuristic is the term-frequency inverse-document-frequency (TFIDF) transformation, which exists in various forms (Aizawa, 2003; Greiff, 1998).The version used here includes the log-transformation:

$$x_{dw}^{\text{tfidf}} = \log(1 + x_{dw}) \log \frac{D}{\sum_{d'=1}^{D} \delta_{d'w}} \tag{6.8}$$

where δ_{dw} is 1 if word w is present in document d. After the TFIDF transformation, document vectors are L_2-normalized:

$$x_{dw}^{\text{norm}} = \frac{x_{dw}^{\text{tfidf}}}{\sqrt{\sum_{w'=1}^{W} {x_{dw}^{\text{tfidf}}}^2}}. \tag{6.9}$$

This makes all document vectors have the same length and therefore the same amount of influence on the model parameters. The combination of TFIDF and L_2-normalization is denoted TW-L below.

A couple of additional heuristics are also applied. The most important is complement modeling (Rennie et al., 2003): the model for a class is trained with all the documents that do not belong to that class. In most cases, by using other classes' data, substantially more data is available for parameter estimation resulting in better modeling of each class:

$$\hat{\theta}_{kw}^{\text{comp}} = \frac{\sum_{i:y_{dk} \neq 1} x_{dw}^{\text{norm}} + \varepsilon}{\sum_{w'=1}^{W} \sum_{i:y_{dk} \neq 1} x_{dw'}^{\text{norm}} + \varepsilon} \tag{6.10}$$

where the class variable y_{dk} equals 1 if document d belongs to class k and 0 otherwise, and ε is a smoothing constant, which is necessary to prevent probabilities for unseen words from becoming zero. Typically, $\varepsilon = 1$, but this value is often much too large, so we also report results below with $\varepsilon = 0.01$. Non-complement models are smoothed in a similar way. DCM models are also smoothed, but differently, by adding a small constant to each parameter α_w. This constant equals 0.01 times the smallest non-zero estimated α_w.

If documents can belong to more than one class, the usual approach is a one-versus-all-but-one classifier. The complement model is the same as the standard model, in these cases. For this reason, the complement model is defined differently for multi-label problems as an all-versus-all-but-one classifier (Rennie et al., 2003, Appendix A).

Finally, the model parameters are log-normalized:

$$\hat{\theta}_{kw}^{\text{norm}} = \frac{\log \hat{\theta}_{kw}^{\text{comp}}}{\sum_{w'=1}^{W} \log \hat{\theta}_{kw'}^{\text{comp}}} \tag{6.11}$$

making the influence of common words smaller (Rennie et al., 2003). The letter C below denotes complement modeling combined with log-normalization of parameters.

The heuristics described above, and others commonly used with multinomial models for text, modify both input data (word counts) and distribution parameters. Therefore, they do not give probability distributions that are properly normalized, i.e. that sum to one appropriately.

6.6.2 Document collections

The industry sector[1] data set contains 9555 documents distributed in 104 classes. The data set has a vocabulary of 55,055 words, and each document contains on average 606 words. The data are split into halves for training and testing. The 20 newsgroups[2] data set contains 18,828 documents belonging to 20 classes. This collection has a vocabulary of 61,298 words with an average document length of 116 words. The data are split into 80/20 fractions for training and testing. In the industry and newsgroup data sets each document belongs to one class only.

[1] `www.cs.umass.edu/~mccallum/code-data.html`
[2] `people.csail.mit.edu/people/jrennie/20Newsgroups`

The Reuters-21578[3] data set contains 21,578 documents. We use the Mod Apte split which only contains 10,789 documents (Apte et al., 1994), those in the 90 classes with at least one training and one test example. The Mod Apte split uses a predefined set of 7,770 training documents and 3,019 test documents. The documents are multi-labeled and can belong to one or more of the 90 classes. This collection has a vocabulary of 15,996 words and the documents have an average length of 70 words.

6.6.3 Perplexity results

We start by evaluating the perplexity of alternative models over the same test data (Blei et al., 2003). When a document is represented as a vector of word counts, it's probability includes a factor $n!/\prod_{w=1}^{W} x_w!$ that measures how many word sequences could generate the same vector of counts. We define perplexity over a set of D documents as

$$\exp\left(\frac{-\sum_{d=1}^{D}\sum_{w=1}^{W}\log p(x_{dw})}{\sum_{d=1}^{D} n_d}\right) \tag{6.12}$$

where $p(x)$ does not include the factor $n_d!/\prod_{w=1}^{W} x_{dw}!$. Perplexity on test data measures how well a model predicts unseen data. A lower value indicates better prediction.

The perplexity measure is calculated for the 20 newsgroups data, with one model trained for each of the 20 classes. The perplexity for multinomial models is 5311 ± 755 versus 2609 ± 382 for DCM models, where both results are means $\pm$ one standard deviation calculated over 10 random splits. We do not report perplexity results for heuristically modified multinomial models, since the transformed parameters and data no longer define a proper probability distribution that sums to one.

6.6.4 Classification results

The performance of the models is compared on the industry and newsgroup collections using precision $\frac{TP}{TP+FP}$ to measure the accuracy of classification. Here TP is the number of true positives, FP the number of false positives, and FN the number of false negatives.

[3] `kdd.ics.uci.edu`

With multi-labeled data, it is necessary to consider both precision and recall $\frac{TP}{TP+FN}$ to get a fair measure of performance. This is the case for the Reuters data. Following previous work, we combine precision and recall by computing the "break-even" point where precision equals recall. This point can be defined using either micro or macro averaging:

$$BE_{micro} = \frac{1}{N}\sum_{k=1}^{K} N_k \frac{TP_k}{TP_k + FP_k} \tag{6.13}$$

$$BE_{macro} = \frac{1}{K}\sum_{k=1}^{K} \frac{TP_k}{TP_k + FP_k} \tag{6.14}$$

where K is the number of document classes, N is the number of documents and N_k is the number of documents in class k. It is not always possible to get exactly the same value for precision and recall, so the average between the two measures is used in these cases. Using micro-averaging, every document is considered equally important. The macro-averaging measure penalizes classifiers that have poor performance on documents from rare classes.

We acknowledge that precision and break-even may not be the best measures of the effectiveness of a text classifier (Sebastiani, 2002), but we use these measures here for comparability with previous work. The results in Tables 1 and 2 are averages over 10 random splits (50/50 for industry sector and 80/20 for 20 newsgroups), shown $\pm$ one standard deviation σ over the 10 splits.

Table 6.1 shows the performance of the different algorithms on the industry sector data set. Our results using multinomial-based methods are similar to those reported by Rennie et al. (2003) and McCallum and Nigam (1998). Smoothing with $\varepsilon = 0.01$ is clearly better than with $\varepsilon = 1$ for non-complement models. The DCM model produces results that are better than the multinomial and the complement-DCM produces results similar to the multinomial with all heuristics applied.

The results in Table 6.2 are obtained using the 20 newsgroups data. As in the industry sector data, the DCM model outperforms the multinomial. In this corpus, each class is represented by many examples, so complement modeling is not as useful and the DCM and complement-DCM models perform similarly to the best multinomial with heuristics. We show results with $\varepsilon = 0.01$ only because results with $\varepsilon = 1$ are worse, as in the industry sector data, and for compatibility with Rennie et al. (2003).

Table 6.1: Classification results for the industry sector collection.

Method	Smoothing ε	Precision $\pm \sigma$
M	1	0.600 ± 0.011
L-M	1	0.654 ± 0.009
M	0.01	0.783 ± 0.008
DCM		0.806 ± 0.006
L-M	0.01	0.812 ± 0.005
TW-L-M	1	0.819 ± 0.004
TW-L-M	0.01	0.868 ± 0.005
C-M	1	0.889 ± 0.006
C-M	0.01	0.889 ± 0.004
C-L-M	0.01	0.899 ± 0.005
C-L-M	1	0.912 ± 0.005
C-DCM		0.917 ± 0.004
C-TW-L-M	0.01	0.919 ± 0.005
C-TW-L-M	1	0.921 ± 0.004

Table 6.2: Classification results for the 20 newsgroups collection.

Method	Smoothing ε	Precision $\pm \sigma$
M	0.01	0.853 ± 0.004
DCM		0.862 ± 0.005
L-M	0.01	0.865 ± 0.005
TW-L-M	0.01	0.876 ± 0.005
C-M	0.01	0.876 ± 0.005
C-L-M	0.01	0.886 ± 0.005
C-DCM		0.892 ± 0.004
C-TW-L-M	0.01	0.893 ± 0.005

Table 6.3 shows results on the Reuters corpus, which is special in that documents contain few words, and many classes only contain a few documents. The DCM and C-DCM methods still perform well. Standard deviations are not given since there is a single standard training set/test set split for this corpus.

We can evaluate the statistical significance of the differences in performance for the industry sector and 20 newsgroups collections. On these two collections, the DCM model outperforms the standard multinomial and a Student's t-test shows that this difference is extremely significant. The complement-DCM model performs slightly worse than the multinomial model with all heuristics applied. A t-test shows that for both data sets, the differences in performance between the complement-DCM model and C-TW-L-M method are not statistically significant.

Table 6.3: Classification results for the Reuters collection. The third column shows macro break-even, while the last column shows micro break-even.

Method	Smoothing ε	Macro BE	Micro BE
M	1	0.268	0.761
L-M	1	0.303	0.756
DCM		0.359	0.740
TW-L-M	1	0.390	0.768
M	0.01	0.405	0.741
L-M	0.01	0.407	0.759
TW-L-M	0.01	0.456	0.753
C-TW-L-M	0.01	0.560	0.732
C-L-M	0.01	0.562	0.759
C-M	1	0.563	0.759
C-L-M	1	0.594	0.764
C-M	0.01	0.607	0.776
C-DCM		0.624	0.823
C-TW-L-M	1	0.657	0.840

6.7 Discussion

We have argued that the Dirichlet compound multinomial (DCM) model is a more appropriate generative model for text documents than the traditional multinomial model. The reason is that a DCM can model burstiness: the phenomenon that if a word appears once, it is more likely to appear again.

We have shown experimentally that the DCM model performs better than the multinomial model for two standard text mining tasks. First, as measured by perplexity, the DCM models a single collection of documents better. Second, when documents are classified using Bayes' rule where a generative model is used for each of the alternative classes, the accuracy using a DCM model is higher than when using a multinomial model. When the most effective known heuristics are applied in addition, accuracy using multinomial models versus using DCM models is similar.

The DCM model is a generative model for the documents within a class. Given a Dirichlet distribution, a document is not generated directly. Instead, the Dirichlet is used to generate a multinomial; this multinomial is then used to generate the document. Conceptually, different documents within the same class are generated by different multinomial distributions. This procedure allows for diversity within the class. The words that are bursty in a particular document are those that have high probability in the particular multinomial used to generate

this document.

A DCM model can represent a topic (i.e. a class of documents) where different documents use alternative terminology. For example, some automotive documents may use the word "hood" while others use the word "bonnet." This within-topic diversity is different from the within-document diversity allowed by latent topic modeling, where each topic is represented by a single multinomial, but each word in a document may be generated by a different topic (Deerwester et al., 1990; Hofmann, 1999; Blei et al., 2003).

Although we here have focused on correctly modeling word counts in documents, the DCM is applicable in other domains as well where burstiness occurs, like e.g. (Kvam & Day, 2001). The results here will therefore be useful outside the domain of modeling documents.

6.8 Summary

We have shown that the DCM is capable of modeling the phenomenon word burstiness correctly in contrast to the commonly used multinomial distribution. By compounding the Dirichlet distribution with the multinomial, the DCM is also capable of modeling sparse data, which made the Dirichlet distribution inappropriate for modeling text directly. The DCM uses an extra degree of freedom in the parameter scaling to model this phenomenon. This was verified looking at the DCM model as an Polya urn model, where the dynamics of the DCM were fully revealed. The experiments have shown that the DCM is better at the classification task than the multinomial, and performs similarly to the multinomial with some heuristical changes. Perplexity analysis of the multinomial and DCM reveals that the DCM is a better at modeling the data, having a lower perplexity.

Chapter 7

Approximating The DCM

7.1 Introduction

In the previous Chapter 6 the Dirichlet compound multinomial was introduced, and was demonstrated capable at modeling word burstiness. The DCM is both qualitatively and quantitatively, better than the multinomial model on standard document collections (Madsen et al., 2005).

The DCM model is however not without problems. Though the Dirichlet distribution and the multinomial distribution both are members of the exponential family, the compound model of the two, the DCM, is not a member of the exponential family. Exponential families have many desirable properties (Banerjee et al., 2005), and it is therefore desirable to use functions within the exponential family. Second, because of the relative complexity of the DCM expression, understanding it's behavior qualitatively is difficult. Third, DCM parameters cannot be estimated quickly, i.e there is no closed form solution. When estimating DCM parameters it is necessary to apply gradient descent methods (Minka, 2003), which are costly and slow. Fast training is important not only for modeling large document collections but also for using DCM distributions in more complex mixtures or hierarchical models, such as LDA (Blei et al., 2003).

In this chapter, we present an approximation of the DCM that is in the ex-

ponential family. An exact solution of the maximum likelihood parameters for the approximate distribution is derived. The approximate distribution can be computed efficiently (more than 100 times faster than the DCM), and has a categorization accuracy similar to that of the DCM.

7.2 Approximation

In this section we derive an exponential family approximation of the DCM distribution that we call the EDCM and investigate the qualitative behavior of the EDCM.

We start by taking a look at the DCM from equation 6.1 where we define the new variable s as the sum of the parameters $s = \sum_w \alpha_w$, still keeping in mind the document length $n = \sum_w x_w$.

$$p(x|\alpha) = \frac{n!}{\prod_w^W x_w!} \frac{\Gamma(s)}{\Gamma(s+n)} \prod_{w=1}^{W} \frac{\Gamma(x_w + \alpha_w)}{\Gamma(\alpha_w)}. \tag{7.1}$$

Where the parameter s is controlling the degree of burstiness in the model. When training the DCM model, we find empirically that $\alpha_w << 1$ for practically all words in the vocabulary. For one class of newsgroup articles, the average α_w is 0.004 and out of the 59,826 parameters, 99% are below 0.1, 17 are above 0.5, and only 5 are above 1.0. The α parameters can therefore be regarded as being small.

A useful approximation that can be applied when the α's are small is: $\Gamma(x+\alpha) \approxeq \Gamma(x)\alpha$. We further use $\Gamma(x) = (x-1)!$ when x is an integer. From these approximations we get the EDCM distribution[1] $q(x|\beta)$.

$$q(x|\beta) = n! \frac{\Gamma(s)}{\Gamma(s+n)} \prod_{w:x_w \geq 1} \frac{\beta_w}{x_w} \tag{7.2}$$

For clarity, we have denoted the EDCM parameters β_w. The parameter s is therefore now $s = \sum_w \beta_w$. The $q(x|\beta)$ is not a proper probability distribution,

[1] The EDCM is not a true distribution while the integral over the EDCM is not exactly one.

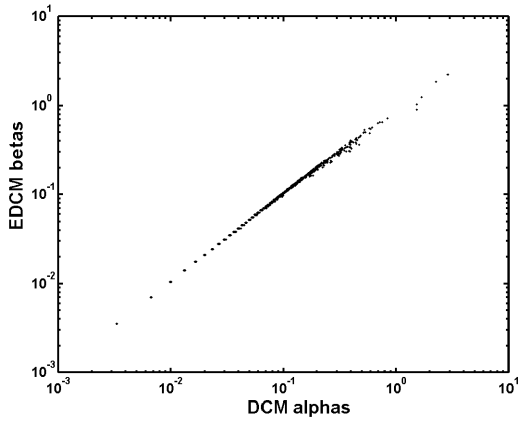


Figure 7.1: Comparison of the α-parameters of the DCM model and the β-parameters of the EDCM model. The parameter values follow a straight line, showing that the two methods of estimation result in almost the same parameter values. Even for the large parameter values, the approximation is quite accurate, though the approximation equations were conditioned on having small α values.

that is it does not sum to one, since we have used the approximation. It is however in principle possible to normalize $q(x|\beta)$ to sum to one, by summing $q(x|\beta)$ over all values of x to get a normalizing constant $Z(\beta)$. This technicality is however not considered here[2]. In practice the values given by Equation 7.2 are very close to those given by Equation 7.1. On a sample set of 4000 documents from 20 different classes $q(x|\alpha)$ is highly correlated with $p(x|\alpha)$. On average, $q(x|\alpha)$ only deviates by 2.2% from $p(x|\beta)$. This high correlation is because the $\Gamma(x+\alpha)/\Gamma(\alpha)$ approximation is highly accurate for small α_w values. For a typical α- vector trained from 800 documents, of the 69,536 non-zero word counts, the approximation is on average 3.9% off. In Figure 7.1 the DCM α-parameters are compared with the EDCM β-parameters. The parameter values are close to be forming a straight line, showing high degree of similarity.

From Equation 7.2 we get some insight about the DCM as well as the EDCM, that was not directly obvious from Equation 7.1. For fixed s and n, the probability of a document is proportional to $\prod_{w:x_w \geq 1} \beta_w / x_w$. This means that the first appearance of a word w reduces the probability of a document by β_w, a word-specific factor that is almost always much less than 1.0, while the m'th appearance of any word reduces the probability by $(m-1)/m$, which tends to

[2]Since $q(x|\beta)$ is not a proper probability distribution, we cannot calculate perplexity or other probabilistic measures that tell how well the EDCM models the data, but the focus is here categorization.

1 as m increases. This behavior reveals how the EDCM, and hence the DCM, allow multiple appearances of the same word to have high probability. In contrast, with a multinomial each appearance of a word reduces the probability by the same factor.

One consideration about the DCM was that it does not belong to the exponential family. We now rewrite Equation 7.2 to the exponential family form. An exponential family distribution has the form $f(x)g(L)\exp[t(x)h(\beta)]$ where $t(x)$ is a vector of "sufficient statistics" and $\theta = h(\beta)$ is the vector of so-called "natural parameters". We can write $q(x|\beta)$ in this form as:

$$q(x|\beta) = \left(\prod_{w:x_w \geq 1} x_w\right) n! \frac{\Gamma(s)}{\Gamma(s+n)} \exp\left[\sum_{w:x_w \geq 1} \beta_w\right] \tag{7.3}$$

For the EDCM distribution, the sufficient statistics for a document x are the normalized data $\langle t_1(x), ..., t_W(x)\rangle$ where $t_w(x) = I(x_w \geq 1)$ and W is the number of words in the vocabulary. The expression also shows that the natural parameters for the EDCM distribution are $\theta_w = \ln \beta_w$.

7.3 Maximum Likelihood Estimation

The maximum likelihood estimate of the EDCM parameters can be determined by taking the derivative of the log-likelihood function. This is in contrast to the complications involved in determining the parameters of the DCM (Minka, 2003). From Equation 7.2 the log-likelihood function can be determined.

$$\mathcal{L}_\beta(x) = \log(n) + \log\left(\Gamma(s+n)\right) + \sum_{w:x_w \geq 1} \log(\beta_w) - \log(x_w) \tag{7.4}$$

Given a set of training documents, we can calculate the partial derivative of the log-likelihood.

$$\frac{d\mathcal{L}_\beta(x)}{d\beta_w} = |D|\Psi(s) - \sum_{d=1}^{D} \Psi(s+n_d) + \frac{I(x_{dw} \geq 1)}{\beta_w} \tag{7.5}$$

Setting the derivative of the log-likelihood equal to zero and solving for the parameters β_w we get Equation 7.6.

$$\beta_w = \frac{\sum_{d=1}^{D} I(x_{dw} \geq 1}{|D|\Psi(s) - \sum_{d=1}^{D} \Psi(s + n_d)} \tag{7.6}$$

Since β_w is part of s, Equation 7.6 is not directly solvable as is. The parameter sum s can be computed though by summing over the words w in Equation 7.6.

$$s = \sum_{w=1}^{W} \beta_w = \frac{\sum_{w=1}^{W} \sum_{d=1}^{D} I(x_{dw} \geq 1}{|D|\Psi(s) - \sum_{d=1}^{D} \Psi(s + n_d)} \tag{7.7}$$

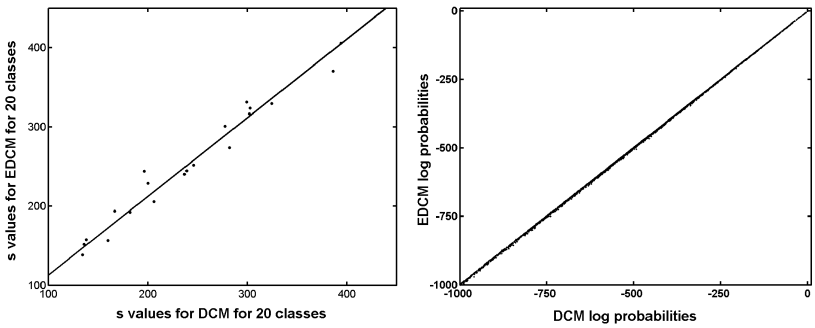
Equation 7.7 can be solved numerically for s efficiently, since it only involves one unknown parameter. Having solved for s, Equation 7.6 is easily solvable.

7.4 Experiments

In Figure 7.2 we start by comparing the parameter sums s of the DCM and EDCM followed by a comparison of the likelihood for the two models. The s-value tells to what extent that burstiness is present in the data. The values are generally close to being the same, revealing that the two models agree on the extent of burstiness. The probability estimates for the two models are also similar, which shows that the approximations used for the EDCM model are accurate.

For classification purposes we have compared the DCM and EDCM with the multinomial model. The multinomial model is smoothed in an optimal way, and the accuracy of the multinomial is therefore higher than in (Rennie et al., 2003).

As we had hoped, the DCM and EDCM model have very similar classification performances. In fact, for 20 newsgroups, the EDCM model actually performs



(a) Sum of DCM α_w versus sum of EDCM β_w

(b) $\mathcal{L}_\alpha(x)$ (DCM) versus $\mathcal{L}_\beta(x)$ (EDCM)

Figure 7.2: Comparing the parameter sums (a) for the DCM and EDCM (degree of burstiness) reveals that the two models agree on the burstiness. In (b) the log-likelihood for the two models is compared for 4000 test documents. The log-likelihood estimated by the approximation is close to the real thing.

Table 7.1: Classification accuracy for the 20 newsgroups and Industry Sector collections, comparing the multinomial, DCM and EDCM. The scores are averages of 10 random splits.

Data Set	Multinomial	DCM	EDCM
20 Newsgroups	0.855	0.862	0.864
Industry Sector	0.789	0.804	0.798

better than the DCM model. This is particularly encouraging considering the EDCM model was almost 150 times faster to train than the DCM model (19 seconds vs. 2788 seconds for a 20 newsgroups split using the fastest DCM fixed point training method). Both the DCM and EDCM use a naive smoothing method and still performs slightly better than the multinomial.

7.5 Summary

The approximated DCM model, the EDCM, has added an additional insight by giving an intuitive explanation for how the DCM models burstiness. The

proposed approximation to the DCM distribution further belongs to the exponential family of distributions and is orders of magnitude faster to train. The estimated EDCM parameters and EDCM approximated probabilities are close to the true DCM values, resulting in similar classification performance for the two models.

Chapter 8

Latent Topic EDCM

8.1 Introduction

In previous chapters we have argued that the DCM and EDCM can be thought of as a within topic distribution. That is a distribution over subtopics within the topic. An example was when modeling the topic "cars", the DCM and EDCM allow for using only parts of the topic, i.e. some documents might use the word "bonnet" a lot while other documents use the word "hood". This model flexibility is also what allows to model burstiness.

There is another aspect in topic modeling that the two models do not capture in their base form, and that is the concept of latent topics. The idea behind latent topics is that different document categories can share latent sub-topics, i.e. the categories "cars" and "oil production" might share the words in the sub-topic "environment". The idea of shared Latent topics, based on the multinomial, has previously been examined in (Hofmann, 1999; Cohn & Hofmann, 2001; Blei et al., 2002; Blei et al., 2003; Girolami & Kaban, 2003), where the conclusion is that latent topic models generally result in better modeling. Latent topic models are further believed to reduce problems with synonymy and polysemy (Deerwester et al., 1990; Larsen et al., 2002). Since the DCM and EDCM are alternatives to the multinomial, we here experiment with a latent topic model based on the EDCM. We have chosen the EDCM rather than the DCM

because of the higher complexity of the DCM, which combined in a latent topic model only will get further complicated. The latent topic model considered is an unsupervised clustering model but could easily be transformed to a supervised model.

8.2 LTEDCM Model

The Latent Topic EDCM (LTEDCM) is a generative probabilistic mixture model. The EDCM model generates a corpora of document x, indexed by d, by taking the following steps. At first, for each of the D documents, a topic mixture π is determined. The K topics, each represented by a set of α-parameters are then blended into a new set of parameters ϕ_d, where each topic is weighted by the mixture coefficient π_k belonging to topic k. The document is then generated by drawing n_d words from the model of mixed topics.

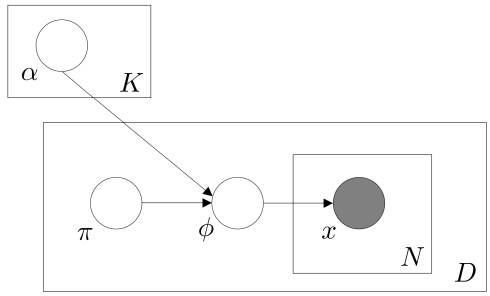


Figure 8.1: Graphical illustration of LTEDCM model. A document is generated by first drawing a topic mixture distribution π. The topic parameters α are then mixed into a new set of parameters ϕ where each topic is weighted π_k. The document is finally generated, drawing n words from the exponential DCM distribution, parameterized with ϕ.

The likelihood for a single document using the LTEDCM generative model is determined directly from the graphical model in Figure 8.2,

$$p(x|\alpha,\pi) = \int p(x|\phi)p(\phi|\alpha,\pi)d\phi \approx p(x|\phi')p(\phi'|\alpha,\pi) \tag{8.1}$$

$$\phi' = \arg\max_{\phi} p(x|\phi)p(\phi|\alpha,\pi) \tag{8.2}$$

where the mixture coefficients π's for a single document d are constrained to sum to one, $\sum \pi_d = 1$. The document generation parameters ϕ_d are mixed by simple summation $\phi_{d,w} = \sum_k \pi_{k,d}\alpha_{k,w}$, where k is the topic index and w is the vocabulary index. Letting $p(x_d|\phi_d)$ be the EDCM, we get,

$$p(x_d|\alpha, \pi_d) = \frac{n!\Gamma(s_d)}{\Gamma(s_d + n_d)} \prod_{w=1}^{W} \left(\frac{\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w}}{x_{d,w}} \right)^{I(x_{d,w}>0)} \tag{8.3}$$

where $\phi_{d,w} = \sum_k \pi_{d,k}\alpha_{k,w}$ has been used to further simplify the equation. The length of document d is denoted n_d and $s_d = \sum_k \sum_w \pi_k \alpha_{k,w}$. The function $I(x_{d,w} > 0)$ is one if the word w appears once or more in the document and zero otherwise.

We define the log likelihood $\mathcal{L}_{\alpha,\pi}$ for the set of D documents as log to the product over the single document probabilities.

$$\mathcal{L}_{\alpha,\pi} = \ln \prod_{d=1}^{D} p(x_d|\alpha, \pi_d) = \ln \prod_{d=1}^{D} \frac{n!\Gamma(s_d)}{\Gamma(s_d + n_d)} \prod_{w=1}^{W} \left(\frac{\sum_{k=1}^{K} \pi_k \alpha_{k,w}}{x_w} \right)^{I(x_w>0)} \tag{8.4}$$

8.3 Maximum Likelihood Estimation

We first propose to use the maximum likelihood approach to determine the parameters of the LTEDCM model. The maximum likelihood approach might overfit the parameters to the training data making it less suitable for inference but is never the less often easy and fast to estimate. Since the parameters and latent variables are dependent, we can't determine the maximum likelihood solution in one shot. We instead use an iterative approach where the parameters are estimated first and then the latent variables are estimated. This iterative process continues until the likelihood has converged. This is conceptually similar to the approach suggested in (Hansen & Larsen, 1998).

We start out estimating the parameters of the model. The gradient for the

parameters is determined by taking the derivative to the log likelihood $\mathcal{L}_{\alpha,\pi}$ with respect to parameters α and setting the expression equal to zero,

$$\begin{aligned}\frac{d\mathcal{L}_{\alpha,\pi}}{d\alpha_{k',w'}} &= 0 = \frac{d}{d\alpha_{k',w'}} \ln \prod_{d=1}^{D} \frac{n_d!\Gamma(s_d)}{\Gamma(s_d+n_d)} \prod_{w=1}^{W} \left(\frac{\sum_{k=1}^{K} \pi_{d,k'}\alpha_{k,w}}{x_w} \right)^{I(x_{d',w}>0)} \\ &= \sum_{d=1}^{D} \pi_{d,k'} \left(\psi(s_d) - \psi(s_d+n_d) + \frac{I(x_{d,w'}>0)}{\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w'}} \right) \end{aligned} \tag{8.5}$$

where there is no need for lagrange multipliers since the α parameters are unconstrained.

Solving equation (8.5) for $\alpha_{k',w'}$ we first split the equation,

$$\sum_{d=1}^{D} \left(\psi(s_{d'}+n_{d'}) - \psi(s_{d'})\right) \pi_{d,k'} = \sum_{d=1}^{D} \frac{I(x_{d,w'}>0)\pi_{d,k'}}{\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w'}} \tag{8.6}$$

whereafter we multiply with $\alpha_{k',w'}$ on sides and sum over k,

$$\sum_{d=1}^{D} \left(\psi(s_{d'}+n_{d'}) - \psi(s_{d'})\right) \sum_{k'=1}^{K} \pi_{d,k'}\alpha_{k',w'} = \sum_{d=1}^{D} I(x_{d,w'}>0) \tag{8.7}$$

solving for $\alpha_{k',w'}$ leaves us with:

$$\alpha_{k',w'} = \frac{\sum_{d=1}^{D} I(x_{d,w'}>0) - \left(\psi(s_{d'}+n_{d'}) - \psi(s_{d'})\right) \sum_{k=1/k'}^{K} \pi_{d,k}\alpha_{k,w'}}{\sum_{d=1}^{D} \left(\psi(s_{d'}+n_{d'}) - \psi(s_{d'})\right) \pi_{d,k'}} \tag{8.8}$$

Since the update equation for $\alpha_{k',w'}$ is dependent on the other α's, the updates have to be performed iteratively. This equation is however not guaranteed to

converge to the correct solution. We therefore propose another solution for finding the α parameters. With the other approach we start by determining s_d from equation (8.6), by first multiplying with $\alpha_{k',w'}$ and then by summing over first k' and then w' giving us,

$$\sum_{d=1}^{D} \left(\psi(s_{d'} + n_{d'}) - \psi(s_{d'})\right) s_d \quad = \quad \sum_{d=1}^{D} \sum_{w'=1}^{W} I(x_{d,w'} > 0) \tag{8.9}$$

where we constraint the equation to be true for all d. By constraining equation (8.9) we ensure that the modeling energy from document d is preserved for modeling exactly that document. The constraint might be inexpedient for the model but makes the following update rules for the parameters simple. After constraining equation (8.9), the estimation of s_d can be performed iteratively,

$$s_d \quad = \quad \frac{\sum_{w'=1}^{W} I(x_{d,w'} > 0)}{\psi(s_{d'} + n_{d'}) - \psi(s_{d'})} \tag{8.10}$$

in a similar manner as in the previous chapter.

The other solution for finding the α parameters is determined by starting out from equation (8.5), where we again assume that the solution to the equation is valid for for all documents d, leading to the following solution.

$$\begin{aligned} \pi_{d,k} \left(\psi(s_d + n_d) - \psi(s_d)\right) \quad &= \quad \frac{\pi_{d,k} I(x_{d,w} > 0)}{\sum_{k=1}^{K} \pi_{d,k} \alpha_{k,w}} \\ \pi_{d,k} \left(\psi(s_d + n_d) - \psi(s_d)\right) \sum_{k=1}^{K} \pi_{d,k} \alpha_{k,w'} \quad &= \quad \pi_{d,k} I(x_{d,w} > 0) \end{aligned} \tag{8.11}$$

Equation (8.11) above, gives $D \times W$ equations that must be satisfied for the α parameters. By multiplying with all the K $\pi_{d,k}$'s instead of just a single one, we reduce to $K \times W$ equations,

$$\begin{aligned} \Pi^T \mathbf{diag}\left(\Psi_{s+n} - \Psi_s\right) \Pi \mathcal{A} \quad &= \quad \Pi I \\ \mathcal{A} \quad &= \quad \left(\Pi^T \mathbf{diag}\left(\Psi_{s+n} - \Psi_s\right) \Pi\right)^{-1} \Pi I \end{aligned} \tag{8.12}$$

where $\mathcal{A}$ is a $K \times W$ matrix with the elements $\alpha_{k,w}$ and Π is a $D \times K$ matrix with the elements $\pi_{d,k}$ and I is a $D \times W$ matrix with the elements $I(x_{d,w} > 0)$ and $\Psi_{s+n} - \Psi_s$ is a vector of length D containing the elements $\psi(s_d + n_d) - \psi(s_d)$.

We find the gradient direction of the latent variables by taking the derivative to the log likelihood $\mathcal{L}_{\alpha,\pi}$ with respect to mixing coefficients π and setting the expression equal to zero,

$$\frac{d\mathcal{L}_{\alpha,\pi}}{d\pi_{d',k'}} = 0 = \frac{d}{d\pi_{d',k'}} \ln \prod_{d=1}^{D} \frac{n!\Gamma(s_d)}{\Gamma(s_d + n_d)} \prod_{w=1}^{W} \left(\frac{\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w}}{x_w} \right)^{I(x_{d',w}>0)} \tag{8.13}$$

$$= (\psi(s_{d'}) - \psi(s_{d'} + n_{d'})) \left(\sum_{w=1}^{W} \alpha_{k',w} \right) \tag{8.14}$$

$$+ \sum_{w=1}^{W} \frac{I(x_{d',w} > 0)}{\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w}} \alpha_{k',w}$$

where the lagrange multiplier $\sum_{d=1}^{D} \lambda_d (1 - \sum_{k=1}^{K} \pi_{d,k})$ must be added to equation (8.13) to ensure that the constraint $\sum_{k=1}^{K} \pi_{d,k} = 1$ is satisfied. The lagrange multiplier has been omitted from the equation for simplicity. In equation (8.14) the derivative to the lagrange multiplier $-\lambda_{d'}$ is again omitted for simplicity.

The lagrange multiplier $\lambda_{d'}$ can be determined by first multiplying equation (8.14) with $\pi_{d',k'}$ and then sum over the number of topics k,

$$\lambda_{d'} = (\psi(s_{d'}) - \psi(s_{d'} + n_{d'}))\, s_d + \sum_{w=1}^{W} I(x_{d',w} > 0) \tag{8.15}$$

where the derivative to the log likelihood $\mathcal{L}_{\alpha,\pi}$ leads to the following solution.

$$\begin{aligned}\frac{d\mathcal{L}_{\alpha,\pi}}{d\pi_{d',k'}} &= (\psi(s_{d'}) - \psi(s_{d'} + n_{d'}))\left(-s_d + \sum_{w=1}^{W} \alpha_{k',w}\right) \\ &+ \sum_{w=1}^{W} I(x_{d',w} > 0) \frac{\alpha_{k',w} - \sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w}}{\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w}} \end{aligned} \tag{8.16}$$

The solution for the latent variables can not be determined as elegantly as the parameters. Instead a gradient descent approach has to be selected to find the maximum likelihood with respect to the latent variables. We choose to fit a polynomial to the likelihood using line-search to find the likelihood optimum, given the parameters. The algorithm for training and inference would take the steps in Table 8.3.

1. Initialize the α parameters to the mean of $1/k$'th of the documents.
2. Initialize the latent variables π to uniformly distributed random values.
3. Iteratively estimate s, α and π until the likelihood for the D training documents has converged, using equation (8.9), (8.12) and (8.16).
4. Estimate the latent topic distribution for the test documents using equation (8.16).
5. Determine class associations for the test documents using the k-nearest-neighbors (kNN) algorithm.

Table 8.1: Algorithm for inference using the maximum likelihood LTEDCM model.

A classification algorithm is here not directly built into the model but instead the kNN algorithm is used to classify documents based on the topic representation. Other classification algorithms like e.g. neural networks might be applied as well. Another approach would be to train a model for each class of documents. The idea of having inter-class latent topics would not be possible using the latter approach though.

8.4 Expectation Maximization

The maximum likelihood approach finds the most likely model for the data on which it was trained. When inference on unseen documents are considered, the ML solution might not be the best approach, while it is likely to overfit the model to the training data. We here apply the generally known Expectation Maximization (EM) Algorithm (Dempster et al., 1977; Dellaert,), which is an iterative procedure to determine the parameters of latent variable models. The EM-Algorithm alternates between an E-step and a M-step, where the E-step finds a distribution of latent variable given the parameters, and the M-step following optimizes the likelihood with respect to the parameters. The iterative concept of the expectation maximization algorithm is a conceptual similar approach to the maximum likelihood approach discussed in the previous section. The main difference between the two approaches is that a distribution of solutions for the latent variable π is considered using the EM approach rather than just the maximum likelihood solution. When a distribution of solutions for the latent variables is considered, overfitting is less likely to occur.

Using the EM approach we first consider the joint distribution.

$$p(x, \pi|\alpha) = p(\pi|x, \alpha)p(x|\alpha) \tag{8.17}$$

We then integrate out the latent variables finding the marginal likelihood for a single document,

$$p(x|\alpha) = \int p(x, \pi|\alpha)d\pi = \int p(\pi|x, \alpha)p(x|\alpha)d\pi \tag{8.18}$$

where all likely configurations of the latent variable is considered.

The integral in eq. 8.18 is generally intractable for a direct maximum likelihood solution. We instead turn to the EM Algorithm (Dempster et al., 1977), to estimate the model parameters. In the second line of equation (8.20) the integral is one, showing that we are basically just finding $p(x_d|\alpha)$. To make use of the EM-Algorithm we introduce an auxiliary distribution over the latent variable $q_{\pi_d}(\pi_d|x_d, \alpha)$ in third line. The auxiliary distribution introduces a lower bound for the log-likelihood $\mathcal{L}_\alpha$.

$$
\begin{aligned}
\mathcal{L}_\alpha &= \ln \prod_{d=1}^{D} \int p(x_d, \pi_d|\alpha) d\pi \\
&= \ln \prod_{d=1}^{D} \int p(\pi_d|x_d, \alpha) d\pi \; p(x_d|\alpha) \\
&= \sum_{d=1}^{D} \ln \int q_{\pi_d}(\pi_d|x_d, \alpha) \frac{p(x_d, \pi_d|\alpha)}{q_{\pi_d}(\pi_d|x_d, \alpha)} d\pi \\
&\geq \sum_{d=1}^{D} \int q_{\pi_d}(\pi_d|x_d, \alpha) \ln \frac{p(x_d, \pi_d|\alpha)}{q_{\pi_d}(\pi_d|x_d, \alpha)} d\pi \qquad (8.19) \\
&= \sum_{d=1}^{D} \int q_{\pi_d}(\pi_d|x_d, \alpha) \ln p(x_d|\alpha) d\pi \\
&- \sum_{d=1}^{D} \int q_{\pi_d}(\pi_d|x_d, \alpha) \ln \frac{q_{\pi_d}(\pi_d|x_d, \alpha)}{p(\pi_d|x_d, \alpha)} d\pi \\
&= \sum_{d=1}^{D} \int q_{\pi_d}(\pi_d|x_d, \alpha) \ln p(x_d, \pi_d|\alpha) \\
&- \sum_{d=1}^{D} \int q_{\pi_d}(\pi_d|x_d, \alpha) \ln q_{\pi_d}(\pi_d|x_d, \alpha) d\pi \\
&\equiv \mathcal{F}(q_{\pi_1}(\pi_1), ..., q_{\pi_D}(\pi_D), \alpha)
\end{aligned}
$$

The auxiliary distribution $q_{\pi_d}(\pi_d|x_d, \alpha)$ is allowing us make use of the Jensen inequality (Jensen, 1906). Using Jensen's inequality a lower bound $\mathcal{F}$ on the likelihood appears. On line 5 we see that bound consist of two parts, the negative Kullbach Leibler distance (Kullbach & Leibler, 1951) between $q_{\pi_d}(\pi_d|x_d, \alpha)$ and $p(\pi_d|x_d, \alpha)$ and another expression $q_{\pi_d}(\pi_d|x_d, \alpha) \ln p(x_d|\alpha)$. The KL distance has previously been used for document categorization (Bigi, 2003).

The expectation step (E-step) maximizes the bound $\mathcal{F}$ with respect to each of the distributions $q_{\pi_d}(\pi_d|x_d, \alpha)$ and the following maximization (M-step) maximizes the bound using the latent variable statistics determined from the E-step,

$$
q_{\pi_d}^{(t)}(\pi_d) = \arg\max_{q_{\pi_d}} \mathcal{F}(q_{\pi_1}^{(t-1)}(\pi_1), ..., q_{\pi_D}^{(t-1)}(\pi_D), \alpha^{(t-1)}) \qquad (8.20)
$$

$$
\alpha^{(t)} = \arg\max_{\alpha} \mathcal{F}(q_{\pi_1}^{(t)}(\pi_1), ..., q_{\pi_D}^{(t)}(\pi_D), \alpha^{(t-1)}) \qquad (8.21)
$$

where the t in the superscript indicates time-step or iteration number for the EM algorithm.

Maximizing the bound with respect to the auxiliary distribution in equation (8.20) can be determined from line 5 in equation (8.20). The bound is here maximized when the Kullbach Leibler (KL) distance is minimized, while the first part of the equation is independent of the actual distribution over π_d. The KL distance is maximized setting $q^t_{\pi_d}(\pi_d|x_d,\alpha)$ equal to $p(\pi_d|x_d,\alpha(t-1))$. Substituting this solution into the bound equation (8.20), the bound $\mathcal{F}$ actually becomes equal to the likelihood $\mathcal{L}$ (Beal, 2003). The solution can also be found by taking the derivative to the bound $\mathcal{F}$ with respect to the auxiliary function $q_{\pi_d}(\pi_d|x_d,\alpha)$ and setting it equal to zero.

Given a set of parameters, the auxiliary distribution $q_{\pi_d}(\pi_d)$ can be found by taking the derivative of the bound $\mathcal{F}$ with respect to the hyper parameters controlling the auxiliary distribution. The expression is then set equal to zero, tightening the bound. We here just parameterize the auxiliary distribution with the parameter β without explaining it further to show the concept.

$$\frac{d\mathcal{F}}{d\beta_{d,k}} = 0 = \sum_{d=1}^{D} \int \frac{dq_{\pi_d}(\pi_d)}{d\beta_{d,k}} \ln \frac{p(x_d,\pi_d|\alpha)}{q_{\pi_d}(\pi_d|x_d,\alpha)} - q_{\pi_d}(\pi_d) \frac{d\ln q_{\pi_d}(\pi_d)}{d\beta_{d,k}} d\pi_d \quad (8.22)$$

The maximization of the bound $\mathcal{F}$ with respect to the parameters can be determined from line 6 in equation (8.20), where the auxiliary distribution is kept constant. The maximum is found by setting the derivative to the equation equal to zero,

$$\frac{d\mathcal{F}}{d\alpha_{k,w}} = 0 = \frac{d}{d\alpha_{k,w}} \sum_{d=1}^{D} \int q_{\pi_d}(\pi_d|x_d,\alpha) \ln p(x_d,\pi_d|\alpha)\pi_d \quad (8.23)$$

where the last part of the equation has been omitted while it is independent of the parameters. Using an approach similarly to that of equation (8.5), we end up with the following result,

$$\alpha_{k,w} = \frac{\sum_{d=1}^{D} I(x_{d,w} > 0) - \int q_{\pi_d}(\pi_d)\psi_{sns} \sum_{k'=1/k}^{K} \pi_{d,k'}\alpha_{k',w} d\pi_d}{\sum_{d=1}^{D} \int q_{\pi_d}(\pi_d)\psi_{sns}\pi_{d,k} d\pi_d} \tag{8.24}$$

where ψ_{sns} is shorthand for $(\psi(s_d + n_d) - \psi(s_d))$ and $q_{\pi_d}(\pi_d)$ is shorthand for $q_{\pi_d}(\pi_d|x_d, \alpha)$. The parameters of equation (8.24) are again mutually dependent, which might cause a convergence problem. If the mutual dependence in equation (8.24) result in convergence problems, a line-search approach using the gradient,

$$\frac{d\mathcal{F}}{d\alpha_{k,w}} = \sum_{d=1}^{D} \int q_{\pi_d}(\pi_d)\ \pi_{d,k} \left(\frac{I(x_{d,w} > 0)}{\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w}} - \psi_{sns} \right) d\pi_d \tag{8.25}$$

can be applied instead.

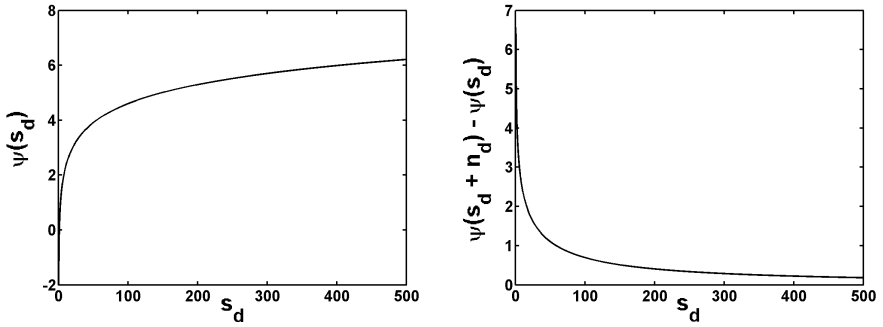
The integrals in equation (8.22), (8.24) and (8.25) are in general intractable.

8.5 Mean Field Approximation

We here apply a variational mean field approximation for determining the bound gradient in the equations (8.25) and (8.22). Mean field approximation covers the approach of determining the untractable integrals by assuming that the auxiliary distribution is factorizable, where we constrain the auxiliary distribution to belong to a given class of distributions.

We start out approximating the integral by keeping ψ_{sns} fixed. The part ψ_{sns} makes the integral hard to compute, while at the same time only posing small changes to the result. From (Madsen et al., 2005) we know that the burstiness in general for text documents results in s values in the range $300 - 500$. In Figure 8.2(a), the derivative of ψ is close to constant within the burstiness range, resulting in small changes in ψ_{sns}, see Figure 8.2(b), if we calculate the full integral.

Another part of the integral in equation (8.25) that is hard to deal with is the



(a) $\psi(s_d)$ (b) $\psi(s_d + n_d) - \psi(s_d)$

Figure 8.2: The derivative of the ψ_s function is more or less constant within the burstiness range ($s_d = 300 - 500$) for document collections. The value of ψ_{sns} is therefore close to constant (we here use $n_d = 100$).

expression $\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w}$ in the denominator. This part is therefore also assumed constant, introducing a small error.

The bound gradient in equation (8.25) is now approximated with simpler expression containing a much simpler integral.

$$\frac{d\mathcal{F}}{d\alpha_{k,w}} \approx \sum_{d=1}^{D} \left(\frac{I(x_{d,w} > 0)}{\sum_{k=1}^{K} \pi_{d,k}\alpha_{k,w}} - \psi_{sns} \right) \int q_{\pi_d}(\pi_d)\ \pi_{d,k}\ d\pi_d \tag{8.26}$$

Using the mean field approximation, we assume that the auxiliary distribution $q_{\pi_d}(\pi_d)$ can be factorized,

$$q_{\pi_d}(\pi_d) = \prod_{k=1}^{K} q_{\pi_{d,k}}(\pi_{d,k}) \tag{8.27}$$

leaving us with the mean field approximated α gradient,

$$\begin{aligned}
\frac{d\mathcal{F}}{d\alpha_{k',w'}} &\approx \sum_{d=1}^{D}\left(\frac{I(x_{d,w'}>0)}{\sum_{k=1}^{K}\pi_{d,k}\alpha_{k,w'}}-\psi_{sns}\right)\int\left(\prod_{k=1}^{K}q_{\pi_{d,k}}(\pi_{d,k'})\right)\pi_{d,k}d\pi_d \\
&= \sum_{d=1}^{D}\left(\frac{I(x_{d,w'}>0)}{\sum_{k=1}^{K}\pi_{d,k}\alpha_{k,w}}-\psi_{sns}\right)\prod_{k=1}^{K}\int q_{\pi_{d,k}}(\pi_{d,k})\pi_{d,k'}d\pi_{d,k} \\
&= \sum_{d=1}^{D}\left(\frac{I(x_{d,w'}>0)}{\sum_{k=1}^{K}\pi_{d,k}\alpha_{k,w'}}-\psi_{sns}\right)\langle q_{\pi_{d,k}}(\pi_{d,k})\pi_{d,k'}\rangle_{q_{\pi_{d,k}}} \qquad (8.28)
\end{aligned}$$

where the $\langle\rangle$ is the mean operator and the factorized auxiliary distribution $q_{\pi_{d,k}}(\pi_{d,k})$ is chosen to be a normalized version of the Laplace distribution. The normalized Laplace distribution defined in the interval from 0 to 1.

$$q_{\pi_{d,k}}(\pi_{d,k}) = \frac{e^{\frac{-|\pi_{d,k}-\mu_{d,k}|}{b_{d,k}}}}{2b_{d,k}-b_{d,k}\left(e^{\frac{-\mu_{d,k}}{b_{d,k}}}+e^{\frac{\mu_{d,k}-1}{b_{d,k}}}\right)} \qquad (8.29)$$

The parameters $b_{d,k}$ and $\mu_{d,k}$ are hyper-parameters for the factorized auxiliary distribution.

Using the normalized Laplace distribution, the gradient of equation (8.28) becomes

$$\frac{d\mathcal{F}}{d\alpha_{k',w'}} \approx \sum_{d=1}^{D}\left(\frac{I(x_{d,w'}>0)}{\sum_{k=1}^{K}\pi_{d,k}\alpha_{k,w'}}-\psi_{sns}\right)\frac{2b\mu+b^2e^{\frac{-\mu}{b}}-(b^2+b)e^{\frac{\mu-1}{b}}}{2b-b\left(e^{\frac{-\mu}{b}}+e^{\frac{\mu-1}{b}}\right)} \qquad (8.30)$$

where the indexes on μ and b have been removed for simplification, while they are all indexed $_{d,k}$.

The hyper-parameters are estimated starting out with equation (8.22), where we first consider the derivative w.r.t. $\mu_{d,k}$.

$$\begin{aligned}
\frac{d\mathcal{F}}{d\mu_{d,k}} &= \sum_{d=1}^{D} \int \frac{dq_{\pi_d}(\pi_d)}{d\mu_{d,k}} \ln \frac{p(x_d, \pi_d|\alpha)}{q_{\pi_d}(\pi_d|x_d, \alpha)} - q_{\pi_d}(\pi_d) \frac{d \ln q_{\pi_d}(\pi_d)}{d\mu_{d,k}} d\pi_d = 0 \\
&= \sum_{d=1}^{D} \int f_1(\mu_{d,k}, b_{d,k}) e^{\frac{-|\pi_{d,k}-\mu_{d,k}|}{b_{d,k}}} \\
&\quad \left(f_2(\mu_{d,k}, b_{d,k}) e^{\frac{-|\pi_{d,k}-\mu_{d,k}|}{b_{d,k}}} + \ln \frac{n!\Gamma(s_d)}{\Gamma(s_d+n_d)} \right) d\pi_d \qquad (8.31) \\
&+ \sum_{d=1}^{D} \int f_1(\mu_{d,k}, b_{d,k}) e^{\frac{-|\pi_{d,k}-\mu_{d,k}|}{b_{d,k}}} \\
&\quad \left(\sum_{w=1}^{W} I(x_{d,w} > 0) \left(\ln \frac{\alpha_{k,w}}{x_{d,w}} + \ln(\pi_{d,k} + f_3(w)) \right) \right) d\pi_d
\end{aligned}$$

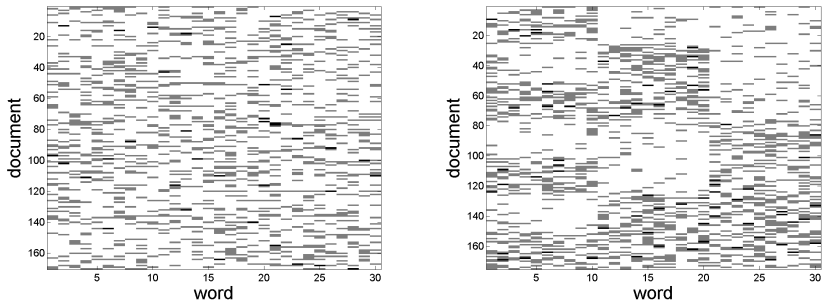
where f_1, f_2 and f_3 are functions that are independent of $\pi_{d,k}$ and $\frac{n!\Gamma(s_d)}{\Gamma(s_d+n_d)}$ is assumed independent of $\pi_{d,k}$ to ease the integral, though still hard. A similar approach can be used to determine the other hyper-parameter b, or it can be fixed for simplification.

8.6 Experiments

In this section we want to verify that the LTEDCM can determine a set of latent topics within a corpora. For these experiments we have generated a set of artificial documents where we know exactly the nature of the latent topics. The latent structure consists of three topics, each containing 10 words that are very likely within the topic. An illustration of the artificial documents is shown in Figure 8.3.

The latent structure in the documents is hard to detect when the documents are unordered, Figure 8.3(a), but can be detected when the documents are ordered, Figure 8.3(b). The question is wether the LTEDCM algorithm can determine the latent structure in the documents.

Because of the rather harsh approximations used in the EM- mean field approximations, we here only consider the simpler maximum likelihood iterative



(a) random order

(b) ordered by class

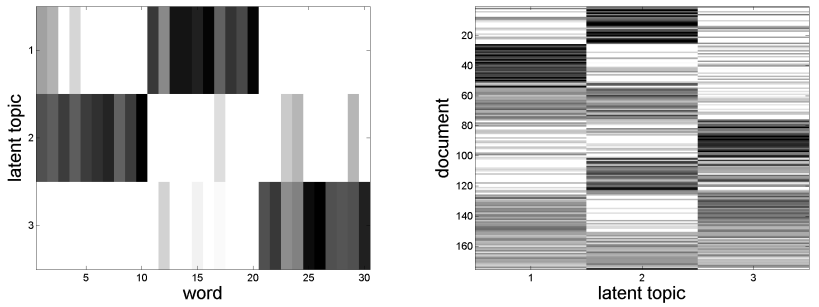
Figure 8.3: Graphical illustration of a the artificial corpora, where the dark points indicate a word being present in a document. The documents are generated using three latent topics combined in seven different ways, each leading to a unique class of documents. There are 10 documents in each class, at first shown in random order (a) followed by an ordered illustration (b) where the latent structure can be spotted. Some noise has also been added to the documents.

gradient approach, when finding the latent structure in a document collection. In Figure 8.4 the LTEDCM algorithm has found the latent structure from the documents in Figure 8.3(a).

The LTEDCM algorithm finds the latent structure of the documents, given by the three topics, each containing 10 words, Figure 8.4(a). Using the latent topic model, documents can be described by the their topic membership, Figure 8.4(b). The latent topic representation clearly reveals the category of the document.

8.7 Discussion

Using the mean field approach, we are forced to make some simplifications that might introduce important errors in the gradient determination process. The impreciseness used in the approximations has left us with an algorithm where convergence is uncertain and slow, while also being much more complex than the simple maximum likelihood gradient approach. Using the standard variational approach (Jordan et al., 1999) (without the mean field approximation)



(a) latent structure

(b) documents represented by latent variables

Figure 8.4: Using latent representation for the documents.

to determine a distribution over the parameters, will also result in hard integrals where approximations must be applied to determine the EM-steps.

Another way to approximate the integrals is by use of samplings based methods, like e.g. Markov chain monte carlo (MCMC) sampling (Andrieu et al., 2003). The samplings based approaches are slower than the variational approaches (Jordan et al., 1999; Blei et al., 2003) but no approximations are necessary that can prevent convergence.

The simple maximum likelihood method that was used here has shown that the latent topic concept works well with the EDCM model. The LTEDCM is unsupervised as is, but can easily be converted to a supervised like in (Blei et al., 2003) or a semi-supervised model as in (Nigam et al., 1998) so that it can be used for text categorization.

8.8 Summary

In this chapter, the EDCM model was integrated in a latent topic scheme, where a document is generated by the EDCM model using a parameterization that originates from multiple latent topics. A maximum likelihood approach and a variational mean field approach have been determined for parameter estimation. The mean field estimate could only be determined using coarse approximations

and is at the same time complex to calculate. In contrast, the maximum likelihood approach consists of a simple iterative procedure which has shown to find latent topics in an artificially generated data-set.

Chapter 9

Conclusion

The focus of this Ph.D. thesis has been aimed at finding new and better approaches to context modeling and categorization. The biggest problem that occur when having machines taking the task of analyzing text is that the machines don't have a huge information bank of knowledge and common sense, like human brains have. When humans are analyzing text content, previous knowledge and common sense is used in the transformation from text into some sort of meaning. Some companies are making progress in integration of knowledge and common sense to text mining systems but that is a very time demanding task and probably won't be available for some more years. The focus has here been on development of new statistical approaches to context modeling and categorization, where everything is learned from the data itself, and no outside knowledge is used.

Previous approaches to context modeling and categorization have made simplifying assumptions about how text is generated. Some of these simplifications have been necessary to reduce the dimensionality of text to an amount that computers can work with. The text simplifications have made modeling and classification feasible and have resulted in relatively accurate models. The simplifications have however also brought along loss of information and noisy data. In this thesis some of these simplifying assumptions have been looked at isolated, and approaches to work around them has been suggested. These approaches has lead to better modeling of text data and better results when

categorizing documents. The approaches presented in this paper have not all been compared mutually, while classification results have been determined on different data-sets. It has not been possible to compare all the methods on the same data-sets.

In previous context categorization approaches, all the words in the vocabulary, with except of a few hundred stopwords, have been used in the categorization process. It is obvious that not all words posses an equally distributed amount of discriminative ability. The issue of determining the best words for discrimination has therefore been pursued. By use of cross validated neural network sensitivities, the parts of the vocabulary that did not posses a consistently high amount of discriminative ability have been determined. Experimental results have shown that removal of a major fraction of the vocabulary, based on the sensitivity score, has resulted in better overall ability of the LSI classifier to categorize documents correctly. This conclusion is also consistent with the idea that the human brain only uses a few important words from a context, when it is to be classified. Though the suggested method is time consuming, the results suggest that other effective ways of determining the discriminative words, can reduce the dimensionality of text pattern recognition systems considerably.

Another issue with previous approaches to context modeling and categorization is the bag-of-words assumption, which assume that the order in which words appear in a text can be disregarded. The assumption has been necessary since the dimensionality of text otherwise would be infinitely large, making it hard for machine learning approaches to generalize in this space. The bag-of-words model however discards some valuable information about the authors style of writing or a fingerprint that tells how the author constructs his sentences. In this thesis, two different approaches has been suggested, for capturing this otherwise lost information. The first approach uses natural language features, characterized by part-of-speech tags, to capture parts of the order information. The part-of-speech tags have been fused with normal word tokens in an early fusion design, resulting in more accurate categorization of documents. The approach is especially valuable when only few training examples are available and when discriminating spam emails. Another approach, using state space models to capture the information from the word order, has been suggested. This approach involved using a hidden Markov model. At first the hidden Markov model was used with a transformed low dimensional vocabulary and secondly with a standard sized vocabulary where the hidden Markov model emission probabilities were approximated. The hidden Markov model approach to text modeling and categorization has shown capability of capturing word sequence information that was different for different categories. The methods were however not more successful than the generic LSI approach, when used for classification. The development of a HMM with shared latent emission parameters for more classes, is however likely to have more success at classifying documents. The natural

language and the hidden Markov model approach, have shown that there is valuable information contained in the order in which words appear in a document, and that this information can be harvested using machine learning methods.

When researchers have created probabilistic generative context models, they have previously made a non-burstiness assumption, hereby also assuming that documents are generated from a static process. The underlying probabilistic process that generate documents is however dynamic because most words in documents are dependent on the words that occurred previously in the document. Researchers have previously used the multinomial distribution to model text documents, which we here have shown fails to model the burstiness phenomenon correctly. We have addressed the burstiness issue and suggested to use the Dirichlet distribution which can be thought of as a bag-of-documents model. The Dirichlet distribution is conceptually modeling the burstiness phenomenon correctly, but this modeling approach is not compatible with the sparseness of text data. It has been suggested instead to use the Dirichlet compound multinomial (DCM) distribution, that can function with the data sparseness while still effectively modeling the burstiness phenomenon correctly. The DCM model has shown to model text data better than the multinomial and at the same time also being better at the categorization task. The DCM does not belong in the exponential family of distributions and does therefore not inherit all the qualities that exponential family distributions possesses. An exponential family approximation of the DCM (EDCM) was introduced, which yields similar categorization performance as the DCM. The EDCM model parameters can further be estimated in a fraction of the time it takes to estimate the DCM parameters. The EDCM model was extended to a latent topic model, where different categories can share latent sub-topics. A maximum likelihood approach and a variational mean field approach has been determined for model estimation. The mean field estimate could only be determined using coarse approximations and is at the same time complex to calculate. In contrast, the maximum likelihood approach consists of a simple iterative procedure. The Latent Topic EDCM model has been shown conceptually effective at finding latent topics in text.

Appendix A

Approximating the Dirichlet Compound Multinomial Distribution

Approximating the Dirichlet Compound Multinomial Distribution

David Kauchak
Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92121
dkauchak@cs.ucsd.edu

Rasmus E. Madsen
Informatics and Mathematical Modelling
Technical University of Denmark
rem@imm.dtu.dk

Charles Elkan
Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92121
elkan@cs.ucsd.edu

Abstract

We investigate the Dirichlet compound multinomial (DCM), which has recently been shown to be a good model for word burstiness in documents. We provide a number of conceptual explanations that account for these recent results. We then derive an exponential family approximation of the DCM that is substantially faster to train, while still producing similar probabilities and classification performance. We also investigate Fisher kernels using the DCM model for generating distributionally based similarity scores. Initial experiments show promise for this type of similarity method.

1 Introduction

In a text document, if a word occurs once, it is likely that the same word will occur again. The multinomial distribution and other related methods do not model burstiness well. We have recently proposed a new distribution for modeling documents, the Dirichlet compound multinomial (DCM [9]).

Both qualitatively and quantitatively, the DCM performs better than the multinomial model on standard document collections ([9]). However, the DCM model is not without problems. First, although the Dirichlet distribution is a member of the exponential family, the DCM is not. Exponential families have many desirable properties ([3]). Second, because of the relative complexity of the DCM expression, understanding its behavior qualitatively is difficult. Third, DCM parameters cannot be estimated quickly; gradient descent methods are necessary [11]. Fast training is important not only for modeling large document collections, but also for using DCM distributions in more complex mixtures or hierarchical models, such as LDA ([4]).

In this paper, we present an approximation of the DCM that is in the exponential family.

We derive an exact solution of the maximum likelihood parameters for the approximate distribution that can be computed efficiently (19 seconds instead of 47 minutes for example).

We also derive the Fisher kernel for the DCM model. Fisher kernels calculate the document similarity in the context of a distribution. The Fisher DCM kernel takes on a similar form to the common document representation TF-IDF ([1]) and therefore provides some explanation to the success of TF-IDF. We examine TF-IDF, the DCM Fisher kernel as well as five standard transforms using a k nearest neighbor classifier for document classification.

2 Multiple Perspectives on the DCM

Given a document x, let x_w be the number of appearances of word w, where w ranges from 1 to the vocabulary size W. The DCM distribution is

$$p(x) = \frac{n!}{\prod_{w=1}^{W} x_w!} \frac{\Gamma(s)}{\Gamma(s+n)} \prod_{w=1}^{W} \frac{\Gamma(x_w + \alpha_w)}{\Gamma(\alpha_w)} \tag{1}$$

where the length of the document is $n = \sum_w x_w$ and $s = \sum_w \alpha_w$ is the sum of the parameters. A DCM is a distribution over alternative count vectors of the same fixed length, n. Although our focus is on word counts in documents, the DCM is applicable in many domains where burstiness (also called contagion) is important, for example [8]. The results of this paper will be useful in these other domains as well.

The DCM, which is also called the multivariate Polya distribution, arises naturally from at least two different generative perspectives. The first perspective is that a document is generated by drawing a multinomial from a Dirichlet distribution, then drawing words from that multinomial. Equation 1 is obtained by integrating over all possible multinomials:

$$p(x) = \int_{\theta} p(x|\theta) p(\theta|\alpha)$$

where $p(\theta|\alpha)$ is a Dirichlet distribution and $p(x|\theta)$ is a multinomial distribution. The intuition behind this perspective is that the multinomial is a document-specific subtopic. For example, articles about cars may use either the word "hood" or the word "bonnet"; whichever word is used is likely to be used repeatedly. (This notion of subtopic is different from the notion of topic used in Hofmann's PLSI ([5]) or Blei's LDA ([4]).)

The second perspective is that a document is generated by an urn scheme ([2] Ch. 35, Section 13.1). Given an urn filled with colored balls, one color for each word in the vocabulary, the simplest scheme is to draw balls with replacement. This yields a multinomial distribution, where the initial proportions of ball colors are the multinomial parameters.

The DCM arises from the Polya-Eggenberger urn scheme ([2] Ch. 40, Section 2). In this scheme, each time one ball is drawn from the urn, two balls of the same color are placed back in the urn. Words that have already been drawn are therefore more likely to be drawn again.

Each DCM parameter α_w is the number of balls of color w in the urn initially. The sum $s = \sum_w \alpha_w$ measures the burstiness of the distribution. Increasing s decreases the burstiness of the distribution, and vice versa. As $s \to \infty$ the DCM tends towards the multinomial.

3 Approximating the DCM

In this section we derive an exponential family approximation of the DCM distribution that we call the EDCM and investigate the qualitative behavior of the EDCM. Empirically, given

a DCM modeling a class of documents, $\alpha_w \ll 1$ for almost all words w. For example, for a DCM trained on one class of newsgroup articles, the average α_w is 0.004. Of the 59,826 parameters, 99% are below 0.1, only 17 are above 0.5, and only 5 are above 1.0.

When α_w is small, a useful approximation is $\Gamma(x+\alpha)/\Gamma(\alpha) \approx \Gamma(x)\alpha$. Using also the fact that for integers, $\Gamma(z) = (z-1)!$, we obtain the EDCM distribution

$$q(x) = \frac{n!}{\prod_{w:x_w \geq 1} x_w} \frac{\Gamma(s)}{\Gamma(s+n)} \prod_{w:x_w \geq 1} \beta_w. \tag{2}$$

For clarity, we denote the EDCM parameters β_w. Because of the approximation used, this distribution is not proper, that is it does not sum to 1.0. In principle, we can sum $q(x)$ over all values of x to get a normalizing constant $Z(\beta)$.

In practice the values given by Equation 2 are very close to those given by Equation 1. On a sample set of 4000 documents from 20 different classes $q_\alpha(x)$ is highly correlated with $p_\alpha(x)$ (0.999995 using Pearson's correlation). On average, $q_\alpha(x)$ only deviates by 2.2% from $p_\alpha(x)$.

This high correlation is because the $\Gamma(x+\alpha)/\Gamma(\alpha)$ approximation is highly accurate for small α_w and small integers x. For a typical α vector trained from 800 documents, of the 69,536 non-zero word counts, the approximation is on average 3.9% off.

Because the EDCM approximates the DCM, insights for one distribution carry over to the other. We can rewrite the EDCM equation as

$$q(x) = n! \frac{\Gamma(s)}{\Gamma(s+n)} \prod_{w:x_w \geq 1} \frac{\beta_w}{x_w}. \tag{3}$$

For fixed s and n, the probability of a document is proportional to $\prod_{w:x_w \geq 1} \beta_w / x_w$. This means that the first appearance of a word w reduces the probability of a document by β_w, a word-specific factor that is almost always much less than 1.0, while the mth appearance of any word reduces the probability by $(m-1)/m$, which tends to 1 as m increases. This behavior reveals how the EDCM, and hence the DCM, allow multiple appearances of the same word to have high probability. In contrast, with a multinomial each appearance of a word reduces the probability by the same factor.

An exponential family distribution has the form $f(x)g(\beta)\exp[t(x) \cdot h(\beta)]$ where $t(x)$ is a vector of sufficient statistics and $\theta = h(\beta)$ is the vector of so-called "natural" parameters. We can write $q(x)$ in this form as

$$q(x) = \left(\prod_{w:x_w \geq 1} x_w^{-1} \right) n! \frac{\Gamma(s)}{\Gamma(s+n)} \exp[\sum_w I(x_w \geq 1) \ln \alpha_w].$$

For the EDCM distribution, the sufficient statistics for a document x are $\langle t_1(x), \cdots, t_W(x) \rangle$ where $t_w(x) = I(x_w \geq 1)$ and W is the number of words in the vocabulary. The expression also shows that the natural parameters for the EDCM distribution are $\theta_w = \ln \beta_w$.

4 MLE of EDCM parameters

We can calculate maximum likelihood parameter estimates for the EDCM simply by taking the derivative of the log-likelihood function, unlike for the DCM. From (3) the log-likelihood function is

$$l(x) = \log n + \log \Gamma(s) - \log \Gamma(s+n) + \sum_{w:x_w \geq 1} \log \beta_w - \log x_w.$$

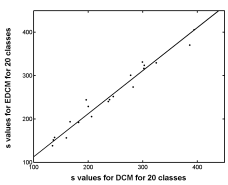


Figure 1: Sum of DCM α_w versus sum of EDCM β_w for 20 classes.

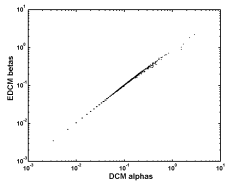


Figure 2: DCM α_w versus EDCM β_w for a single class.

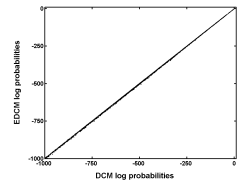


Figure 3: $\ln p_\alpha(x)$ (DCM) versus $\ln q_\beta(x)$ (EDCM) for 4000 test documents over 20 classes.

Given a training set D of documents, where document number d has length n_d and word counts x_{dw}, the partial derivative of the log-likelihood of the data is

$$\frac{\partial l(D)}{\partial \beta_w} = |D|\Psi(s) - \sum_d \Psi(s + n_d) + \sum_d I(x_{dw} \geq 1)\frac{1}{\beta_w}$$

Setting this derivative to zero and solving for β_w we get

$$\beta_w = \frac{\sum_d I(x_{dw} \geq 1)}{|D|\Psi(s) - \sum_d \Psi(s + n_d)}. \tag{4}$$

We can compute $s = \sum_w \beta_w$ by summing Equation (4) over all words, giving

$$s = \frac{\sum_w \sum_d I(x_{dw} \geq 1)}{|D|\Psi(s) - \sum_d \Psi(s + n_d)}$$

where the numerator is the number of times a word appears at least once in a document. This equation can be solved numerically for s efficiently, since it involves only a single unknown.

Figure 4 shows the s values learned using the above method for the EDCM compared to the corresponding DCM s values. Although not exactly the same, they are very similar and highly correlated. Figure 4 shows the learned EDCM β_w compared to DCM α_w. Again, we see that there is a strong correlation between the DCM and EDCM (0.9868).

Figure 4 shows the log probabilities for the DCM model ($p_\alpha(x)$) versus the EDCM model ($q_\beta(x)$) for 4000 documents. Given that $q(x)$ is a very good approximation of $p(x)$ and that α and β are very similar, it is not surprising that the DCM and EDCM probabilities are extremely similar. The probabilities are highly correlated (0.999992) and on average $q_\beta(x)$ only deviates 0.25% from $p_{\alpha(x)}$.

5 Fisher scores for the DCM

The standard approach to measure similarity between document count vectors is cosine similarity:

$$\text{sim}(x, y) = \frac{x \cdot y}{||x||_2 ||y||_2}$$

where $||z||_2 = \sqrt{\sum_i z_i^2}$. This approach is somehow unsatisfying in that it does not take into account document characteristics, for example some words are more informative than others.

The TF-IDF (term frequency-inverse document frequency) representation was proposed to address such concerns ([1]). The term frequencies, x_w (i.e. word counts) are multiplied by the log of the inverse of the number of documents x_w appears in

$$\text{IDF}(x_w) = \log \frac{|D|}{\sum_{d \in D} I(x_{dw} > 0)}$$

where $|D|$ is the number of documents. The TF-IDF representation is then used with the standard similarity measures. In practice, TF-IDF works well, however, there is no strong theoretical justification for using this representation versus other heuristics.

One motivation for TF-IDF is to incorporate knowledge about the document collection into the similarity measure. Given a distribution $p(x)$, the Fisher kernel measures the similarity of x and y in the context of the distribution p:

$$k(x, y) = s(x)^T H s(y)$$

where $s(x) = \nabla x$ is the Fisher score vector for x, i.e. the partial derivatives of the log-likelihood $l(x)$ with respect to the parameters α_w, and H is the Hessian of second partial derivatives of $l(x)$ with respect to the parameters ([7], [6]). With this definition, $k(x, y)$ is invariant to a change in parameterization of p. However, H is usually approximated by the identity matrix, and in this case the Fisher kernel is different for different parameterizations.

For the DCM, the partial derivative of the log-likelihood is

$$\frac{\partial l(x)}{\partial \alpha_w} = \Psi(s) - \Psi(s + n) + \Psi(x_w + \alpha_w) - \Psi(\alpha_w). \tag{5}$$

The Fisher kernel can then be calculated as the dot-product of these partial derivatives for two documents x and y

We can use an approximation for $\Psi(\cdot)$ from [11],

$$\Psi(z) \approx \begin{cases} \log(z - 0.5) & \text{if } z \geq 0.6 \\ -1/z - \Psi(1) & \text{if } z < 0.6 \end{cases}$$

to get insight into this representation. If α_w is small, Equation 5 can then be approximated as

$$\frac{\partial l(x)}{\partial \alpha_w} \approx \Psi(s) - \Psi(s + n) + I(x_w \geq 1)(\log x_w + 1/\alpha_w + \gamma + \log x_w).$$

In this form the Fisher score is analogous to TF-IDF. The first term is the same for all words and documents. The second term takes into account length variation between documents. Finally, $I(x_w \geq 1)$ is the boolean indicator of the term frequency and from Equation (4) we see that $1/\alpha_w$ is approximately equal to

$$\frac{|D|\Psi(s) - \sum_d \Psi(s + n_d)}{\sum_d I(x_{dw} \geq 1)}.$$

Given that $|D|\Psi(s) - \sum_d \Psi(s + n_d) = O(|D|)$, $1/\alpha_w$ is similar to inverse document frequency as used in TF-IDF.

6 Experiments

In this Section, we examine the modeling and classification performance of the DCM and EDCM models in comparison to a number of standard methods. We first examine the problem of modeling, which has recently received a lot of attention ([4], [12]). We then compare three naive Bayes classifiers (multinomial, DCM and EDCM, see [9] for details on naive

Data set	M	DCM	EDCM	L0	$\sqrt{\text{L1}}$	L1	L2	TF-IDF	Fisher-DCM	Fisher-M
20 news	0.843	0.845	0.851	0.606	0.744	0.675	0.778	0.828	0.677	0.665
industry	0.791	0.795	0.781	0.624	0.254	0.017	0.567	0.881	0.761	0.735

Table 1: Accuracy scores for averages of 10 random splits of the 20 newsgroups and industry sector data sets. Scores right of EDCM are using k-NN with $k = 3$.

Bayes classification) and seven k nearest neighbor (k-NN) classification methods. Three k-NN methods represent different length normalization of the documents $x_{norm} = x/||x||_p$, for $p = 0, 1, 2$ known as the L0, L1 and L2 norms. We use the TF-IDF representation with L2 length normalization. The three other methods are kernel methods: Bhattacharyya kernel, which is $\sqrt{x}/||x||_1$ (denoted $\sqrt{\text{L1}}$), the Fisher-DCM kernel as described above[1] and the Fisher multinomial kernel (Fisher-M).

6.1 Data

We use two standard corpora, the industry sector and 20 newsgroups document collections. Documents are tokenized, stop words removed and count vectors extracted using the Rainbow toolbox [10]. The industry sector[2] data set contains 9555 documents distributed in 104 classes. The data set has a vocabulary of 55,055 words, and each document contains on average 606 words. The data are split into halves for training and testing. The 20 newsgroups[3] data set contains 18,828 documents belonging to 20 classes. This collection has a vocabulary of 61,298 words with an average document length of 116 words. The data are split into $80/20$ fractions for training and testing.

Unfortunately, most of the commonly used and available data sets contain relatively short documents. These do provide for interesting initial experiments, but are not representative of the variety of documents encountered in real problems and, because of the shortness, do not contain the strong word burstiness of longer documents that the DCM and EDCM are particularly well suited for. In the final version of this paper, we plan to include experiments on longer documents.

6.2 Modeling Performance

We can measure the modeling performance of a model by the perplexity on a set of unseen documents, D:

$$\exp\left(\frac{-\sum_{d=1}^{D}\sum_{w=1}^{W}\log p(x_{dw})}{\sum_{d=1}^{D} n_d}\right)$$

Perplexity measures how well a model predicts unseen data. A lower value indicates better prediction.

We calculated the perplexity for the 20 newsgroups data, with one model trained for each of the 20 classes over 10 random splits. The perplexity for multinomial models is 5311 ± 755 versus 2609 ± 382 for DCM models, where both results are means $\pm$ one standard deviation. The DCM model predicts unseen data significantly better than the multinomial model. Because the EDCM is not a normalized probability distribution, we cannot calculate perplexity results, however, given that the EDCM probabilities are extremely correlated with the DCM probabilities, we would expect similar modeling performance from the normalized EDCM distribution.

[1] We use β trained using EDCM instead of α for computation reasons

[2] `http://www.cs.umass.edu/~mccallum/code-data.html`

[3] `http://people.csail.mit.edu/people/jrennie/20Newsgroups`

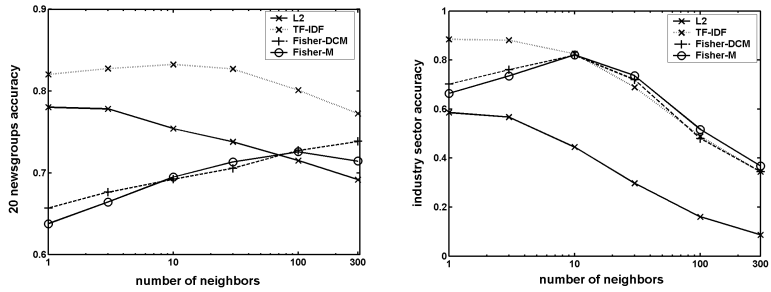


Figure 4: k-NN average accuracy scores for increasing number of nearest neighbors for (left) 20 newsgroups and (right) industry sector.

To get some intuition about the difference in modeling performance between the DCM and the multinomial, we can examine how many multinomials it takes to obtain a similar level of modeling performance to the DCM. Based on the results given in [12], the perplexity results for the DCM are similar to the LDA and DELSA when using 10 latent topics. We plan to investigate this further for the final version of the paper.

6.3 Classification

Table 1 shows accuracy averages for 10 splits of each data set for the ten different classification methods. The first three methods are naive Bayes classifiers and the last seven k-NN classifiers where $k = 3$. As we would hope, the DCM and EDCM model have very similar classification performances. In fact, for 20 newsgroups, the EDCM model actually performs better than the DCM model. This is particularly encouraging considering the EDCM model was almost 150 times faster to train than the DCM model (19 seconds vs. 2788 seconds for a 20 newsgroups split using the fastest DCM training method). Both the DCM and EDCM use a naive smoothing method and still perform similarly to the multinomial, which uses better smoothing (0.01 has been found to work well).

In general, the naive Bayes classifiers perform significantly better than the k-NN classifiers (with the surprising exception of TF-IDF on the industry sector data set). Of the k-NN classifiers, TF-IDF performs the best. Both the Fisher kernel methods perform well, particularly compared to the other NN methods on the industry sector data, which is a sparser data set, i.e. a smaller number of documents per class. The Fisher-DCM method performs better than the Fisher multinomial method (significantly better using a t-test on industry sector).

Figure 4 shows four of the k-NN methods for the two data sets for increasing k. Both Fisher kernel methods continued to utilize more training data on the 20 newsgroups data set, while the performance of other approaches tended to decrease as more data is added. All methods tended to perform poorly on the industry sector data set as k increased. All of the norm based methods did poorly on the industry sector data, particularly as k increased.

7 Discussion

In this paper we have examined the DCM distribution as a model for text documents. The DCM models burstiness well experimentally ([9]) and we presented a number of additional

intuitive explanations to confirm this. However, the DCM distribution is not in the exponential family of distributions and is slow to train. We proposed an approximation to the DCM distribution called the EDCM distribution. The EDCM is in the exponential family of distributions and is orders of magnitude faster to train.

The parameters learned by the EDCM model and the associated probabilities are extremely similar to those learned by the DCM. On two classification tasks, the EDCM model performs similarly to the DCM model. By overcoming the computational complexity problems of the DCM, the EDCM distribution can be used within larger models such as LDA ([4]) or PLSA ([5]). We stress that the DCM and EDCM models are complementary to these recent topic-based mixtures models, which represent a class of documents as a mixture of K fixed topics, each topic being represented by a multinomial. Because of this, these models will also suffer from poor modeling of word burstiness. Future research will investigate the use of the DCM or EDCM to represent a topic, instead.

The only remaining hurdle for this type of integration using the EDCM model is to work out the normalizing constant. Even without the normalizing constant, the EDCM parameters can be used as a starting point for current gradient descent DCM learning methods, which should decrease the training time for the DCM.

We also investigated Fisher kernels based on the DCM model, which try and generate a document similarity score in the context of a specified distribution. Initial experiments show that k-NN classifiers based on the Fisher kernels are robust and perform generally better than standard normalization techniques. Using the Fisher kernel for the DCM, we also provided some insight into why the commonly used TF-IDF transformation performs well. In the future, we plan to investigate these kernels for more complicated classifiers, such as support vector machines.

References

[1] Akiko Aizawa. An information-theoretic perspective of tf-idf measures. *Information Processing and Management*, 39(1):45–65, 2003.

[2] N. Balakrishnan, Norman L. Johnson, and Samuel Kotz. *Discrete Multivariate Distributions*. New York: John Wiley and Sons Inc., 1997.

[3] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. To appear in *Journal of Machine Learning Research*, 2005.

[4] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[5] T. Hofmann. Probabilistic latent semantic indexing (PLSI). In *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval*, pages 50–57, Berkeley, California, 1999. ACM.

[6] Tomas Hofmann. Learning the similarity of documents. In *Proceedings of NIPS*, 2000.

[7] Tommi S. Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *Proceedings of NIPS*, 1999.

[8] Paul Kvam and Dennis Day. The multivariate Polya distribution in combat modeling. *Naval Research Logistics*, 28:1–17, 2001.

[9] Rasmus E. Madsen, David Kauchak, and Charles Elkan. Modeling word burstiness using the Dirichlet distribution. To appear in *Proceedings of ICML*, 2005.

[10] Andrew Kachites McCallum. *Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering*. www.cs.cmu.edu/˜mccallum/bow, 1996.

[11] T.P. Minka. *Estimating a Dirichlet distribution*. www.stat.cmu.edu/˜minka/papers/dirichlet, 2003.

[12] K. Yu, S. Yu, and V. Tresp. Dirichlet enhanced latent semantic analysis. In *AISTAT Workshop Proceedings*, 2005.

Appendix B

Modeling Word Burstiness Using the Dirichlet Distribution

Modeling Word Burstiness Using the Dirichlet Distribution

Rasmus E. Madsen REM@IMM.DTU.DK
Department of Informatics and Mathematical Modelling, Technical University of Denmark

David Kauchak DKAUCHAK@CS.UCSD.EDU
Charles Elkan ELKAN@CS.UCSD.EDU
Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA 92092-0144

Abstract

Multinomial distributions are often used to model text documents. However, they do not capture well the phenomenon that words in a document tend to appear in bursts: if a word appears once, it is more likely to appear again. In this paper, we propose the Dirichlet compound multinomial model (DCM) as an alternative to the multinomial. The DCM model has one additional degree of freedom, which allows it to capture burstiness. We show experimentally that the DCM is substantially better than the multinomial at modeling text data, measured by perplexity. We also show using three standard document collections that the DCM leads to better classification than the multinomial model. DCM performance is comparable to that obtained with multiple heuristic changes to the multinomial model.

1. Introduction

Document classification is the task of identifying what topic(s) a document concerns. Generative approaches to classification are popular since they are relatively easy to interpret and can be trained quickly. With these approaches, the key problem is to develop a probabilistic model that represents the data well. Unfortunately, for text classification too little attention has been devoted to this task. Instead, a generic multinomial model is typically used.

Recent work (Rennie et al., 2003) has pointed out a number of deficiencies of the multinomial model, and suggested heuristics to improve its performance. In this paper, we follow an alternative path. We present a different probabilistic model that, without any heuristic changes, is far better suited for representing a class of text documents.

As most researchers do, we represent an individual document as a vector of word counts (Salton et al., 1975). This bag-of-words representation loses semantic information, but it simplifies further processing. The usual next simplification is the assumption that documents are generated by repeatedly drawing words from a fixed distribution. Under this assumption, word emissions are independent given the class, i.e. the naive Bayes property holds. This property is not valid (Lewis, 1998), but naive Bayes models remain popular (McCallum & Nigam, 1998; Sebastiani, 2002) because they are fast and easy to implement, they can be fit even with limited training data, and they do yield accurate classification when heuristics are applied (Jones, 1972; Rennie et al., 2003).

The central problem with the naive Bayes assumption is that words tend to appear in bursts, as opposed to being emitted independently (Church & Gale, 1995; Katz, 1996). Rennie et al. (2003) address this issue by log-normalizing counts, reducing the impact of burstiness on the likelihood of a document. Teevan and Karger (2003) empirically search for a model that fits documents well within an exponential family of models, while Jansche (2003) proposes a zero-inflated mixture model.

In this paper we go further. We show that the multinomial model is appropriate for common words, but not for other words. The distributions of counts produced by multinomials are fundamentally different from the count distributions of natural text. Zipf's law (Zipf, 1949) states that the probability p_i of occurrence of an event follows a power law $p_i \approx i^{-a}$, where i is the rank of the event and a is a parameter. The most famous

Appearing in *Proceedings of the 22st International Conference on Machine Learning*, Bonn, Germany, 2005.

example of Zipf's law is that the frequency of an English word, as a function of the word's rank, follows a power law with exponent close to minus one.

We propose to model a collection of text documents with a Dirichlet distribution (Minka, 2003). The Dirichlet distribution can be interpreted in two ways for this purpose, either as a bag-of-scaled-documents or as a bag-of-bags-of-words. We show below that the latter approach works well.

Dirichlet distributions have been used previously to model text, but our approach is fundamentally different. In the LDA approach (Blei et al., 2003) the Dirichlet is a distribution over topics, while each topic is modeled in the usual way as a multinomial distribution over words. In our approach, each topic, i.e. each class of documents, is modeled in a novel way by a Dirichlet distribution instead of by a multinomial. Our approach is therefore complementary to the LDA and related approaches.

2. Multinomial modeling of text

When using a multinomial distribution for text modeling, the multinomial specifies the probability of observing a given vector of word counts, where the probability θ_w for the emission of word w is subject to the constraints $\sum_w \theta_w = 1$ and $\theta_w > 0$. The probability of a document x represented as a vector of word counts x_w is

$$p(x|\theta) = \frac{n!}{\prod_{w=1}^{W} x_w!} \prod_{w=1}^{W} \theta_w^{x_w}$$

where x_w is the number of times word w appears in the document, θ_w is the probability of emitting word w, W is the size of the vocabulary, and $n = \sum x_w$.

The multinomial distribution is different for each different document length n. This is not a problem when learning the parameters; it is possible to generalize over documents with different lengths. The maximum likelihood parameter estimates $\hat{\theta}$ are

$$\hat{\theta}_w = \frac{\sum_{d=1}^{D} x_{dw}}{\sum_{w'=1}^{W} \sum_{d=1}^{D} x_{dw'}}$$

where d is the document number and D is the number of documents. These estimates depend only on the fraction of times a given word appears in the entire corpus.

When the multinomial model is used to generate a document, the distribution of the number of emissions (i.e. count) of an individual word is a binomial:

$$p(x_w|\theta) = \binom{n}{x_w} \theta_w^{x_w} (1-\theta_w)^{n-x_w}. \qquad (1)$$

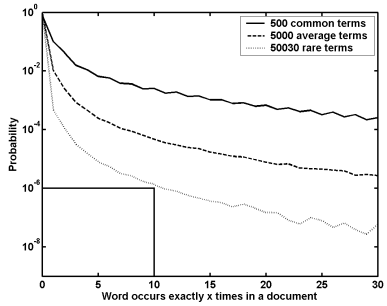


Figure 1. Count probabilities of common, average and rare words in the industry sector corpus. The figure shows, for example, that the probability a given rare word occurs exactly 10 times in a document is 10^{-6}. The ripple effect seen in common words occurs because no vocabulary pruning is done, so certain HTML keywords such as "font" or "table" occur an even number of times in beginning and ending tags.

3. The burstiness phenomenon

The term "burstiness" (Church & Gale, 1995; Katz, 1996) describes the behavior of a rare word appearing many times in a single document. Because of the large number of possible words, most words do not appear in a given document. However, if a word does appear once, it is much more likely to appear again, i.e. words appear in bursts. To illustrate this behavior, the probability that a given word occurs in a document exactly x times is shown in Figure 1 for the industry sector corpus. Words have been split into three categories based on how often they appear in the corpus. The categories are "common," "average," and "rare." The common words are the 500 most frequent words; they represent 1% of the words in the vocabulary and 71% of the emissions. The average words are the next 5000 most common words; they represent 10% of the vocabulary and 21% of the emissions. The rare words are the rest of the vocabulary (50,030 words) and account for 8% of the emissions.

A few things should be noted about Figure 1. Not surprisingly, common words are more probable than average words which are more probable than rare words. Interestingly, though, the curves for the three categories of words are close to parallel and have similar decay rates. Even though average and rare words are less likely to appear, once a word has appeared, the probability that it will occur multiple times is similar across all words.

Equation (1) shows that it is unlikely under the

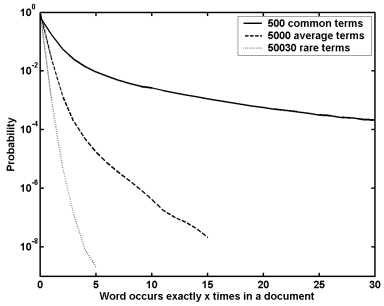


Figure 2. Count probabilities for a maximum likelihood multinomial model, trained with the industry sector corpus.

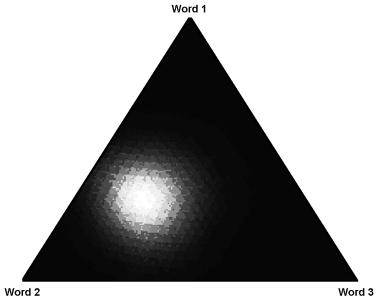


Figure 3. Simplex of possible count vectors using the multinomial bag-of-words model with parameters $\theta = \{0.4375, 0.25, 0.3125\}$ and $n = 50$.

multinomial model for a word to occur many times in a document, because the single word count distribution decays exponentially. Figure 2 shows the average word count probabilities from ten synthetic corpora generated from a multinomial model trained on the industry sector corpus. Each synthetic corpus was generated so its documents have the same length distribution as documents in the industry sector corpus.

The multinomial captures the burstiness of common words, but the burstiness of average and rare words is not modeled correctly. This is a major deficiency in the multinomial model since rare and average words represent 99% of the vocabulary and 29% of emissions and, more importantly, these words are key features for classification. An explanation for this behavior is that the common words are more likely to satisfy the independence assumption, since many of the common words are non-content, function words. The rare and average words are information-carrying words, making them more likely to appear if they have already appeared in a document.

Figure 3 shows the simplex of possible count vectors for three words when a multinomial model is used and the sum of the counts is $n = 50$. All the probability mass is close to the most likely counts, so burstiness is not likely. If bursts were likely, then the probability mass would be on the edges and corners of the simplex.

4. Dirichlet modeling of text

The Dirichlet distribution is a probability density function over distributions. It is defined as

$$p(\theta|\alpha) = \frac{\Gamma\left(\sum_{w=1}^{W} \alpha_w\right)}{\prod_{w=1}^{W} \Gamma(\alpha_w)} \prod_{w=1}^{W} \theta_w^{\alpha_w - 1}$$

where θ is a vector in the W-dimensional probability simplex, i.e. $\sum_w \theta_w = 1$. The α vector entries are the parameters of the Dirichlet.

When modeling text, the θ vector represents a document, making the model have the form $data^{parameter}$. This form makes Dirichlet models qualitatively similar to Zipf distributions, where the parameter is an exponent. In contrast, the multinomial model has the form $parameter^{data}$. By rewriting the Dirichlet distribution in the exponential family form

$$\begin{aligned} \log p(\theta|\alpha) &= \sum_{w=1}^{W} (\alpha_w - 1) \log \theta_w \\ &+ \log \Gamma(\sum_{w=1}^{W} \alpha_w) - \sum_{w=1}^{W} \log \Gamma(\alpha_w) \end{aligned}$$

we see that the log transform of the data is naturally considered. This is again in contrast to the multinomial in exponential form.

In the bag-of-words representation, documents are vectors of word counts. The Dirichlet distribution is a distribution not over count vectors but over probability vectors. There are multiple ways to represent a document as a probability vector. The obvious choice is to let θ be a scaled version of the document vector. We can view this approach as drawing a scaled bag of words (representing one document) from the Dirichlet bag-of-scaled-documents.

The difficulty with this approach is that document vectors are sparse in nature, i.e. each document tends to contain only a small subset of the vocabulary, resulting in many of the entries being zero. Since the Dirichlet likelihood for a probability vector is zero if the vector contains any zeros, smoothing of the training data

is required before a Dirichlet distribution can be estimated. In practice, this results in an over-smoothed distribution, where all the rare words have about the same probability of appearing in all the classes. Since the rare words contain most of the discriminative information, this model is not useful for document classification.

A better approach is hierarchical: let the count vector for each document be generated by a multinomial distribution whose parameters are generated by the Dirichlet distribution. This model is called the Dirichlet compound multinomial (DCM) (Minka, 2003) and can be understood as a bag-of-bags-of-words. To generate a document using the DCM, a sample is first drawn from the Dirichlet to get a multinomial distribution, then words are iteratively drawn for the document based on the multinomial distribution.

Although we talk about the parameters θ, the only genuine parameters for a DCM are the α vector entries. The likelihood of a document of length n is an integral over θ vectors weighted by a Dirichlet distribution:

$$
\begin{aligned}
p(x|\alpha) &= \int_\theta p(x|\theta)p(\theta|\alpha)d\theta \\
&= \int_\theta \frac{n!}{\prod_{w=1}^{W} x_w!}\left(\prod_{w=1}^{W}\theta_w^{x_w}\right)\frac{\Gamma\left(\sum_{w=1}^{W}\alpha_w\right)}{\prod_{w=1}^{W}\Gamma(\alpha_w)}\prod_{w=1}^{W}\theta_w^{\alpha_w-1}d\theta \\
&= \frac{n!}{\prod_{w=1}^{W} x_w!}\frac{\Gamma\left(\sum_{w=1}^{W}\alpha_w\right)}{\prod_{w=1}^{W}\Gamma(\alpha_w)}\int_\theta\prod_{w=1}^{W}\theta_w^{\alpha_w+x_w-1}d\theta \\
&= \frac{n!}{\prod_{w}^{W} x_w!}\frac{\Gamma\left(\sum_{w=1}^{W}\alpha_w\right)}{\Gamma\left(\sum_{w=1}^{W} x_w+\alpha_w\right)}\prod_{w=1}^{W}\frac{\Gamma\left(x_w+\alpha_w\right)}{\Gamma(\alpha_w)}. \qquad (2)
\end{aligned}
$$

The last step of equation (2) is obtained by noticing that $\prod_{w=1}^{W}\theta_w^{x_w}$ combined with $\prod_{w=1}^{W}\theta_w^{\alpha_w-1}$ is the unnormalized version of the Dirichlet distribution $p(\theta|\alpha+x)$, and using the fact that $\int p(\theta|\alpha)d\theta = 1$.

There exists no closed-form solution for the maximum likelihood parameter values for the DCM model. An iterative gradient descent optimization method can be used to estimate the α vector by computing the gradient of the DCM log likelihood. Two bound inequations are used with the gradient, leading to the update

$$
\alpha_w^{new} = \alpha_w \frac{\sum_{d=1}^{D}\Psi(x_{dw}+\alpha_w)-\Psi(\alpha_w)}{\sum_{d=1}^{D}\Psi(x_{dw}+\sum_{w'=1}^{W}\alpha_{w'})-\Psi(\sum_{w'=1}^{W}\alpha_{w'})}
$$

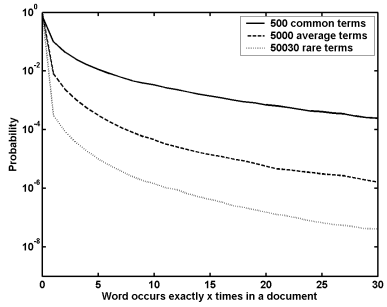


Figure 4. Count probabilities for a maximum likelihood DCM model, trained with the industry sector corpus.

where the digamma function Ψ is defined as $\Psi(\alpha) = \frac{d}{d\alpha}\log\Gamma(\alpha)$. For more information see Minka (2003).

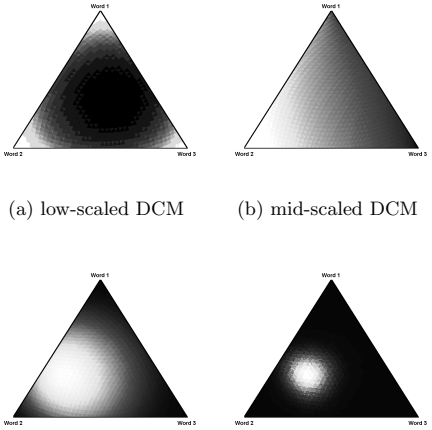
Figure 2 shows that the multinomial is unable to correctly model the burstiness of natural text. Figure 4 shows the probability of a term appearing multiple times in a document under the DCM model. The experimental design is similar to that of Figure 2. The DCM can model burstiness for all word types. The curves for the three categories of words are close to parallel, like in the original data (Figure 1).

A Dirichlet has the same number of parameters as a multinomial, so it is not obvious how the DCM can model burstiness. The reason is that the multinomial parameters are constrained to sum to one, unlike the DCM parameters, so the DCM has one extra degree of freedom. The smaller the α parameters are, the more the emission of words is bursty. Figure 5 shows the simplex of count probabilities given by equation (2) for three words with $n = 50$, for different α vectors. When the parameters are small, most probability mass is located near the corners of the simplex. When the parameters are large, most mass is near the center of the simplex, modeling word counts that are not bursty. As the parameters tend to infinity, the DCM model approaches equivalence with a multinomial model.

Since a DCM has only a single extra degree of freedom compared to a multinomial, it cannot model the individual burstiness of each word. This inflexibility is a trade-off between the expressiveness of the model and its learnability with limited training data.

5. Experiments

We use three standard corpora: the so-called industry sector, 20 newsgroups and Reuters-21578 document



(a) low-scaled DCM (b) mid-scaled DCM

(c) high-scaled DCM (d) multinomial

Figure 5. Three word probability simplices with DCM parameters (a) 0.44, 0.25, 0.31 (b) 1.32, 0.75, 0.93 and (c) 3.94, 2.25, 2.81 and multinomial (d) 0.44, 0.25, 0.31.

collections. We compare the DCM method against the multinomial model as well as against recent heuristically improved versions of the multinomial method, which perform as well as discriminative methods (Rennie et al., 2003). We have made every effort to reproduce previous results in order to ensure that a fair comparison is made.

Documents are preprocessed and count vectors are extracted using the Rainbow toolbox (McCallum, 1996). The 500 most common words are removed from the vocabulary to ensure that our results are comparable with previous results. The Dirichlet toolbox (Minka, 2003) is used to estimate the parameters of the DCM model.

When using DCM or multinomial models for classification, we apply Bayes' rule $p(y|x) = p(x|y)p(y)/p(x)$ with a uniform prior $p(y)$ over the classes, following (Rennie et al., 2003). The class y with the highest probability $p(y|x)$ is the predicted label.

5.1. Heuristics to improve multinomial models

Various heuristics have been applied to the multinomial and related models to enhance classification performance (Rennie et al., 2003). We briefly review them here for completeness. The first heuristic is the log-transformation of the data, which has been shown to mitigate problems caused by burstiness. In the tables of results below, L is shorthand for the transformation

$$x_{dw}^{\log} = \log(1 + x_{dw})$$

where all logarithms are natural, i.e. base e. One traditional information retrieval heuristic is the term-frequency inverse-document-frequency (TFIDF) transformation, which exists in various forms (Aizawa, 2003). The version used here includes the log-transformation:

$$x_{dw}^{\text{tfidf}} = \log(1 + x_{dw}) \log \frac{D}{\sum_{d'=1}^{D} \delta_{d'w}}$$

where δ_{dw} is 1 if word w is present in document d. After the TFIDF transformation, document vectors are L_2-normalized:

$$x_{dw}^{\text{norm}} = \frac{x_{dw}^{\text{tfidf}}}{\sqrt{\sum_{w'=1}^{W} {x_{dw}^{\text{tfidf}}}^2}}.$$

This makes all document vectors have the same length and therefore the same amount of influence on the model parameters. The combination of TFIDF and L_2-normalization is denoted TW-L below.

A couple of additional heuristics are also applied. The most important is complement modeling (Rennie et al., 2003): the model for a class is trained with all the documents that do not belong to that class. In most cases, by using other classes' data, substantially more data is available for parameter estimation resulting in better modeling of each class:

$$\hat{\theta}_{kw}^{\text{comp}} = \frac{\sum_{i:y_{dk}\neq 1} x_{dw}^{\text{norm}} + \varepsilon}{\sum_{w'=1}^{W} \sum_{i:y_{dk}\neq 1} x_{dw'}^{\text{norm}} + \varepsilon}$$

where the class variable y_{dk} equals 1 if document d belongs to class k and 0 otherwise, and ε is a smoothing constant, which is necessary to prevent probabilities for unseen words from becoming zero. Typically, $\varepsilon = 1$, but this value is often much too large, so we also report results below with $\varepsilon = 0.01$. Non-complement models are smoothed in a similar way. DCM models are also smoothed, but differently, by adding a small constant to each parameter α_w. This constant equals 0.01 times the smallest non-zero estimated α_w.

If documents can belong to more than one class, the usual approach is a one-versus-all-but-one classifier. The complement model is the same as the standard model, in these cases. For this reason, the complement model is defined differently for multi-label problems as an all-versus-all-but-one classifier (Rennie et al., 2003, Appendix A).

Finally, the model parameters are log-normalized:

$$\hat{\theta}_{kw}^{\text{norm}} = \frac{\log \hat{\theta}_{kw}^{\text{comp}}}{\sum_{w'=1}^{W} \log \hat{\theta}_{kw'}^{\text{comp}}}$$

making the influence of common words smaller (Rennie et al., 2003). The letter C below denotes complement modeling combined with log-normalization of parameters.

The heuristics described above, and others commonly used with multinomial models for text, modify both input data (word counts) and distribution parameters. Therefore, they do not give probability distributions that are properly normalized, i.e. that sum to one appropriately.

5.2. Document collections

The industry sector[1] data set contains 9555 documents distributed in 104 classes. The data set has a vocabulary of 55,055 words, and each document contains on average 606 words. The data are split into halves for training and testing. The 20 newsgroups[2] data set contains 18,828 documents belonging to 20 classes. This collection has a vocabulary of 61,298 words with an average document length of 116 words. The data are split into 80/20 fractions for training and testing. In the industry and newsgroup data sets each document belongs to one class only.

The Reuters-21578[3] data set contains 21,578 documents. We use the Mod Apte split which only contains 10,789 documents (Apte et al., 1994), those in the 90 classes with at least one training and one test example. The Mod Apte split uses a predefined set of 7,770 training documents and 3,019 test documents. The documents are multi-labeled and can belong to one or more of the 90 classes. This collection has a vocabulary of 15,996 words and the documents have an average length of 70 words.

5.3. Perplexity results

We start by evaluating the perplexity of alternative models over the same test data (Blei et al., 2003). When a document is represented as a vector of word counts, its probability includes a factor $n!/\prod_{w=1}^{W} x_w!$ that measures how many word sequences could generate the same vector of counts. We define perplexity over a set of D documents as

$$\exp\left(\frac{-\sum_{d=1}^{D}\sum_{w=1}^{W} \log p(x_{dw})}{\sum_{d=1}^{D} n_d}\right)$$

where $p(x)$ does not include the factor $n_d!/\prod_{w=1}^{W} x_{dw}!$. Perplexity on test data measures how well a model predicts unseen data. A lower value indicates better prediction.

The perplexity measure is calculated for the 20 newsgroups data, with one model trained for each of the 20 classes. The perplexity for multinomial models is 5311 ± 755 versus 2609 ± 382 for DCM models, where both results are means $\pm$ one standard deviation calculated over 10 random splits. We do not report perplexity results for heuristically modified multinomial models, since the transformed parameters and data no longer define a proper probability distribution that sums to one.

5.4. Classification results

The performance of the models is compared on the industry and newsgroup collections using precision $\frac{TP}{TP+FP}$ to measure the accuracy of classification. Here TP is the number of true positives, FP the number of false positives, and FN the number of false negatives.

With multi-labeled data, it is necessary to consider both precision and recall $\frac{TP}{TP+FN}$ to get a fair measure of performance. This is the case for the Reuters data. Following previous work, we combine precision and recall by computing the "break-even" point where precision equals recall. This point can be defined using either micro or macro averaging:

$$BE_{micro} = \frac{1}{N}\sum_{k=1}^{K} N_k \frac{TP_k}{TP_k + FP_k}$$

$$BE_{macro} = \frac{1}{K}\sum_{k=1}^{K} \frac{TP_k}{TP_k + FP_k}$$

where K is the number of document classes, N is the number of documents and N_k is the number of documents in class k. It is not always possible to get exactly the same value for precision and recall, so the average between the two measures is used in these cases. Using micro-averaging, every document is considered equally important. The macro-averaging measure penalizes classifiers that have poor performance on documents from rare classes.

We acknowledge that precision and break-even may not be the best measures of the effectiveness of a text classifier (Sebastiani, 2002), but we use these measures here for comparability with previous work. The results in Tables 1 and 2 are averages over 10 random splits (50/50 for industry sector and 80/20 for 20 newsgroups), shown $\pm$ one standard deviation σ over the 10 splits.

[1] `www.cs.umass.edu/~mccallum/code-data.html`
[2] `people.csail.mit.edu/people/jrennie/20Newsgroups`
[3] `kdd.ics.uci.edu`

Table 1. Classification results for the industry sector collection.

METHOD	SMOOTHING ε	PRECISION $\pm \sigma$
M	1	0.600 ± 0.011
L-M	1	0.654 ± 0.009
M	0.01	0.783 ± 0.008
DCM		0.806 ± 0.006
L-M	0.01	0.812 ± 0.005
TW-L-M	1	0.819 ± 0.004
TW-L-M	0.01	0.868 ± 0.005
C-M	1	0.889 ± 0.006
C-M	0.01	0.889 ± 0.004
C-L-M	0.01	0.899 ± 0.005
C-L-M	1	0.912 ± 0.005
C-DCM		0.917 ± 0.004
C-TW-L-M	0.01	0.919 ± 0.005
C-TW-L-M	1	0.921 ± 0.004

Table 2. Classification results for the 20 newsgroups collection.

METHOD	SMOOTHING ε	PRECISION $\pm \sigma$
M	0.01	0.853 ± 0.004
L-M	0.01	0.865 ± 0.005
TW-L-M	0.01	0.876 ± 0.005
C-M	0.01	0.876 ± 0.005
C-L-M	0.01	0.886 ± 0.005
DCM		0.890 ± 0.005
C-DCM		0.892 ± 0.004
C-TW-L-M	0.01	0.893 ± 0.005

Table 1 shows the performance of the different algorithms on the industry sector data set. Our results using multinomial-based methods are similar to those reported by Rennie et al. (2003) and McCallum and Nigam (1998). Smoothing with $\varepsilon = 0.01$ is clearly better than with $\varepsilon = 1$ for non-complement models. The DCM model produces results that are better than the multinomial and the complement-DCM produces results similar to the multinomial with all heuristics applied.

The results in Table 2 are obtained using the 20 newsgroups data. As in the industry sector data, the DCM model outperforms the multinomial. In this corpus, each class is represented by many examples, so complement modeling is not as useful and the DCM and complement-DCM models perform similarly to the best multinomial with heuristics. We show results with $\varepsilon = 0.01$ only because results with $\varepsilon = 1$ are worse, as in the industry sector data, and for compatibility with Rennie et al. (2003).

Table 3. Classification results for the Reuters collection. The third column shows macro break-even, while the last column shows micro break-even.

METHOD	SMOOTHING ε	MACRO BE	MICRO BE
M	1	0.268	0.761
L-M	1	0.303	0.756
DCM		0.359	0.740
TW-L-M	1	0.390	0.768
M	0.01	0.405	0.741
L-M	0.01	0.407	0.759
TW-L-M	0.01	0.456	0.753
C-TW-L-M	0.01	0.560	0.732
C-L-M	0.01	0.562	0.759
C-M	1	0.563	0.759
C-L-M	1	0.594	0.764
C-M	0.01	0.607	0.776
C-DCM		0.624	0.823
C-TW-L-M	1	0.657	0.840

Table 3 shows results on the Reuters corpus, which is special in that documents contain few words, and many classes only contain a few documents. The DCM and C-DCM methods still perform well. Standard deviations are not given since there is a single standard training set/test set split for this corpus.

We can evaluate the statistical significance of the differences in performance for the industry sector and 20 newsgroups collections. On these two collections, the DCM model outperforms the standard multinomial and a Student's t-test shows that this difference is extremely significant. The complement-DCM model performs slightly worse than the multinomial model with all heuristics applied. A t-test shows that for both data sets, the differences in performance between the complement-DCM model and C-TW-L-M method are not statistically significant.

6. Discussion

We have argued that the Dirichlet compound multinomial (DCM) model is a more appropriate generative model for text documents than the traditional multinomial model. The reason is that a DCM can model burstiness: the phenomenon that if a word appears once, it is more likely to appear again.

We have shown experimentally that the DCM model performs better than the multinomial model for two standard text mining tasks. First, as measured by perplexity, the DCM models a single collection of documents better. Second, when documents are classified using Bayes' rule using a generative model for each of

the alternative classes, accuracy using a DCM model for each class is higher than when using a multinomial model for each class. When the most effective known heuristics are applied in addition, accuracy using multinomial models versus using DCM models is similar.

The DCM model is a generative model for the documents within a class. Given a Dirichlet distribution, a document is not generated directly. Instead, the Dirichlet is used to generate a multinomial; this multinomial is then used to generate the document. Conceptually, different documents within the same class are generated by different multinomials. This procedure allows for diversity within the class. The words that are bursty in a particular document are those that have high probability in the particular multinomial used to generate this document.

A DCM model can represent a topic (i.e. a class of documents) where different documents use alternative terminology. For example, some automotive documents may use the word "hood" while others use the word "bonnet." This within-topic diversity is different from the within-document diversity allowed by latent topic modeling, where each topic is represented by a single multinomial, but each word in a document may be generated by a different topic (Deerwester et al., 1990; Hofmann, 1999; Blei et al., 2003).

We hope that many applications of text modeling in addition to those outlined in this paper will benefit from using DCM models in the future.

Acknowledgments. This paper is based in part on work sponsored by the Air Force Research Laboratory under contract F30602-03-C-0075, performed in conjunction with Lockheed Martin Information Assurance. This research was also supported in part by Fair Isaac, Inc. and by Sun Microsystems, Inc.

References

Aizawa, A. (2003). An information-theoretic perspective of tf-idf measures. *Information Processing and Management*, *39*, 45–65.

Apte, C., Damerau, F. J., & Weiss, S. M. (1994). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, *12*, 233–251.

Blei, D., Ng, A., & Jordan, M. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Church, K. W., & Gale, W. A. (1995). Poisson mixtures. *Natural Language Engineering*, *1*, 163–190.

Deerwester, S., Dumais, S., Landauer, T., Furnas, G., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, *41*, 391–407.

Hofmann, T. (1999). Probabilistic latent semantic indexing (PLSI). *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval* (pp. 50–57). Berkeley, California: ACM.

Jansche, M. (2003). Parametric models of linguistic count data. *41st Annual Meeting of the Association for Computational Linguistics* (pp. 288–295). Sapporo, Japan.

Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, *28*, 11–21.

Katz, S. M. (1996). Distribution of content words and phrases in text and language modelling. *Natural Language Engineering*, *2*, 15–59.

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. *Proceedings of ECML-98, 10th European Conference on Machine Learning* (pp. 4–15). Chemnitz, Germany: Springer Verlag, Heidelberg, Germany.

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification. *AAAI/ICML-98 Workshop on Learning for Text Categorization* (pp. 41–48). AAAI Press.

McCallum, A. K. (1996). *Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering.* www.cs.cmu.edu/~mccallum/bow.

Minka, T. (2003). *Estimating a Dirichlet distribution.* www.stat.cmu.edu/~minka/papers/dirichlet.

Rennie, J. D. M., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the poor assumptions of naive Bayes text classifiers. *Proceedings of the Twentieth International Conference on Machine Learning.* Washington, D.C., US: Morgan Kaufmann Publishers, San Francisco, US.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*, 613–620.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*, 1–47.

Teevan, J., & Karger, D. R. (2003). Empirical development of an exponential probabilistic model for text retrieval: Using textual analysis to build a better model. *Proceedings of the 26th Annual ACM Conference on Research and Development in Information Retrieval (SIGIR '03).* Toronto, CA.

Zipf, G. (1949). *Human behaviour and the principle of least effort: An introduction to human ecology.* Addison-Wesley.

Appendix C

Vocabulary Pruning for Improved Context Recognition

Pruning The Vocabulary For Better Context Recognition

Rasmus Elsborg Madsen, Sigurdur Sigurdsson, Lars Kai Hansen and Jan Larsen
Informatics and Mathematical Modelling, Technical University of Denmark
Richard Petersens Plads, Building 321, DK-2800 Kongens Lyngby, Denmark
web: www.imm.dtu.dk, e-mail: rem,siggi,lkh,jl@imm.dtu.dk

***Abstract*— Language independent 'bag-of-words' representations are surprisingly effective for text classification. The representation is high dimensional though, containing many non-consistent words for text categorization. These non-consistent words result in reduced generalization performance of subsequent classifiers, e.g., from ill-posed principal component transformations. In this communication our aim is to study the effect of reducing the least relevant words from the bag-of-words representation. We consider a new approach, using neural network based sensitivity maps and information gain for determination of term relevancy, when pruning the vocabularies. With reduced vocabularies documents are classified using a latent semantic indexing representation and a probabilistic neural network classifier. Reducing the bag-of-words vocabularies with 90%-98%, we find consistent classification improvement using two mid size data-sets. We also study the applicability of information gain and sensitivity maps for automated keyword generation.**

I. Introduction

The world wide web is an unstructured and fast growing database. Today's search tools often leave web users in frustration by the low precision and recall [6]. It is widely believed that machine learning techniques can come to play an important role in web search. Ambitious plans have been launched for supporting intelligent use of the web, i.e., a "semantic web" [4]. IBM's WebFountain [5] and the Stanford University semantic web platform TAP [13] are examples of machine learning methods coming into play, making human web navigation easier. Here we consider web content mining in the form of internet document classification - an information retrieval (IR) aspect of web-mining [17]. Internet documents contain text, hyper-links, meta-data, images, and other multimedia content which can be used for classification [17], [16]. This paper focuses on classification based on text part, i.e., text categorization. Text categorization is the process of creating a supervised automatic text classifier, by means of machine learning techniques. The classifier labels documents from the corpus $\mathcal{D} = [d_1, \cdot, d_j, \cdot, d_{|\mathcal{D}|}]$ into a set of classes $\mathcal{C} = [c_1, \cdot, c_k, \cdot, c_{|\mathcal{C}|}]$, based on an initial set of labeled documents.

Generic text categorization systems are based on the bag-of-words representation, which is surprisingly effective for the task. In the bag-of-words representation we summarize documents by their term histograms. The main motivation for this reduction (removing the semantics) is that it is easily automated and needs minimal user intervention beyond filtering of the term list. The term list typically contains in the range of $10^3 - 10^5$ terms, hence further reduction is necessary for most pattern recognition devices. Latent semantic indexing (LSI) [12], [11] aka principal component analysis is often used to construct low dimensional representations. LSI is furthermore believed to reduce synonymy and polysemy problems [11], [19]. Synonymy is when multiple words have the same meaning and polysemy is when a single word have multiple meanings. Although LSI and other more elaborate vector space models have been successful in text classification in small and medium size databases, see e.g., [16], [14], it is still not at human level text classification performance. When training classifiers on relatively small databases generalizability is a key issue. How well does a model adapted on one set of data predict the labels of another test data set? Generalizability is in general a function of the number of training cases and of the effective model dimension.

Various methods and techniques have been purposed to improve generalization in text categorization. WordNet [2], a lexical database containing synonym sets and other lexical concept, has been used for classification improvement. In [9], the synonymy part in WordNet has been used to expand term-lists for each text category, enhancing the accuracy of the text classifier significantly. In [15], text classification based WordNet's word meanings has been attempted. These experiments have not given any significant classification accuracy enhancement. On the other hand, the use of words part-of-speech (POS) has showed to improve text categorization generalization. A POS-tagger analyzes sentences and tag words with their part-of-speech, i.e., noun, verb, adverb, number, punctuation, etc. In [3], words have been tagged with their POS, avoiding confusion between similar words with different meanings. This approach resulted in a positive effect on classification accuracy. In [1], a POS-tagger has been used to extract more than 3.000.000 compound words from texts, improving classification accuracy. Using unlabeled documents when categorizing texts has an improving effect. The unlabeled documents can be used in various ways, see e.g. [24], [20] and [31]. In [18], multiple classifiers are combined, and a consensus voting scheme among the classifiers performs better than any single classifier.

In this communication, the aim is to improve generalizability of the supervised document classifier by pruning the document vocabulary $\mathcal{T} = [t_1, \cdot, t_i, \cdot, t_{|\mathcal{T}|}]$, i.e., removing the term which is least suited for discrimination. Many terms posses

little or no generalizable discriminative power, and should be regarded as noise. Pruning the vocabulary, the task is to determine the least discriminative terms, based on the training set only. Automated vocabulary reduction has been attempted with success previously in [30], using a k-nearest-neighbors classifier. We here use another method for term reduction, and experiment within the LSI representation. To estimate term relevance we will use Information Gain and scaled sensitivity, which is computed using the so-called NPAIRS split-half re-sampling procedure [29]. Our hypothesis is that sensitivity maps can determine which terms are consistently important, hence, likely to be of general use for classification relative to terms that are of low or highly variable sensitivity.

The rest of this article is organized as follows. In Section II, we discuss the generic bag-of-words approach for text categorization, and the vocabulary pruning methods. In Section III, explains the data sets used for the experiments. Section IV presents the results obtained using vocabulary pruning. Section V concludes on the methods and results.

II. Methods

Using the generic bag-of-words approach, documents are arranged in a term-document matrix $\mathbf{X}$, where $X_{i,j}$ is the number of times term i occur in document j. The dimensionality of $\mathbf{X}$ is reduced by filtering and stemming. Stemming refers to a process in which words with different endings are merged, e.g., 'train', 'trained' and 'training' are merged into the common stem 'train'. This example also indicates the main problem with stemming, namely that it introduces an artificial increased polysemy. We have decided to 'live with this problem' since without stemming vocabularies would grow prohibitively large. About 500 common non-discriminative stop-words, i.e. ('a', 'i', 'and', 'an', 'as', 'at') are removed from the term list. In addition high and low frequency words are also removed from the term list. The term-document matrix can be normalized in various ways. In [10] experiments with different term weighting schemes are carried out. The term frequency / inverse document frequency (TFIDF) weighting is consistently good among term weighting methods purposed, and is the method generally used. After TFIDF normalization the resulting elements in $\mathbf{X}$ becomes

$$X_{i,j}^{\text{tfidf}} = X_{i,j}^{\text{tf}} \log \frac{|\mathcal{D}|}{DF_i} \tag{1}$$

where DF_i is the document frequency of term i and $X_{i,j}^{\text{tf}}$ is the log normalized term frequency.

$$X_{i,j}^{\text{tf}} = \begin{cases} 1 + \log(X_{i,j}) & \text{if} X_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The length of the documents is often a good prior for predicting the content within a little corpora. While document length might be a solid variable within the corpora, it is likely that this is not generally a valid parameter. The length of the documents is usually normalized to prevent the influence the document length might have. The Frobenius norm is used to length normalize the term document matrix to one.

$$X_{i,j}^{\text{n2tfidf}} = \frac{X_{i,j}^{\text{tfidf}}}{\sqrt{|\mathcal{T}|^{-1} \sum_{i'=1}^{|\mathcal{T}|} X_{i',j}^{\text{tfidf}^2}}} \tag{3}$$

To emphasize the influence of document lengths, the distribution of the term standard deviations for the spam and non-spam documents, in the email data-set, are illustrated in Figure 1,

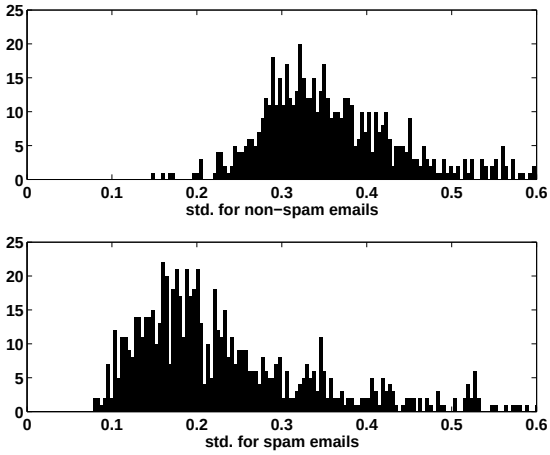


Fig. 1. Distribution of the standard deviation for the email data-set. The distribution for the spam class and the non-spam class varies a lot. The standard deviation is a good discriminator, but probably not general outside this data-set. Using only the standard deviation for classification, the generalization error is 22%.

Using only the standard deviation measure for classification, 78% of the documents can be classified correctly. This clearly shows that document length is a good prior.

It is suggested to use a reduced normalized vocabulary, using sensitivity maps and information gain. The reduction factor ξ determines the fraction of the vocabulary, which is removed.

$$\xi = \frac{|\mathcal{T}| - |\mathcal{T}'|}{|\mathcal{T}|} \tag{4}$$

Where $\mathcal{T}'$ is the new vocabulary, a subset of the full vocabulary $\mathcal{T}$.

Using sensitivity maps for pruning, we use the definition of class specific sensitivity proposed in [32], [28] for a set of N samples,

$$\mathbf{s}_k = \frac{1}{N} \sum_{n=1}^{N} \left| \frac{\partial P(c_k|\mathbf{f}_n)}{\partial \mathbf{x}} \right| \tag{5}$$

and where $P(c_k|\mathbf{f}_n)$ is the posterior probability of class k given the feature vector $\mathbf{f}_n$. $\mathbf{s}_k$ is the K-dimensional sensitivity vector for class k. The K-dimensional derivative is obtained using the projection (8) [28]. A split-half re-sampling procedure is invoked to determine the statistical significance of the

sensitivity [29]. Multiple splits are generated of the original training set and classifiers trained on each of the splits. For each classifier a sensitivity map is computed. Since the two maps obtained from a given split are exchangeable the mean map is an unbiased estimate of the 'true' sensitivity map, while the squared difference is a noisy, but unbiased estimate of the variance of the sensitivity map. By repeated re-sampling and averaging the sensitivity map and its variance are estimated. We finally obtain a scaled sensitivity map by normalization trough the standard deviation.

The scaled sensitivity way of pruning will be compared with information gain pruning. The information gain [30] for the term t_i is defined as:

$$\begin{aligned} IG_{t_i} = & -\sum_{k=1}^{|\mathcal{C}|} P(c_k) \log P(c_k) \\ & + P(t_i) \sum_{k=1}^{|\mathcal{C}|} P(c_k|t_i) \log P(c_k|t_i) \\ & + P(\bar{t}_i) \sum_{k=1}^{|\mathcal{C}|} P(c_k|\bar{t}_i) \log P(c_k|\bar{t}_i), \end{aligned} \tag{6}$$

where $P(t_i)$ is the probability that term t_i appears at least once in a document and $P(\bar{t}_i)$ is the probability that the term does not appear in a document.

The normalized and pruned term document matrix $\mathbf{X}_\mathrm{p}$ is reduced to a feature-document matrix using PCA, carried out by an 'economy size' singular value decomposition,

$$\mathbf{X}_\mathrm{p} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T. \tag{7}$$

Where the orthogonal $|\mathcal{T}'| \times |\mathcal{D}|$ matrix $\mathbf{U}$ contains the eigenvectors corresponding to the non-zero eigenvalues of the symmetric matrix $\mathbf{X}_\mathrm{p}\mathbf{X}_\mathrm{p}^T$. $\mathbf{\Lambda}$ is a $|\mathcal{D}| \times |\mathcal{D}|$ diagonal matrix of singular values ranked in decreasing order and the $|\mathcal{D}| \times |\mathcal{D}|$ matrix $\mathbf{V}^T$ contains eigenvectors of the symmetric matrix $\mathbf{X}_\mathrm{p}^T\mathbf{X}_\mathrm{p}$. The LSI representation is obtained by projecting document histograms on the basis vectors in $\mathbf{U}$,

$$\mathbf{F} = \mathbf{U}^T\mathbf{X} = \mathbf{\Lambda}\mathbf{V}^T. \tag{8}$$

Typically, the majority of the singular values are small and can regarded as noise. Consequently, only a subset of K ($K < |\mathcal{T}'|$) features is retained as input to the classification algorithm. The representational potential of these LSI features is illustrated in Figure 2.

A wide variety of classification algorithms have been applied to the text categorization problem, see e.g., [17]. We have extensive experience with probabilistic neural network classifiers and a well tested ANN toolbox is available [26]. The toolbox adapts the network weights and tunes complexity by adaptive regularization and outlier detection using the Bayesian ML-II framework, hence, requires minimal user intervention [27].

III. Data

Two data-sets, 'Email' [23] and 'WebKB' [7] are used to illustrate and test the hypothesis. No less than ten split-half re-samples are used in all experiments. The Email data-set consists of texts from 1431 emails in three categories: conference (370), job (272) and spam (789). The WebKB set

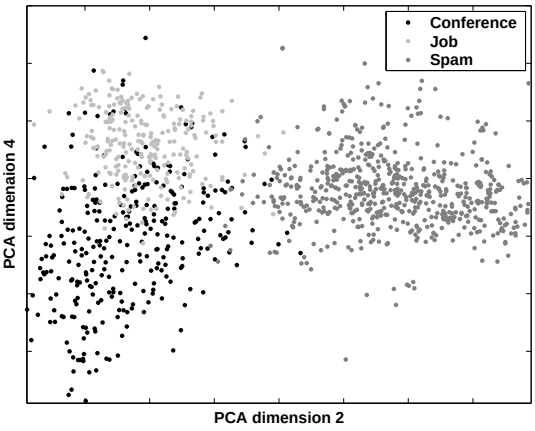


Fig. 2. Illustration of the document distribution in feature space. Here we show the Email corpus projected onto the 2nd and 4th principal directions. In this projection the 'spam' class is well separated while the two other classes in the set ('conferences' and 'jobs') show some overlap.

contains 8282 web-pages from US university computer science departments. Here we have used a subset [8] of 2240 pages from the WebKB earlier used in [14] and [19]. The WebKB categories are: project (353), faculty (483), course (553) and student (851). All html tags were removed from the data-set.

IV. Results

The standard performance measure for text categorization systems is precision and recall. Precision measures how many of the retrieved entries are relevant precision $=$ true positive$/$(true positive $+$ false positive). Recall measures how many relevant entries were found compared to the amount of relevant entries in the collection recall $=$ true positive$/$(true positive $+$ false negative. The F_β measure [21] weights the importance of precision and recall, where $\beta = 1$ weights precision and recall equally,

$$F_\beta = \frac{(\beta^2 + 1) \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}} \tag{9}$$

Micro averaging [25] over all the classes $|\mathcal{C}|$, is rewarded when classifiers of frequent categories performs well. When each document belongs to less than two classes, which is the case for the collections considered here, the micro averaged precision and recall simplifies to the fraction of correct classified documents. It follows from that the F_1 measure also becomes the fraction of correct classified documents. In the following we use the error function defined as $1 - F_1$.

Preprocessing the documents, all letters in all the terms have been converted to lowercase and punctuations have been removed. A simple stemmer has transformed words with basic endings into their common stem. Preliminary experiments indicated that a reduced feature space of $K = 48$ projections and a neural network classifier with five hidden units were sufficient for the task (data not shown). All results have been

validated using 10 fold split half re-sampling cross validation. The neural network based term sensitivity is a function of the given training set. Terms for which the sensitivity is high but also highly variable are less likely to support generalizability compared to terms that have a consistent high or medium sensitivity. The empirical distribution of mean and standard deviations the terms sensitivities of the Email set are shown in Figure 3.

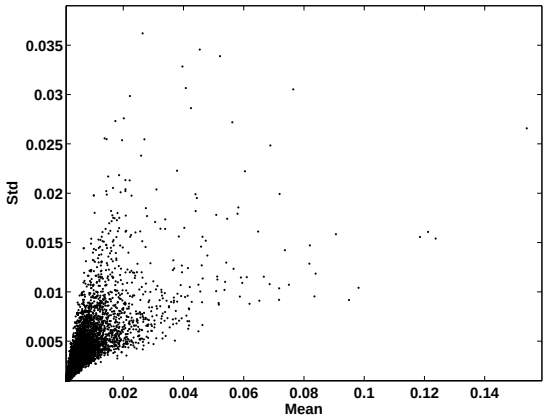


Fig. 3. Mean and standard deviation of the term sensitivity. The most relevant terms have consistently high sensitivity in each re-sampling split, i.e., a high mean and relatively low standard deviation. These terms occupy the lower right part of the plot.

The empirical scaled sensitivities $Z_i = \mu_i/\sigma_i$ of the terms t_i were used to determine the term relevance. Term relevance were also determined using information gain. The two methods are quite different, and so is the distribution of their estimated term relevance. In Figure 4, the distribution of term relevance is shown for the Email data, using the two methods.

Both relevance measure distributions have large slender tails, showing that few terms posses much information. It is likely that the vocabularies can be pruned intensively.

Based on the scaled sensitivities, relevant keywords for the text categories has been extracted. For the Email data the five highest scores for the Conference category are (*Paper, Conference, Deadline, Neural, Topic*) and for the Job category (*Research, Position, Candidate, University, Edt*) and for the Spam category (*Money, Remove, Free, Thousand, Simply*). Similarly, information gain has been used to determine relevant keywords for the Email data. The five highest scores for the Conference category are (*Neural, Conference, Paper, Science, Workshop*) and for the Job category (*University, Research, Candidate, Computational, Position*) and for the Spam category (*Money, Free, Remove, Business, Simply*). The two sets of keywords posses high relevancy for the three classes, though the two methods does not find the exact same keywords.

The vocabularies of the WebKB and Email data are pruned with an increasing reduction factor ξ. The generalization error as function of $1 - \xi$ is shown in Figure 5.

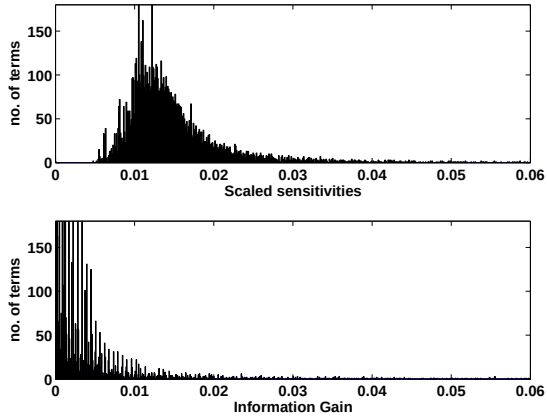


Fig. 4. Distribution of term relevance scores, using scaled sensitivities and information gain, for the Email data. Both distributions have large slender tails, indicating that few terms posses much higher relevancy than others, and intensive pruning should be performed. 10% of the terms have a scaled sensitivity higher than 0.025 and 10% of the terms have information gain higher than 0.008.

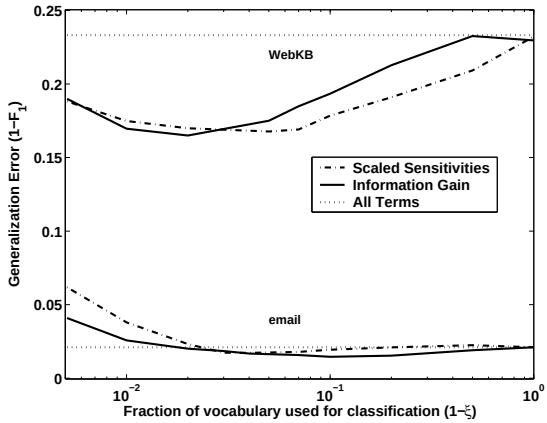


Fig. 5. Generalization error pruning the vocabulary for the Email and WebKB data, using scaled sensitivities and information gain as term relevance measure. Pruning with use of information gain gives slightly better generalization error than when using scaled sensitivities. Reducing the vocabulary with 90%, using Information gain, is optimal for the Email data-set. The generalization error is then reduced with 26%. For the WebKB data-set the lowest generalization error is found, reducing the vocabulary with 98%, where the error is reduced with 29%. The results were found using 20% of the samples for training.

Using all the terms, the generalization classification error rate is 23.3% in the WebKB and 2.1% in Email data. Removing respectively 98% and 90% of the vocabularies with the lowest information gain, the generalization error for the WebKB is reduced to 16.5% and to 1.5% for the Email data. Removing terms with the lowest information gain, the performance is slightly better than when using scaled sensitivities for term removal.

In Figure 6 we show that learning curves are consistently improved for a range of training sets for the WebKB and the Email data, based on a fixed reduction of respectively 98% and 90% of their original vocabulary.

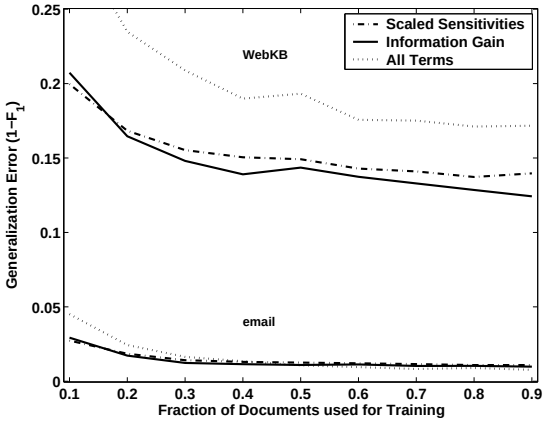


Fig. 6. Learning curves using full and pruned vocabularies. Learning curves shows decreased generalization error for a range of training set sizes. For the WebKB, both pruning methods shows consistent reduced generalization error of about 25% for the whole range of training set sizes. For the Email data, pruning decreases the generalization error when using less than 40% of the data-set for training. When 40% or more of the data-set samples are used for training, the generalization error is not reduced further. Noise within the data might prevent classification from further optimization.

Pruning the vocabulary to a small fraction of the original sizes, results in better generalization in the whole range of training-set sizes, however, a somewhat larger effect for small training sets. For both data-sets, pruning lowers the generalization error with approximately 25%. For the Email set, generalization error is not lowered using 40% of the data or more for training. It is likely that noise within the data-set prevents the classifier from lowering the generalization error any further. For both data-sets information gain is generally slightly better than scaled sensitivities, at determining which terms are relevant for classification. Information gain is significantly cheaper to compute than the scaled sensitivities, which make them the obvious choice among the two methods.

V. Conclusion

Neural network sensitivity maps were introduced in a LSI based context recognition framework. Scaled sensitivity information gain were compared for vocabulary pruning. Using two mid-size data-sets, both methods have consistently shown reduction in text classification error when pruning the vocabularies. Both methods lower the generalization error by approximately 25% over a range of training set sizes. Information gain is generally better at determining the relevant vocabulary information, resulting in slightly better generalization error relative to using scaled sensitivities. Finally, we noted that information gain and the scaled sensitivity are also useful for identifying class specific keywords.

Acknowledgment

The work is supported by the European Commission through the sixth framework IST Network of Excellence: Pattern Analysis, Statistical Modelling and Computational Learning (PASCAL), contract no. 506778.

References

[1] A. Aizawa. Linguistic techniques to improve the performance of automatic text categorization. In *Proceedings of NLPRS-01, 6th Natural Language Processing Pacific Rim Symposium*, pages 307–314, Tokyo, JP, 2001.

[2] Cognitive Science Laboratory at Princeton University. Wordnet 2.0. http://www.cogsci.princeton.edu/ wn/, 2003.

[3] R. Basili, A. Moschitti, and M.T. Pazienza. NLP-driven IR: Evaluating performances over a text classification task. In Bernhard Nebel, editor, *Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence*, pages 1286–1291, Seattle, US, 2001.

[4] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 2001.

[5] IBM Almaden Research Center. The WebFountain. http://www.almaden.ibm.com/webfountain/publications/.

[6] S. Chakrabarti. Data mining for hypertext: a tutorial survey. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, 1:1–11, 2000.

[7] CMU-WebKB. The 4 universities data set. http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/, 1997.

[8] CMU-WebKB-2240. A subset of the webkb. http://www.imm.dtu.dk/∼rem/, 1999.

[9] M. De Buenaga Rodríguez, José María Gómez-Hidalgo, and Belén Díaz-Agudo. Using WordNet to complement training information in text categorization. In Ruslan Milkov, Nicolas Nicolov, and Nilokai Nikolov, editors, *Proceedings of RANLP-97, 2nd International Conference on Recent Advances in Natural Language Processing*, Tzigov Chark, BL, 1997.

[10] F. Debole and F. Sebastiani. Supervised term weighting for automated text categorization. In *Proceedings of SAC-03, 18th ACM Symposium on Applied Computing*, pages 784–788, Melbourne, US, 2003. ACM Press, New York, US.

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

[12] G.W. Furnas, S. Deerwester, S.T. Dumais, T.K. Landauer, R.A. Harshman, L.A. Streeter, and K.E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. In *The 11th International Conference on Research and Development in Information Retrieval*, pages 465–480, Grenoble, France, 1988. ACM Press.

[13] R. Guha and R. McCool. Tap: A semantic web platform. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 42:557–577, 2003.

[14] L.K. Hansen, S. Sigurdsson, T. Kolenda, F.A. Nielsen, U. Kjems, and J. Larsen. Modeling text with generalizable gaussian mixtures. In *International Conference on Acoustics, Speech and Signal Processing*, pages 3494–3497. IEEE, 2000.

[15] Athanasios Kehagias, Vassilios Petridis, Vassilis G. Kaburlasos, and Pavlina Fragkou. A comparison of word- and sense-based text categorization using several classification algorithms. *Journal of Intelligent Information Systems*, 21(3):227–247, 2003.

[16] T. Kolenda, L.K. Hansen, J. Larsen, and O. Winther. Independent component analysis for understanding multimedia content. In S. Bengio J. Larsen H. Bourlard, T. Adali and S. Douglas, editors, *Proceedings of IEEE Workshop on Neural Networks for Signal Processing XII*, pages 757–766, Piscataway, New Jersey, 2002. IEEE Press.

[17] R. Kosala and H. Blockeel. Web mining research: A survey. In *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, pages 1–15. ACM Press, 2000.

[18] L.S. Larkey and W.B. Croft. Combining classifiers in text categorization. In Hans-Peter Frei, Donna Harman, Peter Schäuble, and Ross Wilkinson, editors, *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*, pages 289–297, Zürich, CH, 1996. ACM Press, New York, US.

[19] J. Larsen, L.K. Hansen, A.S. Have, T. Christiansen, and T. Kolenda. Webmining: learning from the world wide web. *Computational Statistics and Data Analysis*, 38:517–532, 2002.

[20] J. Larsen, A. S. Have, and Hansen L. K. Probabilistic hierarchical clustering with labeled and unlabeled data. *International Journal of Knowledge-Based Intelligent Engineering Systems*, 6:56–62, 2002.

[21] D.D. Lewis and W.A. Gale. A sequential algorithm for training text classifiers. In W. Bruce Croft and Cornelis J. van Rijsbergen, editors, *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 3–12, Dublin, IE, 1994. Springer Verlag, Heidelberg, DE.

[22] R.E. Madsen and L.K. Hansen. Part-of-speech enhanced context recognition. In *The 17th international conference on pattern recognition*, Cambridge, UK, 2004.

[23] F.Å. Nielsen. Email data-set. http://www.imm.dtu.dk/∼rem/, 2001.

[24] K. Nigam, A.K. McCallum, S. Thrun, and T.M. Mitchell. Learning to classify text from labeled and unlabeled documents. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence*, pages 792–799, Madison, US, 1998. AAAI Press, Menlo Park, US.

[25] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47, 2002.

[26] S. Sigurdsson. The dtu: Artificial neural network toolbox. http://mole.imm.dtu.dk/toolbox/ann/, 2002.

[27] S. Sigurdsson, J. Larsen, L.K. Hansen, P. A. Philipsen, and H. C. Wulf. Outlier estimation and detection: Application to skin lesion classification. In *International conference on acoustics, speech and signal processing*, pages 1049–1052, 2002.

[28] S. Sigurdsson, P.A. Philipsen, L.K. Hansen, J. Larsen, M. Gniadecka, and H.C. Wulf. Detection of skin cancer by classification of Raman spectra. *Accepted for IEEE Transactions on Biomedical Engineering*, 2004.

[29] S.C. Strother, J Anderson, L.K. Hansen, U. Kjems, R. Kustra, J. Siditis, S. Frutiger, S. Muley, S. LaConte, and D. Rottenberg. The quantitative evaluation of functional neuroimaging experiments: The NPAIRS data analysis framework. *Neuroimage*, 15:747–771, 2002.

[30] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 412–420, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.

[31] S. Zelikovitz and H. Hirsh. Using LSI for text classification in the presence of background text. In Henrique Paques, Ling Liu, and David Grossman, editors, *Proceedings of CIKM-01, 10th ACM International Conference on Information and Knowledge Management*, pages 113–118, Atlanta, US, 2001. ACM Press, New York, US.

[32] J.M. Zurada, A. Malinowski, and Cloete I. Sensitivity analysis for minimization of input data dimension for feedforward neural network. In *Proceedings of the IEEE Symposium on Circuits and Systems*, pages 447–450, 1994.

Appendix D

Part-of-Speech Enhanced Context Recognition

PART-OF-SPEECH ENHANCED CONTEXT RECOGNITION

Rasmus Elsborg Madsen, Jan Larsen and Lars Kai Hansen
Department of Mathematical Modeling, Building 321
Technical University of Denmark, DK-2800 Lyngby, Denmark
Phone: +45 4525 3894
Fax: +45 4587 2599
E-mail: rem,jl,lkh@imm.dtu.dk
Web: isp.imm.dtu.dk

Abstract. Language independent 'bag-of-words' representations are surprisingly effective for text classification. In this communication our aim is to elucidate the synergy between language independent features and simple language model features. We consider term tag features estimated by a so-called part-of-speech tagger. The feature sets are combined in an early binding design with an optimized binding coefficient that allows weighting of the relative variance contributions of the participating feature sets. With the combined features documents are classified using a latent semantic indexing representation and a probabilistic neural network classifier. Three medium size data-sets are analyzed and we find consistent synergy between the term and natural language features in all three sets for a range of training set sizes. The most significant enhancement is found for small text databases where high recognition rates are possible.

INTRODUCTION

The World Wide Web is a huge, unstructured, and fast growing database, but web users are often left in frustration by the low precision and recall of today's search tools [6]. It is widely believed that machine learning techniques will play an important role in creating more efficient searches. Ambitious plans have been launched for supporting intelligent use of the web, i.e., a "semantic web" [4]. IBM's WebFountain [5] and the Stanford University semantic web platform TAP [13] are examples of machine learning methods coming into play, making human web navigation easier. Here we consider web content mining in the form of internet document classification - an information retrieval (IR) aspect of web-mining [18]. Internet documents contain text, hyper-links, meta-data, images, and other multimedia content

which can be used for classification [18, 17]. This paper focuses on classification based on text part, i.e., text categorization. Text categorization is the process of creating a supervised automatic text classifier, by means of machine learning techniques. The classifier labels documents from the corpus $\mathcal{D} = [d_1, \cdot, d_j, \cdot, d_{|\mathcal{D}|}]$ into a set of classes $\mathcal{C} = [c_1, \cdot, c_k, \cdot, c_{|\mathcal{C}|}]$, based on an initial set of labeled documents.

Generic text categorization systems are based on the bag-of-words representation, which is surprisingly effective for the task. In the bag-of-words representation we summarize documents by their term histograms. The main motivation for this reduction (removing the semantics) is that it is easily automated and needs minimal user intervention beyond filtering of the term list. The term list typically contains in the range of $10^3 - 10^5$ terms, hence further reduction is necessary for most pattern recognition devices. Latent semantic indexing (LSI) [12, 11] aka principal component analysis is often used to construct low dimensional representations. LSI is furthermore believed to reduce synonymy and polysemy problems [11, 19]. Synonymy is when multiple words have the same meaning and polysemy is when a single word have multiple meanings. Although LSI and other more elaborate vector space models have been successful in text classification in small and medium size databases, see e.g., [17, 14], it is still not at human level text classification performance. When training classifiers on relatively small databases generalizability is a key issue. How well does a model adapted on one set of data predict the labels of another test data set? Generalizability is in general a function of the number of training cases and of the effective model dimension.

In this communication our aim is to understand the role of natural language features for classification. Specifically, we are interested in the role of term characteristics as derived by natural language processing (NLP). We have chosen the so-called QTAG [20] part-of-speech (POS)-tagger to estimate term characteristics. Synergy of bag-of-words features and POS-features will be evaluated by the effects their combination has on document classification rates. We will use 'early binding' combining the feature sets prior to LSI projection.

NLP features have been used for document classification in a number of studies. In the so-called WordNet system [9] synonymy features were used to expand term-lists for each text category. This strategy enhanced the accuracy of the text classifier significantly. Limited improvements were obtained by invoking semantic features from WordNet's lexical database [15]. In [3] and [2] enhanced classification ability was reported by the use of POS-tagged terms to avoid the confusion from polysemy. In [1] a POS-tagger was used to extract more than $3.0 \cdot 10^6$ compound terms in a database. A classifier based on the extended term list showed improved classification rates.

METHODS

The documents are arranged in a term document matrix $\mathbf{X}$, where $X_{i,j}$ is the number of times term i occur in document j. The dimensionality of $\mathbf{X}$ is reduced by filtering and stemming. Stemming refers to a process in which words with different endings are merged, e.g., 'trained' and 'training' are merged into the common stem 'train'. About 500 common non-discriminative stop-words, i.e. (a, i, and, an, as, at), are removed by filtering. In addition high and low frequency words are also removed from the term list. The term-document matrix can be normalized in various ways. In [10] experiments with different term weighting schemes are carried out. The term frequency / inverse document frequency (TFIDF) weighting is consistently good among term weighting methods purposed, and is the method generally used. After TFIDF normalization the resulting elements in $\mathbf{X}$ becomes

$$X_{i,j}^{\text{tfidf}} = X_{i,j}^{\text{tf}} \log \frac{|\mathcal{D}|}{DF_i} \tag{1}$$

where DF_i is the document frequency of term i and $X_{i,j}^{\text{tf}}$ is the log normalized term frequency.

$$X_{i,j}^{\text{tf}} = \begin{cases} 1 + \log(X_{i,j}) & \text{if} X_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The length of the documents is often a good prior for predicting the content within a little corpora. While document length might be a solid variable within the corpora, it is likely that this is not generally a valid parameter. The length of the documents is usually normalized to prevent the influence the document length might have. The Frobenius norm is used to length normalize the term document matrix to one.

$$X_{i,j}^{\text{n2tfidf}} = \frac{X_{i,j}^{\text{tfidf}}}{\sqrt{|\mathcal{T}|^{-1} \sum_{i'=1}^{|\mathcal{T}|} X_{i',j}^{\text{tfidf}^2}}} \tag{3}$$

We use POS-tags in a design similar to the bag-of-words representation. A tag-document matrix $\mathbf{Y}$ is generated, where Y_{gj} is the number of times tag g occur in document j. The POS-tagger analyzes all sentences in the documents and words part-of-speech function is determined, i.e. noun, verb, adverb, number, punctuation, etc. The POS-tagger distinguishes between 90 different tags. The tagging accuracy of QTAG is approximately 97% [22]. The tag document matrix is normalized as the term document matrix.

Feature set combination is often referred to as 'binding' in analogy with the ability of human brain to bind multiple features for enhanced pattern recognition. Binding can be achieved at different levels. In 'early binding' features are combined in the pre-processing steps. Early binding of feature sets with different statistics and based on variance decomposition requires determination of the relative weights of the participating feature sets. One possibility would be to use variance decomposition based on factor analysis

which is insensitive to relative scaling of variables. For simplicity, we have chosen to introduce a single binding coefficient α which can be tuned for each corpus separately,

$$\mathbf{Z} = \begin{bmatrix} \alpha\mathbf{X} \\ (1-\alpha)\mathbf{Y} \end{bmatrix} \tag{4}$$

If $\alpha \approx 0$ variance is dominated by tag features while when $\alpha \approx 1$ term features dominate.

The combined matrix $\mathbf{Z}$ is reduced to a feature-document matrix using LSI. The reduced dimension features are found by projecting the matrix $\mathbf{Z}$, onto a set of orthogonal basis vectors found by singular value decomposition $\mathbf{Z} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^{\mathbf{T}}$. A wide variety of classification algorithms have been applied

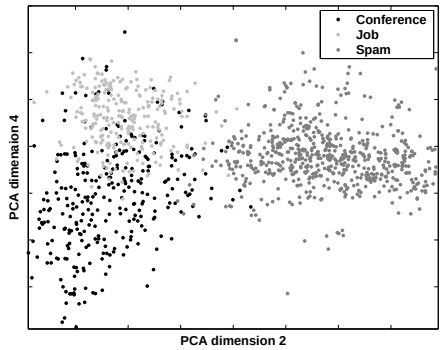


Figure 1: Illustration of the document distribution in feature space. Here we show the email corpus projected onto the 2nd and 4th principal directions. In this projection the 'spam' class is well separated while the two other classes in the set ('conferences' and 'jobs') show some overlap.

to the text mining problem, see e.g., [18]. We have extensive experience with probabilistic neural network classifiers and a well tested ANN toolbox is available[24]. The toolbox adapts the network weights and tunes complexity by adaptive regularization using the Bayesian ML-II framework, hence, requires minimal user intervention [25]. According to [23], this baseline method is among the best for text classification.

DATA

We measure the synergy of term and POS features in three corpora: Email [21], Multimedia [16] [17] and webKB [7]. The data-sets have been split into training and test sets and have been re-sampled for statistical verification of the results. 10 splits were used in all experiments. The email set consists of texts from 1431 emails manually classified in three categories: Conference (370), job (272) and spam (789). The multimedia corpus consists of texts

and images from 1200 web pages. Only the text part is considered here. The categories are: Sports (400), aviation (400) and paintball (400). The WebKB contains 8282 web-pages from various universities computer science departments. We use a subset of the corpus extracted in [8], and used in [14] and [19], containing 2240 pages. The categories are: Project (353), faculty (483), course (553) and student (851). All 'html' tags were removed from the corpus in this investigation. The multimedia data has a relatively small vocabulary with only 3500 terms after preprocessing. The email data has 9500 terms, and the WebKB data has 13000 terms after preprocessing. The POS-tag features represent a space which is smaller than the term space by a factor of 40-140 for the three data-sets.

RESULTS

Preliminary experiments indicated that a reduced feature space of $K = 48$ projections and a neural network classifier with five hidden units were sufficient for the task. These parameters have been estimated, using cross-validation re-sampling of the training data, see e.g. [26] (data not shown). The complexity of the combined system is optimized by adaptive regularization ('weight decay') for each corpus separately by the neural network training procedure which is based on Bayesian ML-II methods [24].

In previous studies on the three corpora it has been shown that the email and multimedia data set are relatively well classified with term features alone, while the WebKB data set is relatively hard to classify. In figure 1 we show a 2D projection of the email set indicating that the classes are indeed well separated.

We performed three types of experiments. Using the POS-tags alone, using the terms alone and using the combined feature set. We split the corpora in 20% for training and 80% for testing (the role of the split ratio is discussed below). The POS-tags features alone (i.e., using only the relative frequencies of word category) are surprisingly potent: We found that 89.7% of the multimedia data-set is classified correctly using 90 POS-tag features. This should be compared to 96.6% classification accuracy obtained with the almost 3500 term features. For the email data, using the POS-tag and term features separately resulted in accuracies of 74.6% and 94.2% respectively. The WebKB data is somewhat harder to classify. Here the POS-tag and term features lead to accuracies of 57.2% and 76.1% respectively.

The potential synergy of terms and POS-tags is illustrated in figure 2. The figure shows the performance correlation between the classifiers trained on the individual feature sets. The bars labelled 'independent' indicate the rates of events where the two classifiers are both correct as well as events where one is correct and one is incorrect obtained from their basic performance and assuming independence of their decisions. In bars labelled 'real' we show the actually observed rates. Note that there is a high potential synergy, since the observed performances are close to those predicted by independence. We

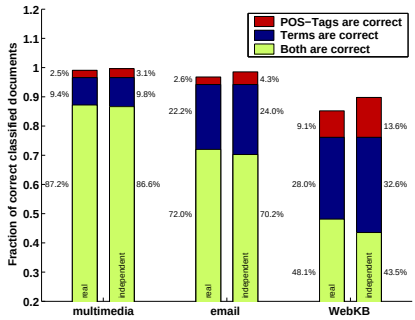


Figure 2: Fraction of correct classified documents for the POS-tag and term representations. The bars labelled 'real' indicate observed rates of events where the two feature sets lead to correct decision and one correct/one incorrect respectively. This is compared with rates estimated from assumed independence of errors (bars labelled 'independent'). The figure indicates that the errors made by classifiers based on POS-tags and the term features sets are relatively independent, hence, that there is a potential synergy to be gained from binding the feature sets.

next turned to the combined feature set. In figure 3 we illustrate the role of the binding coefficient α, c.f., (4). The classification test set error rates (an unbiased estimate of the generalization error defined as the probability of misclassification of a random test datum) were obtained by ten-fold cross-validation. We observed significant synergy: The performance of the term features ($\alpha = 1$) is indeed improved by adding POS-tag feature information. The effect is relatively high for the multimedia data-set (reducing the error by almost 30%), while the effect is smaller for the harder WebKB set (the error is reduced by about 8%). The synergistic advantage is likely to depend on the size of the database, to further investigate this we estimated 'learning curves' for the the combined system by changing the split ratio allowing for variable training set sizes. The results are provided in figure 4. In these ten-fold cross-validation experiments we used the 'optimal' binding coefficients found in figure 3. In these relatively limited data sets there is a positive, albeit diminishing, synergy to be obtained for all training set sizes.

CONCLUSION

Natural language features in the form of part-of-speech (POS) tags were introduced to supplement bag-of-words features. We propose simple statistical POS-tag features: The frequency of different term types. By early binding of POS-tags and term features we find a synergistic effect for a range of binding coefficients and for all training sets sizes studied. The results were consistent for three different corpora posing variable classification difficulties. As the POS-tag features are relatively automatic and computationally 'inex-

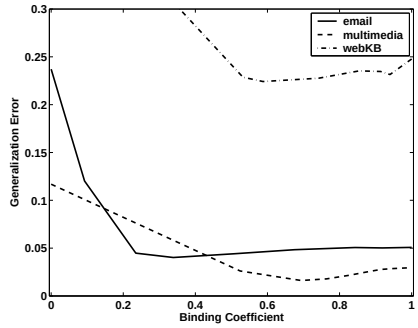


Figure 3: Misclassification error obtained by binding POS-tags and Terms with variable a binding coefficient. $\alpha = \mathbf{1}$ corresponds to tag features only. Optimal binding results in reduction of the error rate by 30%, 22% and 8% in the email, multimedia and WebKB corpora respectively Results obtained by ten-fold cross-validation using a 20/80 train/test set split ratio.

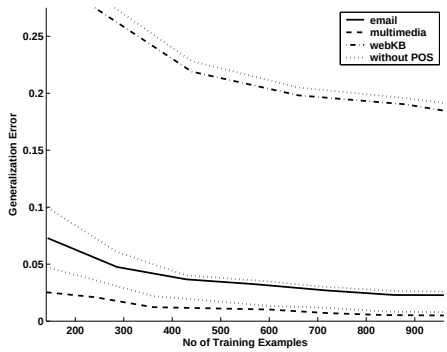


Figure 4: Learning curves with and without binding. The binding of natural language and conventional term features improves performance for all the training set sizes investigated.

pensive' to estimate we recommend that these feature be included in future text/contex classification applications.

REFERENCES

[1] A. Aizawa, "Linguistic Techniques to Improve the Performance of Automatic Text Categorization," in **Proceedings of NLPRS-01, 6th Natural Language Processing Pacific Rim Symposium**, Tokyo, JP, 2001, pp. 307–314.

[2] R. Basili and A. Moschitti, "A robust model for intelligent text classification,"

in **Proceedings of ICTAI-01, 13th IEEE International Conference on Tools with Artificial Intelligence**, Dallas, US: IEEE Computer Society Press, Los Alamitos, US, 2001, pp. 265–272.

[3] R. Basili, A. Moschitti and M. Pazienza, "NLP-driven IR: Evaluating Performances over a Text Classification task," in B. Nebel (ed.), **Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence**, Seattle, US, 2001, pp. 1286–1291.

[4] T. Berners-Lee, J. Hendler and O. Lassila, "The Semantic Web," **Scientific American**, 2001.

[5] I. A. R. Center, "The WebFountain," http://www.almaden.ibm.com /webfountain/publications/.

[6] S. Chakrabarti, "Data mining for hypertext: a tutorial survey," **SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM**, vol. 1, pp. 1–11, 2000.

[7] CMU-WebKB, "The 4 Universities Data Set," http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/, 1997.

[8] CMU-WebKB-2240, "A subset of the WebKB," http://www.imm.dtu.dk/∼rem/, 1999.

[9] M. De Buenaga Rodríguez, J. M. Gómez-Hidalgo and B. Díaz-Agudo, "Using WordNet to Complement Training Information in Text Categorization," in R. Milkov, N. Nicolov and N. Nikolov (eds.), **Proceedings of RANLP-97, 2nd International Conference on Recent Advances in Natural Language Processing**, Tzigov Chark, BL, 1997.

[10] F. Debole and F. Sebastiani, "Supervised term weighting for automated text categorization," in **Proceedings of SAC-03, 18th ACM Symposium on Applied Computing**, Melbourne, US: ACM Press, New York, US, 2003, pp. 784–788.

[11] S. Deerwester, S. Dumais, T. Landauer, G. Furnas and R. Harshman, "Indexing by Latent Semantic Analysis," **Journal of the American Society of Information Science**, vol. 41, pp. 391–407, 1990.

[12] G. Furnas, S. Deerwester, S. Dumais, T. Landauer, R. Harshman, L. Streeter and K. Lochbaum, "Information retrieval using a singular value decomposition model of latent semantic structure," in **The 11th International Conference on Research and Development in Information Retrieval**, Grenoble, France: ACM Press, 1988, pp. 465–480.

[13] R. Guha and R. McCool, "TAP: A Semantic Web Platform," **Computer Networks: The International Journal of Computer and Telecommunications Networking**, vol. 42, pp. 557–577, 2003.

[14] L. Hansen, S. Sigurdsson, T. Kolenda, F. Nielsen, U. Kjems and J. Larsen, "Modeling text with generalizable Gaussian mixtures," in **International Conference on Acoustics, Speech and Signal Processing**, IEEE, 2000, pp. 3494–3497.

[15] A. Kehagias, V. Petridis, V. G. Kaburlasos and P. Fragkou, "A Comparison of Word- and Sense-based Text Categorization Using Several Classification Algorithms," **Journal of Intelligent Information Systems**, vol. 21, no. 3, pp. 227–247, 2003.

[16] T. Kolenda, "Multimedia Dataset," http://mole.imm.dtu.dk/faq/MMdata/, 2002.

[17] T. Kolenda, L. Hansen, J. Larsen and O. Winther, "Independent component analysis for understanding multimedia content," in S. B. J. L. H. Bourlard, T. Adali and S. Douglas (eds.), **Proceedings of IEEE Workshop on Neural Networks for Signal Processing XII**, Piscataway, New Jersey: IEEE Press, 2002, pp. 757–766.

[18] R. Kosala and H. Blockeel, "Web Mining Research: A Survey," in **SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM**, ACM Press, 2000, pp. 1–15.

[19] J. Larsen, L. Hansen, A. Have, T. Christiansen and T. Kolenda, "Webmining: learning from the world wide web," **Computational Statistics and Data Analysis**, vol. 38, pp. 517–532, 2002.

[20] O. Mason, "Probabilistic Part-of-Speech Tagger," http://web.bham.ac.uk/o.mason/software/tagger/, 2003.

[21] F. Nielsen, "Email Data-Set," http://www.imm.dtu.dk/∼rem/, 2001.

[22] H. Schmid, "Probabilistic Part-of-Speech Tagging Using Decision Trees." in **International Conference on New Methods in Language Processing**, Manchester, UK, 1994.

[23] F. Sebastiani, "Machine learning in automated text categorization," **ACM Computing Surveys**, vol. 34, pp. 1–47, 2002.

[24] S. Sigurdsson, "The DTU: Artificial Neural Network Toolbox," http://mole.imm.dtu.dk/toolbox/ann/, 2002.

[25] S. Sigurdsson, J. Larsen and L. Hansen, "On Comparison of Adaptive Regularization Methods," in B. Widrow, L. Guan, K. Paliwa, T. Adali, J. Larsen, E. Wilson and S. Douglas (eds.), **Proceedings of the IEEE Workshop on Neural Networks for Signal Processing**, 2000, pp. 221–230.

[26] S. Strother, J. Anderson, L. Hansen, U. Kjems, R. Kustra, J. Siditis, S. Frutiger, S. Muley, S. LaConte and D. Rottenberg, "The quantitative evaluation of functional neuroimaging experiments: The NPAIRS data analysis framework," **Neuroimage**, vol. 15, pp. 747–771, 2002.

Appendix E

Enhanced Context Recognition by Sensitivity Pruned Vocabularies

Enhanced Context Recognition by Sensitivity Pruned Vocabularies

Rasmus Elsborg Madsen, Sigurdur Sigurdsson and Lars Kai Hansen
Informatics and Mathematical Modelling
Technical University of Denmark
Building 321, DK-2800 Kgs. Lyngby, Denmark
rem,siggi,lkh@imm.dtu.dk

Abstract

Document categorization tasks using the "bag-of-words" representation have been successful in instances [11]. The relatively low dimensional bag-of-words form, is well suited for machine learning methods. The pattern recognition methods suffers though, from the well-known curse of dimensionality, since the number of input dimensions (words) usually supersedes the number of examples (documents). This high dimensional representation is also containing many inconsistent words, possessing little or no generalizable discriminative power, and should therefore be regarded as noise. Using all the words in the vocabulary is therefore resulting in reduced generalization performance of classifiers. We here study the effect of sensitivity based pruning of the bag-of-words representation. We consider neural network based sensitivity maps for determination of term relevancy, when pruning the vocabularies. With reduced vocabularies documents are classified using a latent semantic indexing representation and a probabilistic neural network classifier. Pruning the vocabularies to approximately 3% of the original size, we find consistent context recognition enhancement for two mid size data-sets for a range of training set sizes. We also study the applicability of the sensitivity measure for automated keyword generation.

1 Introduction

The world wide web is an unstructured and fast growing database. Today's search tools often leave web users in frustration by the low precision and recall[1]. It is widely believed that machine learning techniques can come to play an important role in web search. Here we consider web content mining in the form of document classification - an information retrieval aspect of web-mining [8]. Our aim is to improve generalizability of supervised document classification by vocabulary pruning.

In the bag-of-words representation we summarize documents by their term histograms. The main motivation for this reduction is that it is easily automated and needs minimal user intervention beyond filtering of the term list. The term list typically contains in the range of $10^3 - 10^5$ terms, hence further reduction is necessary for most pattern recognition devices. Latent semantic indexing (LSI) [4, 3] aka principal component analysis is often used to construct low dimensional representations. LSI is furthermore believed to reduce synonymy and polysemy problems [3, 9]. Although LSI and other more elaborate vector space models have been successful in text classification in small and medium size databases, see e.g., [7, 5], we are still not at human level text classification performance. When training classifiers on relatively small databases generalizability is a key issue. How well does a model adapted on one set of data predict the labels of another test data set? Generalizability is in general a function of the number of training cases and of the effective model dimension. We are interested in investigating whether automated vocabulary reduction methods can contribute to classification accuracy by reducing the model complexity. To estimate term relevance we will use the notion of the scaled sensitivity, which is computed using a the so-called NPAIRS split-half re-sampling procedure [15]. Our hypothesis is that 'sensitivity maps' can determine which terms are consistently important. That is terms which are likely to be of general use for classification, relative to terms that are of low or highly variable sensitivity. As a side we also illustrate the feasibility of using the sensitivity to generate class specific keywords.

2 Methods

Documents are arranged in a term-document matrix $\mathbf{X}$, where X_{td} is the number of times term t occur in document d. The dimensionality of $\mathbf{X}$ is reduced by filtering and stemming. Stemming refers to a process in which words with different endings are merged, e.g., 'trained' and 'training' are merged into the common stem 'train'. This example also

indicates the main problem with stemming, namely that introducing an artificial increased polysemy. We have decided to 'live with this problem' since without stemming vocabularies would grow prohibitively large. About 500 common non-discriminative stop-words, i.e. (a, i, and, an, as, at) are removed from the term list. In addition high and low frequency words are also removed from the term list. The resulting elements in $\mathbf{X}$ are term frequency/inverse document frequency normalized,

$$X_{t,d}^{\text{tfidf}} = \frac{X_{t,d}}{\sqrt{\sum_{t'=1}^{T} X_{t',d}^2}} \log \frac{D}{F_t}. \quad (1)$$

Where $\mathbf{X}^{\text{tfidf}}$ is the normalized term document matrix, T is the number of terms, D is the number of documents and F_t is the document frequency of term t. The normalized term document matrix is reduced to a feature-document matrix using PCA, carried out by an 'economy size' singular value decomposition,

$$\mathbf{X} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T. \quad (2)$$

Where the orthonormal $T \times D$ matrix $\mathbf{U}$ contains the eigenvectors corresponding to the non-zero eigenvalues of the symmetric matrix $\mathbf{X}\mathbf{X}^T$. $\mathbf{\Lambda}$ is a $D \times D$ diagonal matrix of singular values ranked in decreasing order and the $D \times D$ matrix $\mathbf{V}^T$ contains eigenvectors of the symmetric matrix $\mathbf{X}^T\mathbf{X}$. The LSI representation is obtained by projecting document histograms on the basis vectors in $\mathbf{U}$,

$$\mathbf{F} = \mathbf{U}^T\mathbf{X} = \mathbf{\Lambda}\mathbf{V}^T. \quad (3)$$

Typically, the majority of the singular values are small and can regarded as noise. Consequently, only a subset of K ($K << T$) features are retained as input to the classifier algorithm. The representational potential of these LSI features is illustrated in figure 1. A wide variety of classification algorithms have been applied to the text categorization problem, see e.g., [8]. We have extensive experience with probabilistic neural network classifiers and a well tested ANN toolbox is available [12]. The toolbox adapts the network weights and tunes complexity by adaptive regularization and outlier detection using the Bayesian ML-II framework, hence, requires minimal user intervention [12, 13].

We use the definition of class specific sensitivity proposed in [16, 14] for a set of N samples,

$$\mathbf{s}_k = \frac{1}{N}\sum_{n=1}^{N}\left|\frac{\partial P(c_k|\mathbf{f}_n)}{\partial \mathbf{x}}\right| \quad (4)$$

and where $P(c_k|\mathbf{f}_n)$ is the posterior probability of class k given the feature vector $\mathbf{f}_n$. $\mathbf{s}_k$ is the T-dimensional sensitivity vector for class k. The T-dimensional derivative is obtained using the projection (3) [14]. A split-half resampling procedure is invoked to determine the statistical significance of the sensitivity [15]. Multiple splits are generated of the original training set and classifiers trained on each of the splits. For each classifier a sensitivity map is computed. Since the two maps obtained from a given split are exchangeable the mean map is an unbiased estimate of the 'true' sensitivity map, while the squared difference is a noisy, but unbiased estimate of the variance of the sensitivity map. By repeated re-sampling and averaging the sensitivity map and its variance is estimated. We finally obtain a scaled sensitivity map by normalization trough the standard deviation.

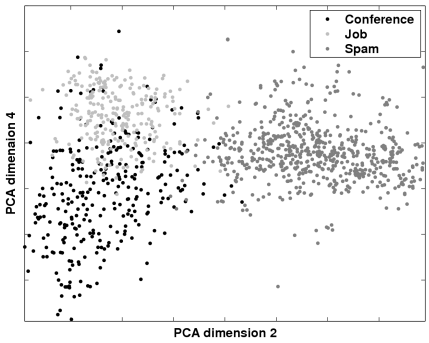


Figure 1. Illustration of the document distribution in feature space. Here we show the Email corpus projected onto the 2nd and 4th principal directions. In this projection the 'spam' class is well separated while the two other classes in the set ('conferences' and 'jobs') show some overlap.

3 Data

Three data-sets, 'Email' [10], 'Multimedia' [6] [7] and 'WebKB' [2] are used to illustrate and test the hypothesis. No less than ten split-half re-samples are used in all experiments. The Email data-set consists of texts from 1431 emails in three categories: conference (370), job (272) and spam (789). The multimedia data set consists of texts and images from 1200 web pages. Only the text part is considered here. The categories are: sports (400), aviation (400) and paintball (400). The WebKB set contains 8282 webpages from US university computer science departments. Here we have used a subset of 2240 pages from the WebKB earlier used in [5] and [9]. The WebKB categories are: project (353), faculty (483), course (553) and student (851). All html tags were removed from the data-set.

4 Results

Preliminary experiments indicated that a reduced feature space of $K = 48$ projections and a neural network classifier with five hidden units were sufficient for the task (data not shown). All results have been validated using 10 fold split half re-sampling cross validation. The neural network based term sensitivity is a function of the given training set. Terms for which the sensitivity is high but also highly variable are less likely to support generalizability compared to terms that have a consistent high or medium sensitivity. The empirical distribution of mean and standard deviations the terms in the Email set are shown in figure 2. The empirical

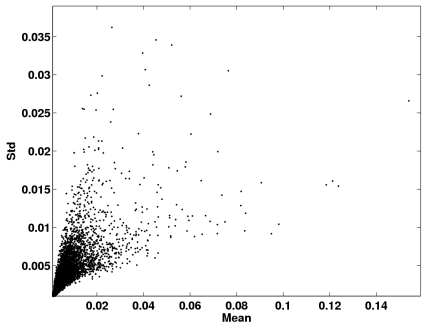


Figure 2. Mean and standard deviation of the term sensitivity. The most relevant terms have consistent high sensitivity in each re-sampling split, i.e., a high mean and relatively low standard deviation. These terms occupy the lower right part of the plot.

scaled sensitivities $Z_t = \mu_t/\sigma_t$ of the terms in the Email data, are shown in figure 3. The scaled sensitivities can be used also to select relevant keywords for the text categories. For the Email data the five highest scores for the Conference category are (*Paper, Conference, Deadline, Neural, Topic*) and for the Job category (*Research, Position, Candidate, University, Edt*) and for the Spam category (*Money, Remove, Free, Thousand, Simply*). We then remove increasing fractions of the terms using the scaled sensitivity ranking. Using all the terms, the generalization classification error rate is 16.9% in the WebKB and 2.7% in Email data. Removing 97% of the terms with the lowest scaled sensitivity, the generalization error for the WebKB is reduced to 14.3% and to 2.3% for the Email data. This is a reduction of about 15% for both data-sets. For the Multimedia data the pruning does not lower the generalization error, but about 80% of the multimedia vocabulary can be removed without loss of generalizability. The generalizability as function of pruning fraction is presented in figure 4. The multimedia

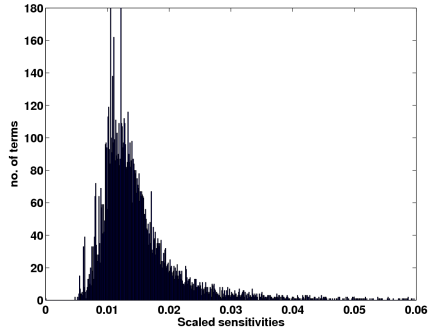


Figure 3. Scaled sensitivities Email data set terms. Many of the terms have a standard deviation on the order of the mean value indicating that they are not consistently sensitive. 3% of the terms has a scaled sensitivity higher than 3.75.

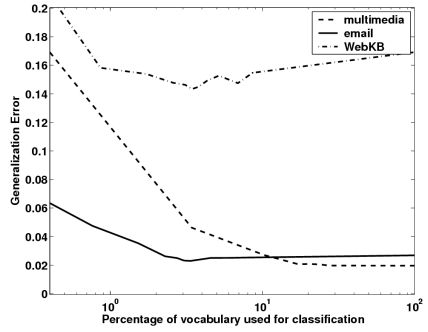


Figure 4. Generalization error using subsets of the vocabulary for the WebKB, Email and Multimedia data. The terms retained are those with the highest scaled sensitivity. For the WebKB and Email data-sets pruning to 3% of the vocabulary gives the lowest generalization error with error rates reduced about 15%. The relatively limited vocabulary of the Multimedia data set can be reduced to about 50% without decreasing performance but does not improve performance.

data has the sparsest vocabulary with only 3500 terms after filtering while the Email data set has 9500 terms, and the WebKB data has 12950 terms after filtering. In figure 5 we show that learning curves are consistently improved for a range of training sets for the WebKB and the Email data

based on a fixed reduction to 3% of their original vocabulary. Pruning the vocabulary to 3% of the original size, re-

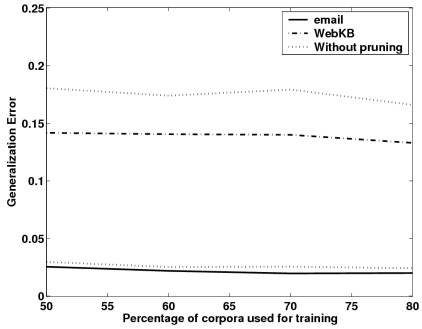


Figure 5. Learning curves for full and reduced the vocabularies for the WebKB and Email data sets. Both vocabularies are reduced to 3% of their original sizes. While the effect of pruning is consistently positive for all training set sizes, the effect is most pronounced for small samples.

sults in better generalization in the whole range of training-set sizes and has the largest effect for small training sets.

5 Conclusion

Neural network sensitivity maps has been applied to a latent semantic indexing context recognition framework. The use of scaled sensitivities for reducing vocabularies, has resulted in use of only 3% of the original vocabulary. The vocabulary reduction has lead to a simpler modeling while also significantly improving classification performance for two large vocabulary data sets. This indicates that these context classification problems are generalization limited by model complexity for the sample sizes studied. Classification of a third data-set with a more limited vocabulary was not enhanced by vocabulary pruning. However, the vocabulary could be pruned to 80% of the initial size without loss of performance. We thus recommend to monitor the test set classification performance as function of the pruning fraction. The scaled sensitivity has also been useful for identifying class specific keywords.

References

[1] S. Chakrabarti. Data mining for hypertext: a tutorial survey. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, 1:1–11, 2000.

[2] CMU-WebKB. The 4 universities data set. http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/, 1997.

[3] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41:391–407, 1990.

[4] G. Furnas, S. Deerwester, S. Dumais, T. Landauer, R. Harshman, L. Streeter, and K. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. In *The 11th International Conference on Research and Development in Information Retrieval*, pages 465–480, Grenoble, France, 1988. ACM Press.

[5] L. Hansen, S. Sigurdsson, T. Kolenda, F. Nielsen, U. Kjems, and J. Larsen. Modeling text with generalizable gaussian mixtures. In *International Conference on Acoustics, Speech and Signal Processing*, pages 3494–3497. IEEE, 2000.

[6] T. Kolenda. Multimedia dataset. http://mole.imm.dtu.dk/faq/MMdata/, 2002.

[7] T. Kolenda, L. Hansen, J. Larsen, and O. Winther. Independent component analysis for understanding multimedia content. In S. B. J. L. H. Bourlard, T. Adali and S. Douglas, editors, *Proceedings of IEEE Workshop on Neural Networks for Signal Processing XII*, pages 757–766, Piscataway, New Jersey, 2002. IEEE Press.

[8] R. Kosala and H. Blockeel. Web mining research: A survey. In *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, pages 1–15. ACM Press, 2000.

[9] J. Larsen, L. Hansen, A. Have, T. Christiansen, and T. Kolenda. Webmining: learning from the world wide web. *Computational Statistics and Data Analysis*, 38:517–532, 2002.

[10] F. Nielsen. Email data-set. http://www.imm.dtu.dk/∼rem/, 2001.

[11] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47, 2002.

[12] S. Sigurdsson. The dtu: Artificial neural network toolbox. http://mole.imm.dtu.dk/toolbox/ann/, 2002.

[13] S. Sigurdsson, J. Larsen, L. Hansen, P. A. Philipsen, and H. C. Wulf. Outlier estimation and detection: Application to skin lesion classification. In *International conference on acoustics, speech and signal processing*, pages 1049–1052, 2002.

[14] S. Sigurdsson, P. Philipsen, L. Hansen, J. Larsen, M. Gniadecka, and H. Wulf. Detection of skin cancer by classification of Raman spectra. *Accepted for IEEE Transactions on Biomedical Engineering*, 2004.

[15] S. Strother, J. Anderson, L. Hansen, U. Kjems, R. Kustra, J. Siditis, S. Frutiger, S. Muley, S. LaConte, and D. Rottenberg. The quantitative evaluation of functional neuroimaging experiments: The NPAIRS data analysis framework. *Neuroimage*, 15:747–771, 2002.

[16] J. Zurada, A. Malinowski, and C. I. Sensitivity analysis for minimization of input data dimension for feedforward neural network. In *Proceedings of the IEEE Symposium on Circuits and Systems*, pages 447–450, 1994.

Appendix F

Modeling Text using State Space Models

Modeling Text using State Space Models

Rasmus E. Madsen
Department of Mathematical Modelling
Technical University of Denmark
Lyngby, DK-2800
rem@imm.dtu.dk

Abstract

Generic "bag-of-words" text categorization methods are only based on the information contained in word count histograms. These methods does therefore not capture the information contained in the order in which the words appear in a document. We here consider models that is acting on both parts of information at the same time, that is the information about what words appear and in what order they appear. State-space models has the ability to capture information from the order in which the words appear, and combine it with the word appearance probabilities. The state-space models should therefore conceptually super-seed the bag-of-words/vector-space models, in ability to model documents correctly. In the following we experiment with two state space model approaches, for making categorization better.

1 Introduction

The document vector space model (Salton et al., 1975), the bag-of-words model and its varieties are effective document simplifications, that make machine learning approaches to text modeling and classification simple. The two document representations has resulted in the development of many different algorithms (Deerwester et al., 1990; Hofmann, 1999; Sebastiani, 2002; Blei et al., 2003) who are effective for text classification. The models that use these representations however loose a big fraction of the information contained in the documents, by considering only the counts of how many times words appear in a given document. The other part of information contained in documents is the information about the order in which the words appears. Though the major part of document information is contained in the knowledge about which word occur, some important information might be captured from the word appearance order, that could make document classification accuracy better. One way to interpret the word order information is as being the authors style of writing, i.e. a fingerprint that tells how the author constructs his sentences. Some authors might construct grammatically different sentences from others. This grammatical difference might not be captured when only word histograms are considered.

It is easy to extract the word appearance information from a document and form it into some meaningful representation that can be used for machine learning, i.e. vectors or histograms. The word order information is however harder to extract to some simple low dimensional representation, which is easily portable to a machine learning algorithms. We therefore consider state space models, which can model sequences of data, instead of the counts.

State space models have previously been used for language modeling, e.g. in context of predicting the next word in handwritten text recognition systems (Zimmermann & Bunke, 2004), and has been successful so. It is therefore further likely that the state-space model can capture valuable information that can be used for text classification.

We here consider two different state-space based approaches, both based on an underlying Markov state space model. Both approaches suggest a method to overcome the dimensionality problem of text, which otherwise makes the state-space models extremely slow. The first approach suggested here generates a new lower dimensional vocabulary, which is later used in a hidden Markov model. Using the second approach, the state part of a hidden Markov model is used in conjunction with LSI emission probabilities.

2 Discrete Markov Process

The discrete Markov process (Rabiner & Juang, 1986) is a state space model that can model and generate sequences of discrete symbols. The discrete Markov process considers a system with K states s_k, where for each time-step t the process changes state, where the new state can be the same as the previous state. The actual state at time t is denoted q_t, which can be interpreted as the discrete symbol generated at time t. The probability of changing state to a new state $q_{t+1} = s_j$ from the state $q_t = s_i$ is determined by the transition probabilities $a_{s_i,s_j} = P(q_{t+1} = s_j | q_t = s_i)$, where $\sum_{j=1}^{K} a_{s_i,s_j} = 1$ and $a_{s_i,s_j} \geq 0$. The transition probabilities are therefore only dependent on the current state of the process and not the time t or previous states $q_{t-t'}$. A tutorial on Markov processes can be found in (Rabiner, 1989).

The discrete Markov process assembles an urn scheme where there is one urn for each state in the Markov process. When the time-step changes, a new urn is selected according to the transition probabilities, and a ball from that urn is drawn, and the color noted, whereafter the ball is returned into the urn. Each urn contains only balls with the same color.

The urn model analogy to text modeling is straight forward. Instead of balls, each urn is filled with words, again only one kind of words for each urn. When a document is generated, we start out with one particular urn and draw a word from it, and continue to another urn and draw a new word here. The transition probabilities determines what words are likely to appear after the present one. The Markov process will therefore be able to model parts of the semantics of the language model, by the transition probabilities. These semantics are not modeled at all when only word appearances alone are considered, i.e. using the vector space model representation.

Different kinds of documents might contain the same kinds of words, where the order of the appearances of the words, can change the meaning of the content. The word “train” could for example be used in documents about transportation or in documents about exercising in the gym. The words appearing around the word train, will therefore change the meaning of that particular word. The difference in meaning could therefore be captured by the Markov model. Another example of when the order of the words appearances can change the meaning of a sentence, is when the word “not” is used. Yet another example where transition probabilities could be useful is in spam email detection systems.

The drawback of the Markov model is that it models a huge probability space, since it considers all the possible word-pairs in the vocabulary. Since most document collection vocabularies considers about $100,000$ words, the model must consider $10,000,000,000$ possible transition probabilities. The transition probabilities would therefore consume to much memory for holding this data representation. By use of a grammar, many of the transition probabilities could be pruned away, while many word pairs can’t be used in grammatically correct sentences. Though the pruning approach would reduce the amount

of modeled probabilities tremendously, the amount of memory used to represent the model would still be very large. On top of the memory consumption, the model would also need a lot of data to be able to estimate all the transition probabilities. For existing document collections, the amount of data is far too limited to estimate the probabilities, making a huge need for smoothing, which usually result in bad modeling performance. Human brains can probably work with some variety of this modeling approach, while we can generalize many probabilities in the model by use of grammar and can therefore easily prune away the unlikely Markov model transition probabilities.

3 Hidden Markov Model

The hidden Markov model (HMM) (Rabiner & Juang, 1986; Rabiner, 1989) extends the discrete Markov process by adding an additional emission parameter to each state. The emission parameters controls the output that is generated from each state, i.e. a discrete symbol. For the HMM, each state therefore has the potential to generate all the symbols in the vocabulary of symbols. For each time-step t the HMM still changes state according to the transition probabilities a_{s_i,s_j}, but the the symbol is now generated using the emission probabilities $b_{s_j,v_m} = P(x_t = v_m|q_t = s_j)$, where x_t is the symbol generated at time t and v_m is symbol number m from the vocabulary of M symbols.

The HMM assembles an urn scheme that is similar to the Markov process urn scheme. A new urn is still selected at each time-step according to the transition probabilities, and a ball from the new urn is drawn. The color of the ball is noted whereafter the ball is returned into the urn. Using the HMM each urn now contains a distribution of balls that each has one of M different colors.

Since the number of symbols that can be generated M is independent of the number of states K, the memory consumption of the model can be reduced remarkably when the vocabulary is huge. If we consider a vocabulary of about $100,000$ words and use a state-space of 100 states, the amount of probabilities used to describe the model is approximately $10,000,000$, which is only $1/1000$ of the amount of memory needed to describe the Markov process for the same vocabulary.

The HMM parameters can be estimated using the expectation maximization (EM) algorithm (Dempster et al., 1977), resulting in an iterative update procedure that estimates the model parameters using the so called forward-backward approach (Rabiner, 1989),

$$\pi_{s_i} = \gamma_{1,s_i} \tag{1}$$

$$a_{s_i,s_j} = \frac{\sum_{t=1}^{T-1} \xi_{t,s_i,s_j}}{\sum_{t=1}^{T-1} \gamma_{t,s_i}} \tag{2}$$

$$b_{s_j,v_m} = \frac{\sum_{t=1}^{T} (O_t = v_m)\xi_{t,s_i,s_j}}{\sum_{t=1}^{T} \gamma_{t,s_j}} \tag{3}$$

where π_{s_i} is the probability of starting in state s_i and $\gamma_{t,s_i} = \sum_{j=1}^{K} \xi_{t,s_i,s_j}$ and $\xi_{t,i,j}$ is the probability of being in state s_i at time t and in state s_j at time $t+1$ and $(O_t = v_m)$ is 1 if the observation at time t is symbol v_m, and zero otherwise. The full description of the learning rules can be found in (Rabiner, 1989).

4 HMM with LSI GMM Vocabulary

The HMM approach reduces the memory needs, comparing it with a Markov process with a similar vocabulary size, making it possible to represent the model in a standard computer of today. The HMM model is however still fairly large and the EM updates that estimates the parameters are very demanding, computationally. In the approach described here, the vocabulary is therefore projected to a lower dimensional representation using latent semantic indexing (LSI) (Deerwester et al., 1990) with a SVD basis (Madsen et al., 2003) and gaussian mixture models (GMM). In Figure 1, the the lower dimensional representation of the vocabulary is shown.

The procedure of transforming the vocabulary to a lower dimensional representation, takes place in the following way:

1. Documents are cut into substrings of length L, with 50% overlap.
2. A common LSI representation for the substrings in all the documents is estimated using SVD.
3. The substrings are clustered using GMM on the first H dimensions of the LSI representation.
4. The clusters are now forming a new and much smaller vocabulary for the substrings, where each substring is transformed to an the index associated with the closest cluster.
5. A HMM is trained for each class of documents using the new vocabulary.
6. New documents are classified using the HMM forward backward classification algorithm.

The classification algorithm is using the forward-backward approach which is also used to estimate the parameters.

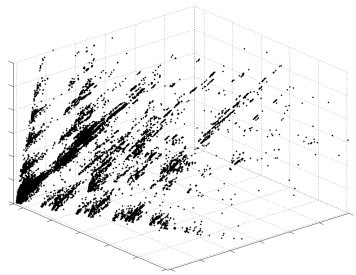


Figure 1: Space for the new vocabulary.

5 HMM with LSI emission probabilities

In the section about the hidden Markov model, we reject the model for use on text directly, while the high number of parameters for the model would make it converge slowly, due to size of the vocabulary. It is further undesirable to use the HMM directly on each single class while the classes wont be able to share the emission probabilities. It is desirable to share

the emission probabilities for all the classes while they can be thought of as latent topics, where there is a latent topic for each single state in the HMM. This idea is conceptually similar to the ideas from latent semantic indexing and it's varieties (Furnas et al., 1988; Deerwester et al., 1990; Hofmann, 1999; Kolenda et al., 2002; Blei et al., 2002; Blei et al., 2003).

The problems of shared latent topic emissions could be overcome by redefining the HMM to be a model with more state space transition models, but only one single state emission model. This model would be likely to inherit the slow convergence property of the normal HMM. We therefore reject the model here, knowing that it probably would be the best modeling approach to the problem.

The alternative to a redefined HMM, is to estimate the emission probabilities b_{s_j,v_m} using another algorithm and keeping them fixed when first estimated. Using this approach it would only be necessary to estimate the state transition parameters a_{s_i,s_j} and initial state probabilities π_{s_i} for each separate class. This estimation procedure would further not need to run in an iterative EM-loop where the one set of parameters are estimated based on an estimate of the other set of parameters. The transition parameters would therefore only need one or very few iterations to converge.

There are more alternative ways to determine a set of shared latent topic emission parameters. Three possible approaches are *independent component analysis* (ICA) (Bell & Sejnowski, 1995b; Bell & Sejnowski, 1995a; Molgedey & Schuster, 1994), *singular value decomposition* (SVD) (Madsen et al., 2003) and *non-negative matrix factorization* (NMF) (Lee & Seung, 1999; Lee & Seung, 2001). The latter approach has the advantage of estimating non-negative values when factorizing the data, which is valuable since these values reflect emission probabilities, i.e. they have to be positive and sum to zero. NMF has also shown valuable for text clustering previously (Xu et al., 2003). In practise however the NMF does not work well with the sparse structure of the text data, resulting in very few active words in each NMF latent topic. When only few words are active it is necessary to either use a lot of smoothing or use many latent topics. Neither of these fixes are likely to give us a good model or classifier, so we turn to SVD approach instead. The latent topics estimated by the SVD all have many active words. The problem of probabilities being negative is solved by simply setting negative values equal to zero, and then normalize the distribution.

The procedure of using the HMM state space model with LSI estimated emission probabilities, takes place in the following way:

1. A common set of HMM emission parameters are estimated using the LSI approach on the documents using the histogram representation.
2. A set of HMM state space parameters are estimated for each class using the word sequences for each document.
3. New documents (sequences of words) are classified using the HMM forward backward classification algorithm.

6 Experiments

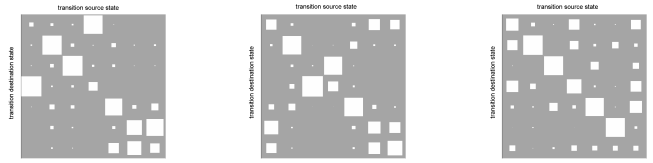
We are here working with the three corpora: email, WebKB and multimedia. The number of words in the three corpora are reduced by use of stemming and stop-word removal. Though we here only show results for the email-data, similar results where gained by use of the two other data-sets. The TF-IDF transformation has been applied to the document collections, when performing experiments using the HMM with LSI-GMM generated vocabulary. In the experiments where using the HMM with LSI emission probabilities, the TF-IDF transformation has not been applied. The reason is that the HMM works on sequences where

each unit in the sequence must be unity. A weighting scheme could be applied to the HMM, where the TF-IDF coefficients could be applied as weights. At first we are interested in investigating if the model works conceptually, and have therefore skipped the transformation step.

We start by training the HMM with LSI-GMM generated vocabulary (HLG) using the email-data. The largest class in the email data-set (spam) accounts for 0.55% of the emails. A naive classifier should therefore have a classification accuracy of about 0.55. The two models considered should therefore have generalization error below 0.45%.

We find that the HLG approach works best when a LSI subspace of 4 dimensions is used to form the new HLG generated vocabulary. A set of 100 gaussian's is used to cover the 4-dimensional space forming an new vocabulary of 100 words. The first three dimensions of the subspace are shown in Figure 1, where the structure of the data are much different from the structure found by the generic LSI representation Figure **??**. Each cluster that is put in the space in Figure 1 now represents a word in the new vocabulary.

Estimating the HMM for the new sequences, the transition probabilities for the three classes in the email-set, show us whether there is a sequential difference between the three classes that is captured by the model. In Figure 2, a graphical illustration of the transition probabilities is shown. Seven states is used in the HMM to best model the new sequences.



(a) Job (b) Conference (c) Spam

Figure 2: Graphical illustration of the transition probabilities for the HLG model. For the Job category (a), state 1 and 4 are paired and almost isolated from the other states making them a semantic chain for the category. Similarly state 5, 6 and 7 forms a state group that are likely to generate long sequences of words. The Conference category (b) has a similar group formation where the states 3 and 4 forms a group, and state 1,7 and 8 forms a group. The spam category (c) does not have the same strong group formation as the two other categories, but is instead less symmetric. There is however weak group formation between the states 1, 2 and four, and the states 3 and 5. The illustration of the transition probabilities reveal that there is a sequential pattern that is captured by the model.

The illustration in Figure 2 shows clearly that there is information in the order in which the words appear in a document, and that this information can be captured by the HMM.

There are more settings that determines the optimal HLG model, i.e. number of LSI dimensions, number of states in the HMM, the number of gaussian mixtures and the length of the substrings used to form the vocabulary. In Figure 3 the classification accuracy for the HLG model as function of the substring length is plotted, where the settings for the remaining parameters are close to optimal.

The HLG model has an accuracy that is lower than the accuracy of the LSI model, for all possible substring-length values.

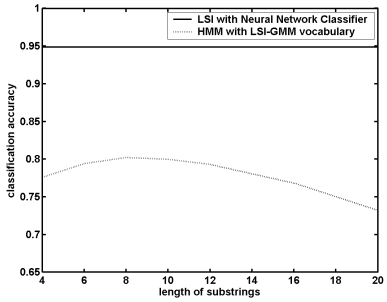
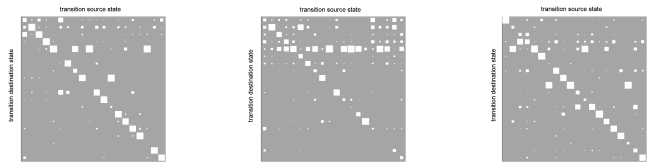


Figure 3: Classification accuracy using the HMM with LSI-GMM reduced vocabulary, as function of the substring length. The classification results are compared with a neural network classifier using a LSI subspace on TFIDF normalized data. The data used are email data where 20% of the data are used for training.

We next turn to the HMM model with LSI estimated emission probabilities. We again take a look at the transition probabilities for the three classes in the email-set, for discovering whether there is a sequential difference between the three classes that is captured by the model. In Figure 4 an illustration of the transition probabilities is shown. The Illustration shows transition probabilities for a model with 20 states, where the best model instead uses about 120 states. The smaller model is shown while it is easier to survey.



(a) Job (b) Conference (c) Spam

Figure 4: Graphical illustration of the transition probabilities for the LSI-HMM model. The group formations for the LSI-HMM state space is harder to discover than those for the HLG model. There is however small groupings like state 1 and 3 for the Job category (a).

The formation of state groups is not as obvious as it was for the HLG model. It is therefore less obvious whether or not, the LSI-HMM model has captured much sequential information about the documents.

In Figure 5 we show the learning curves for the LSI-HMM compared with the generic LSI model. The LSI model performs slightly better than the LSI-HMM model when used for classification. The performance of the two models follow along for the whole range of training set sizes.

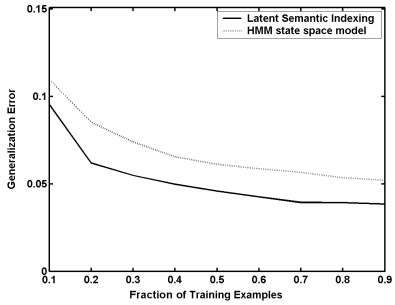


Figure 5: Learning curves for the LSI HMM model. The LSI-HMM model is slightly worse at classifying documents correctly than the generic LSI model.

7 Discussion

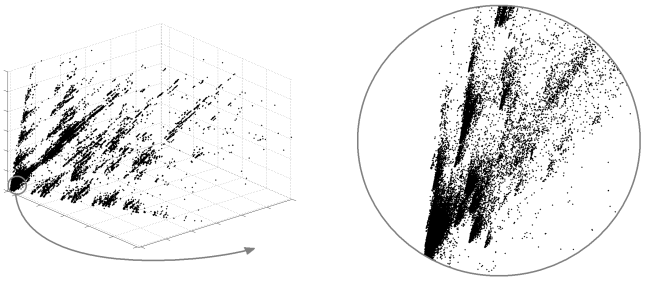
The first approach to capture information from the word sequences in documents, the HLG model did not perform well at the classification task. The state transition probabilities however showed that word order information was captured in the model. The reason for the lack of classification performance is therefore not to be found in the use of the state space model, but rather in the transformation of the vocabulary to a lower dimensional LSI-GMM vocabulary.

Previous experiments has shown that the 50 or more LSI components are needed to create an efficient classifier. It is therefore likely that valuable information is lost when we only use 4 LSI principal component directions here. The reason for only using few LSI components is that the use of many components makes it hard for the GMM to model the semantic space correctly. As illustrated in Figure 6, the density in the LSI subspace is very high in some areas, and the clusters are not very gaussian in shape. A very high amount of gaussian mixtures is therefore needed if they should cover a higher dimensional subspace. In practise the gaussian mixtures are poor at modeling the new LSI subspace, while they tend to cluster around high density areas when too many LSI dimensions are used. This gives a bad fit to many of the outer data-points, resulting in poor classification performance.

The HMM model with LSI estimated emission probabilities was much better at classifying documents correctly than the HLG model. The state transition probabilities did however not seem to capture any valuable information about the differences in word sequences for the three classes. It is likely that a true EM estimate of emission probabilities would have resulted in different transition probabilities that would capture more of the word order information, leading to better classification. A true EM estimate of shared latent topic emissions will however require that the a new HMM must be redefined and update rules determined.

8 Conclusion

We have used two state space model approaches to capture the information, that is contained in the order in which words appear in documents. The first approach involved a transformation of the document vocabulary, into a smaller LSI vocabulary, whereon a HMM could be trained. This approach lacked in classification ability but was conceptual



(a) full space (b) zoom

Figure 6: Zooming in on the LSI space of the document substrings. Some of the spaces are very dense on data, making the areas very attractive for the gaussian mixtures. When using high dimensional representations of the LSI substring space, the outer data points therefore tend to be badly modeled. Since much variation exists for the data in non dense areas, lack of modeling in these areas are likely to result in loss of information.

successful at capturing word order information. The second approach involved making an estimate of latent topic emission probabilities for at HMM using LSI. This approach had less success at capturing word order information, but was better at the classification task. We have hope that the HMM approach will have greater success by the development of a HMM with shared latent topic emission probabilities.

References

Bell, A., & Sejnowski, T. (1995a). Blind separation and blind deconvolution: An information-theoretic approach. *International Conference on Accoustics Speech and Signal Processing (ICASSP)* (pp. 3415–3418). Detroit, US.

Bell, A., & Sejnowski, T. (1995b). An information maximization approach to blind separation and blind deconvolution. *Neural Computation, 7*, 1129–1159.

Blei, D., Ng, A., & Jordan, M. (2002). Latent dirichlet allocation. *Advances in Neural Information Processing Systems 14* (pp. 601–608). Cambridge, MA: MIT Press.

Blei, D., Ng, A., & Jordan, M. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research, 3*, 993–1022.

Deerwester, S., Dumais, S., Landauer, T., Furnas, G., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science, 41*, 391–407.

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society Series B, 39*, 1–38.

Furnas, G., Deerwester, S., Dumais, S., Landauer, T., Harshman, R., Streeter, L., & Lochbaum, K. (1988). Information retrieval using a singular value decomposition model of latent semantic structure. *The 11th International Conference on Research and Development in Information Retrieval* (pp. 465–480). Grenoble, France: ACM Press.

Hofmann, T. (1999). Probabilistic latent semantic indexing (PLSI). *Proceedings of the*

22nd Annual ACM Conference on Research and Development in Information Retrieval (pp. 50–57). Berkeley, California: ACM.

Kolenda, T., Hansen, L., Larsen, J., & Winther, O. (2002). Independent component analysis for understanding multimedia content. *Proceedings of IEEE Workshop on Neural Networks for Signal Processing XII* (pp. 757–766). Piscataway, New Jersey: IEEE Press.

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401*, 789–792.

Lee, D. D., & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems 13* (pp. 556–562). Cambridge, MA: MIT Press.

Madsen, R., Hansen, L., & Winther, O. (2003). Singular value decomposition and principal component analysis, isp technical report.

Molgedey, L., & Schuster, H. (1994). Separation of independent signals using time-delayed correlations. *Physical Review Letters*, *72*, 3634–3637.

Rabiner, L. (1989). A tutorial on hidden markov models. *Proceedings of the IEEE*, *77*, 257–286.

Rabiner, L., & Juang, B. (1986). An introduction to hidden markov models. *IEEE ASSP Magazine*, *74*, 4–15.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*, 613–620.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*, 1–47.

Xu, W., Liu, X., & Gong, Y. (2003). Document clustering based on non-negative matrix factorization. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (pp. 267 – 273). Toronto, CA: ACM press.

Zimmermann, M., & Bunke, H. (2004). Optimizing the integration of a statistical language model in hmm based online handwritten text recognition. *Proceedings of the 17th International Conference on (ICPR'04)* (pp. 541–544).

Appendix G

Multi-Subject fMRI Generalization with Independent Component Representation

Multi-Subject fMRI Generalization with Independent Component Representation

Rasmus E. Madsen[1]

Technical University of Denmark, Kgs. Lyngby DK-2800, Denmark,
rem@imm.dtu.dk,
http://www.imm.dtu.dk/∼rem

Abstract. Generalizability in a multi-subject fMRI study is investigated. The analysis is based on principal and independent component representations. Subsequent supervised learning and classification is carried out by canonical variates analysis and clustering methods. The generalization error is estimated by cross-validation, forming the so-called learning curves. The fMRI case story is a motor-control study, involving multiple applied static force levels. Despite the relative complexity of this case study, the classification of the 'stimulus' shows good generalizability, measured by the test set error rate. It is shown that independent component representation leads to improvement in the classification rate, and that canonical variates analysis is needed for making generalization cross multiple subjects.

Keywords: Independent Component Analysis (ICA), functional Magnetic Resonance Imaging (fMRI), Canonical Variates Analysis (CVA), Principal Component Analysis (PCA), Multiple Subjects.

1 Introduction

Biomedical signals, that originate from physiological processes, are in general difficult to measure isolated. Especially when non-invasive measuring techniques are used. The signals measured from the body are often a mixture of signals from different physiological processes, contaminated with noise and artifacts from the data acquisition equipment. This is also the case when we here are analyzing neuroimages, estimated by use of functional magnetic resonance imaging (fMRI). fMRI signals measured from the brain further has the disadvantage of being high dimensional and highly correlated, due to the high degree of connectivity in the brain.

From the neuroimages we seek to reveal knowledge, giving us the opportunity to model the functionality of the brain. To complete this task, it is essential to isolate the interesting macroscopic spatial and temporal patterns of brain activation, to create a reliable model. For this model to be interesting, generalizability across subjects must be adapted into the model, so one group of subjects also can be used to interpret another group of subjects. Due to the problems mentioned, the task of generating reliable generalizable models is a non-trivial task.

Multivariate statistical tools that can help us understand the brain activation patterns is therefore topic of great interest.

Independent Component Analysis (ICA) has been applied to different biomedical signals, but only recently to Functional Magnetic Resonance Imaging (fMRI) [23]. Many experiments where ICA has been applied to fMRI, has been binary experiments, where the subjects are either exposed or not exposed to some stimuli, see eg. [12]. In the experiments presented here, the subjects are exposed to different degree of stimuli. The following classification of the experiment results, therefore falls into multiple classes. At the same time the experiment is performed by multiple subjects, making classification based on group inference. We here examine how the classification generalization performance is affected by choice of an ICA representation instead of the often used representation based on PCA.

2 Functional Magnetic Resonance Imaging

fMRI is a sub-species of Magnetic Resonance Imaging (MRI) techniques. In MRI, the difference in magnetic susceptibility in different tissue in the human body, is used as a non-invasive technique, to localize different body structures. In fMRI, the difference in magnetic susceptibility, in De-oxygenated hemoglobin (HbR) and oxygenated hemoglobin (HbO_2) is used determine changes in blood-flow [11]. The Blood Oxygen Level Dependent (BOLD) contrast is the most common signal, used to determine blood-flow changes, and is therefore a indirect measurement of brain-areas with neural activity.

The measured fMRI signal has many sources, that originate from various physiological processes, including processes that are not related to experiment stimuli. The most prominent confound signal components originates from the cardiac (about 1Hz) and the physiological respiratory signal (about 0.3Hz). Artifacts from eye and body movement also influences the measured signals. The sampling frequency used in this paper and many other fMRI experiments, is well below 1Hz. This implies that some of the confound signal components becomes aliased, resulting in non-trivial temporal behavior for these confounds. On top of physiological confounds, also noise from the data acquisition equipment occur in the data.

Apart from the confounds, the signals we want to measure are not ideal. This is because the response in blood-flow to the neural active areas in the brain is not instant. The response of the blood-flow is described by the Hemodynamic Response Function (HRF) [11]. The HRF is the theoretical impulse response, that BOLD fMRI measures, when a subject is exposed to a very short stimulus. In [7] it is shown that differences in HRF time-to-peak values, varies from 2.7 to 6.2 sec. In [10] it is shown, that the HRF to the same type of subject-stimuli varies. It is also shown that the HRF may vary due to trial, site, stimulus and subject. It is not easy to make a reliable model of the HRF. There has been a lot of effort trying to model the HRF, see eg. [6], but a complete model has not yet been discovered. In this paper we try to eliminate the effect of the HRF on

our experiment signals, by simply removing the samples, where the HRF takes place. It is reasonable to believe that by removing 8 seconds of the samples, before and after subject stimuli, the effect of HRF will be eliminated. Our reason for eliminating the HRF is that neural networks applied to the fMRI data, waste too much effort trying to model the nonlinearities in the HRF, instead of the modelling the underlying stimuli function. It is reasonable to believe that other nonlinear models will have similar problems with the HRF. The confounds and the HRF makes data analysis of fMRI signals, a non-trivial task.

3 Neuroimaging data acquisition and Preprocessing

The fMRI data are acquired during a motor control study, where 16 involved subjects were doing a static force task. In the experiment the subjects were to apply a static force to a pressure gauge, using right hand thumb and index-finger. Following a visual cue, the subjects were to apply five different force levels (200,400,600,800,1000g) to the pressure gauge. The order of the force-levels was randomized. The subjects could visually monitor the force-level on the pressure gauge during the experiment. Between each force level, there was a baseline resting period. The baseline and force periods were approximately 10 TR's (TR = 4 seconds). The experiment was carried out on a Siemens 1.5T clinical scanner (fMRI: EPI BOLD, TR/TE=3986/60 m.sec., FOV=22×22×15cm, slices=30, voxels=3.44×3.44×5.00mm, MRI: T1-weighted 3D FLASH).

The scans from the 16 different subjects has been aligned, using AIR1 and AIR7 six-parameter rigid body transformation with 5th order polynomial warp to a reference MRI [20] [19]. The alignment reduces the inter-subject variance, hence increases the generalizability. The data has following been spatial smoothed with a 2D Gaussian kernel (FWHM = 0 or 6.0 pixels). The voxel time series were de-trended using linear basis of cosine basis functions.

4 Modelling

The brain activation is modelled by the relationship between the subject stimulus and the fMRI response. This is carried out by the joint probability distribution $p(X, G)$ between the microscopic variables X and the macroscopic variables G. The macroscopic variables covers the whole experimental setup that is used during the data acquisition, including the subject stimulus. The microscopic data are the observations measured during the experiment.

When modelling the joint distribution, two approaches can be chosen see eg. [18]. The joint distribution $p(X, G)$ can be factorized into either $p(X|G)p(G)$ or $p(G|X)p(X)$. With $p(X|G)p(G)$, $p(X|G)$ is modelled as a high dimensional conditional density estimate in the space of the macroscopic data. In the other approach, $p(G|X)p(X)$, the dependency $p(G|X)$ is modelled as a low dimensional conditional density estimate. In the ladder approach the dimension of X is reduced, leaving the conditional density estimate in much lower dimension than the first approach. The ladder approach has been used here.

5 Representation and data reduction

In this study, two representation (PCA and ICA) for the fMRI data are used. Both representations are obtained by modelling $p(X)$ by use of unsupervised learning, based on generative models of the form (1).

$$p(X) = \int p(X|S, A)p(S)dS \tag{1}$$

Where $p(X|S, A) = \delta(X - AS)$ is the observation model and $p(S)$ is the source distribution. For PCA, the source distributions 1st and 2nd order moments are uncorrelated. For ICA also higher order moments are uncorrelated.

Principal Component Analysis is carried out by Singular Value Decomposition (SVD). PCA applied to the $V \times T$ matrix X, where V is the number of voxels and T is the time.

$$X = U\Lambda V^T, \qquad X_{m,n} = \sum_{i=1}^{T} U_{m,i}\Lambda_{i,i}(V^T)_{i,n} \tag{2}$$

Where U is a $M \times N$ matrix, and Λ, V are $N \times N$ matrixes. Λ is a diagonal matrix containing the singular values, arranged by size. U contains the eigenvectors corresponding to the eigenvalues of XX^T, in the columns. V contains the eigenvectors corresponding to the eigenvalues of X^TX, in the rows. The dimension of X is reduced from T to K, by simply using only the K first columns of U and the first K rows of V^T as representation.

The ICA can be applied to the fMRI in either spatial or temporal domain, to produce either independent time-series or independent image components. The general ICA decomposition is defined in eq. 3, where X is a $M \times N$ matrix containing the fMRI.

$$X = AS, \qquad X_{m,n} = \sum_{i=1}^{K} A_{m,i}S_{i,n} \tag{3}$$

Where A is a matrix of image columns and S the corresponding matrix of time-series. When doing spatial ICA the columns of A becomes independent, and similarly the rows of S becomes independent when temporal ICA is applied. Temporal ICA is can be defined:

$$Y \equiv U^TX = U^TAS \equiv BS \tag{4}$$

Where Y is the $N \times N$ matrix containing the PCA time-series and S are the independent time-series. On the other hand we can define spatial ICA by the transformation:

$$Y^T \equiv V^TX^T = V^TS^TA^T \equiv (BS)^T \tag{5}$$

Here Y is the $N \times M$ matrix containing the PCA images and S are the independent images. Both transformations (Spatial and Temporal) are simple re-writings of the separation problem, and no loss of generality is generality is introduced.

The spatial and temporal ICA approaches should probably not compete against each other but could be used together. The independent time series should be used to model the paradigm. It is most likely that the independent time series will model the paradigm best. The images that are associated with the independent time series, will model multiple areas in the brain, that are active with the paradigm. The independent images will model volumes in the brain that are independent. These places are most likely isolating the functional different places in the brain, that are used during the experiment. The brain area for vision would fx. follow the experiment paradigm, but would also be influenced by the eye flickering. In [3] fMRI data was analyzed searching for temporal independent activation sequences. Here temporal ICA was able to separate two induced effects and CO_2 inhalation (hypercapnia). Since hypercapnia induces a global spatial effect, temporal independence is more appropriate than spatial independence.

ICA can be carried out by various algorithms. Different assumptions has led to multiple approaches for solving the ICA problem. In [12] three ICA algorithms were compared. The first approach is using De-correlation techniques, which was first proposed by Molgedey and Schuster [14]. The algorithm was later enhanced [15][16][22], eliminating the limitations in the original algorithm and applying a delay estimate. The second approach is the info-max algorithm [2][1]. The info-max algorithm maximizes the information-transfer through a artificial neural network (ANN), thereby separating independent components. This approach can also be seen from a maximum likelihood point of view [5]. The info-max approach needs a probability density function (PDF) estimate of the components to make a correct estimate of the independent components. In the first algorithms, the components was just expected to have the same PDF's, often super-Gaussian. Later the info-max algorithm was extended [24], enabling it to distinguish between either a super- or sub-Gaussian PDF. In the third approach, Dynamic Component Analysis (DCA) [8] [9], the assumptions from the De-correlation and information-maximization algorithms are combined into one single algorithm. The three algorithms were compared using spatial and temporal modes and shown to produce consistent spatial activation maps and corresponding time-series. There are lots of algorithms to choose from, each having different advantages over each other. In the following, the enhanced De-correlation algorithm has been used [21], due to low computational time, which has been important due to size of the fMRI data sets.

6 Subject Inference and Classification

When using PCA and ICA for preprocessing the fMRI data, inference between subjects is not achieved. Especially the ICA algorithm separates the signals associated with different subjects. The ICA algorithm makes the signals independent resulting in independent signals for different subjects. when wanting to make classification of fMRI data with multiple subjects, group inference is needed. Canonical Variates Analysis (CVA), can create group inference based

on labelling. The objective in CVA is to find a linear transformation of two data sets, x and y, so that the transformed data-sets have the largest possible correlation [13]. We here use the CVA to find the largest possible correlation between the labels and the ICA components. In figure 1 lack of group inference for the ICA components is shown together with the CVA components where inter-subject inference is present. The CVA algorithm combines the ICA components

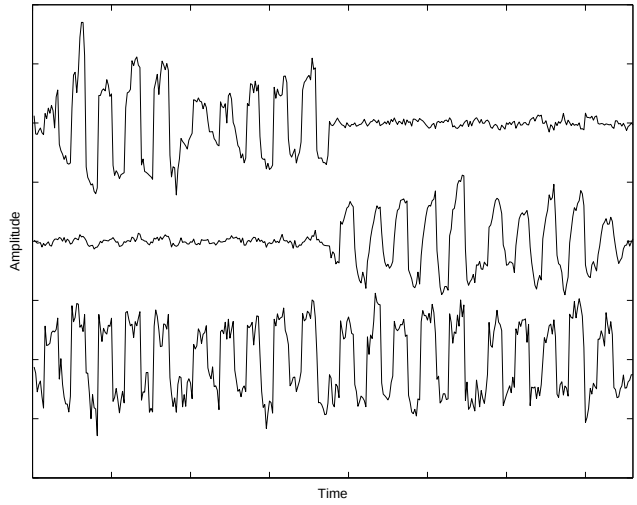


Fig. 1. CVA and ICA components. The ICA algorithm has separated similar components from different subjects. This makes it impossible to make inter-subject learning. The CVA algorithm combines the ICA components into few components that can be used for classification. The CVA algorithm combines the components by maximizing the correlation between the paradigm and ICA components.

by maximizing the correlation between the paradigm and ICA components. Full subject inference can only be achieved for the training data. Subject inference for the test data is only achieved if the training data looks similar to part of the test data. The CVA components can following be used for classification with various clustering algorithms. Due to relative few data points, we here use the N Nearest Neighbors approach, see e.g. [4].

7 Experiments

The fMRI data from the 16 subjects is arranged in a 2D data matrix, where the first dimension is the time and the second dimension is the 3D voxel image

arranged in one dimension. The subjects are stacked in the first dimension, in hope that the subject are activated in the same parts of the brain, during the experiments. The data could also be stacked in the second dimension, if the time activation patterns were expected to be the same. This is not possible in our case, since the static force task is performed in random order. Data reduction is performed by PCA, reducing the fMRI-data from 2984 to 400 components. The no. of relevant components to be found in the fMRI data was first estimated by use of the Bayesian Information Criteria [17]. The result is shown in figure 2.

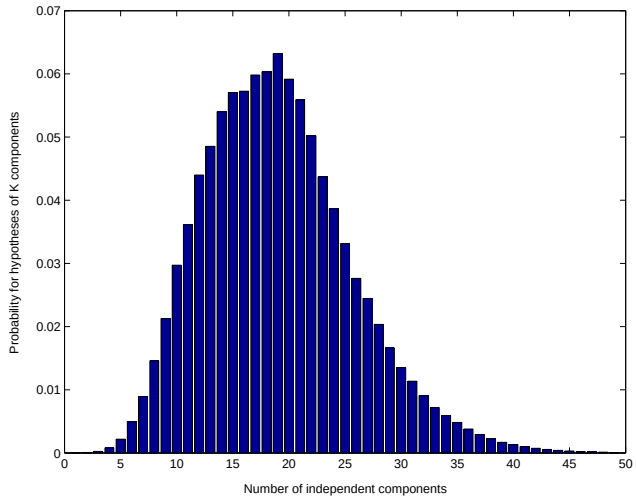


Fig. 2. The Bayesian Information Criteria BIC is applied to determine the no. of relevant components to use from the fMRI data. BIC finds approximately 20 relevant components in the fMRI data. At least 5 times the no. of components found by BIC, must be used to achieve accurate classification.

The BIC settles for about 20 components as the most optimal choice. Unfortunately the no. of components BIC finds optimal, shows very poor performance when CVA and classification is applied. At least 5 times the no. of components found by BIC, must be used to achieve accurate classification. The reason that BIC fails to find the optimal no. of components for classification, should probably be found in the fact that the ICA algorithm separates similar components that emerge in different subjects.

The optimal no. of components is instead found by estimating the bias-variance tradeoff, shown in figure 3.

When using low-dimensional representation of the data the classifier has high error-rates, because the representation is not rich enough, i.e. biased. On

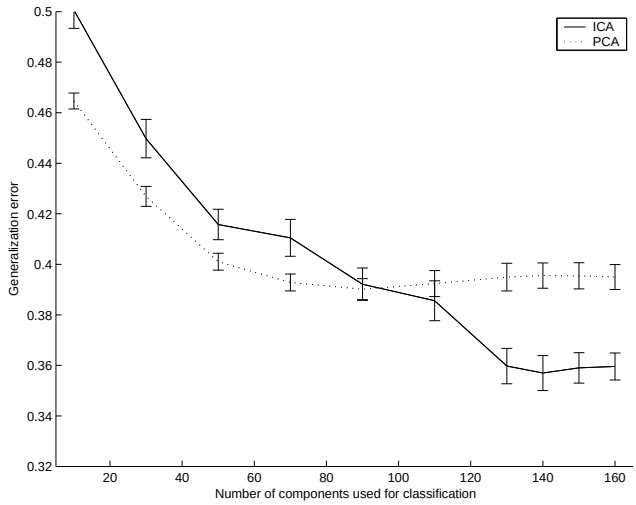


Fig. 3. Bias variance tradeoffs for ICA and PCA. The classification error rates based on test-sets is shown, when the classifier is based on ICA and PCA bases. For low dimensional bases the classifier has high error-rate because the representation is not rich enough, i.e. biased. For to high bases, for PCA D $>$ 90 and ICA D $>$ 140, the test error-rates increases because of the over-fit of the classifier in the high-dimensional representation. When using the best ICA representation, the generalization error is much lower than when using the best PCA representation. It is likely that the ICA algorithm is better suppressing the noise to the lower components, leading to enhanced generalization error.

the other hand, when representation is high-dimensional, the error-rate increase because the classifier over-fits. The best bias-variance tradeoff is approximately 90 components for the PCA representation, and 140 components for the ICA representation.

The generalization error, when using the best ICA representation is much lower than when using the best PCA representation. When using ICA, the components from 90 to 140 can be used for generalization, without over-fitting the model. It is likely that the ICA algorithm is better suppressing the noise to the lower components. This will result in more useful components which can be used to lower the generalization error.

Data spatially smoothed with a 2D gaussian kernel are compared with non-smoothed data. Two different brain warp approaches are also compared [20] [19]. Learning curves [18] for the four combinations are shown in figure 4.

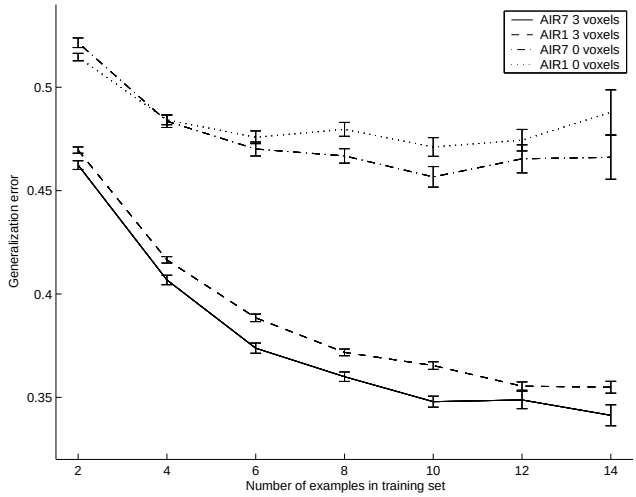


Fig. 4. Learning curves for different smoothing and registration warp method (AIR1 vs. AIR7). Without smoothing the generalization error is high. The effect of the different warp methods is limited, but AIR7 performs better than AIR1. This result was obtained with random re-sampled disjoint training- and test-sets. The classifier is based on CVA with an ICA representation of 140 basis vectors. The error-bars represent the standard deviation of the mean, and are estimated from 100 re-samples.

From figure 4, it is clear that spatial smoothing is a important parameter when making subject inference. The difference in using AIR1 or AIR7 warp technique is not dramatic. The AIR7 warp method is the best though.

Our primary objective is the question of representation. In figure 5 the learning curves for high dimensional PCA and ICA bases are shown. We are considering a six way classifier, making it interesting to see whether the classifier can distinguish between baseline and force, and following to see how well it predicts the actual force level. To be able to tell whether we can make the distinction between baseline and force, the force-level distinction line is introduced in figure 5. The error rate at the force level distinction line is $P = 0.4$, and is calculated by random selection between the force-levels.

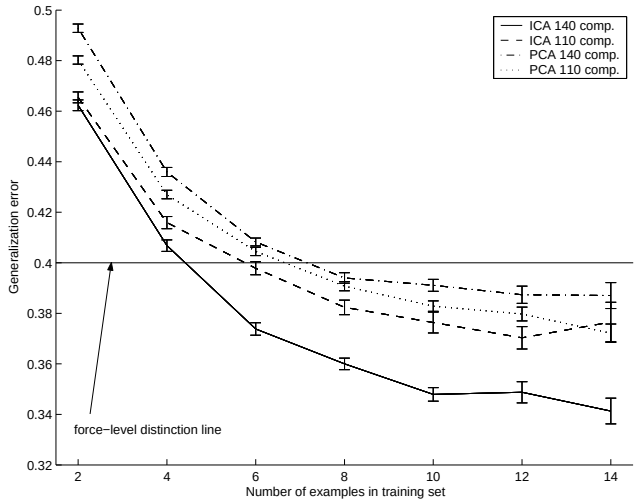


Fig. 5. Learning curves when using ICA and PCA representation with 110 and 140 components. The best performance for the PCA algorithm is found when using 110 components, and for ICA representation using 140 components. The error bars represent the std. of the mean for 100 repetitions. When using the ICA basis, the force-level distinction line is clearly passed. There are still scans though, for which baseline- and force-labels are confused.

Even though the ICA basis clearly passed the force-level distinction line in figure 5 , there are scans for which baseline- and force-labels are confused. All four representations has lower generalization error than the force-level distinction line, when using a sufficient amount of training examples, i.e. some knowledge about the force-level is preserved for all representations. Using the force-level distinction line as offset, the best ICA basis is clearly the best representation.

The distribution of the errors for the ICA representation with 140 components, is further elaborated in figure 6. Here the different types of errors that occurs in the classification experiment are shown. The figure shows that when

the classifier makes an classification error, the correct force level is typically not far away in terms of force level. To specify this, the errors are compared with baseline probabilities assuming that the classifier would only be able to distinguish between force and baseline states.

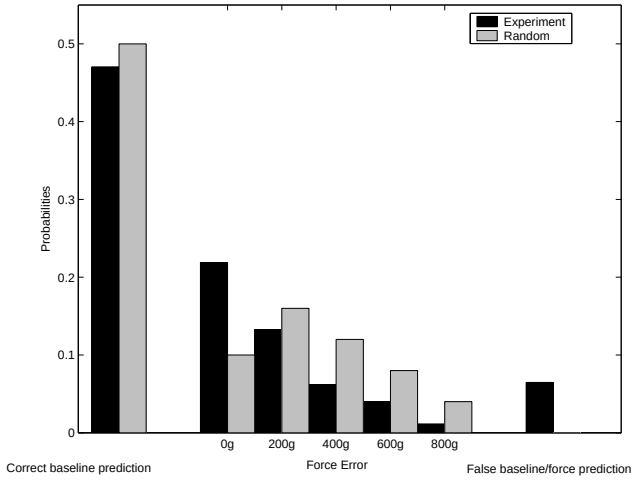


Fig. 6. Error distribution for the multi-subject experiment. The errors are compared with baseline probabilities assuming that the classifier would only be able to distinguish between force and baseline states. The re-sampling experiment is based on a CVA classifier using an ICA basis with 140 vectors. We used 14 training subjects and AIR7 warp with smoothing corresponding to 3 voxels. The 'false force/baseline' distinction is used to indicate scan classifications where the subject is in the resting baseline state but the classifier outputs a force level label or vise versa. The 'correct baseline', are the scans for which the scan is correctly estimated to be resting. The 'force error' is the difference in grams predicted by the classifier, hence zero 'force error' indicates that the force level is estimated correct.

The classifiers ability to predict the force states is further illustrated in figure 7, where the reference activation function (the 'paradigm') is compared with the classifier predictions.

In figure 8, a 3D model of the most salient spatial activation regions in the brain are shown. The regions are based on the two first CVA components. The first component forms the force baseline discriminant, and the second the force-level discriminant. The statistical means of the two CVA components are shown in figure 9

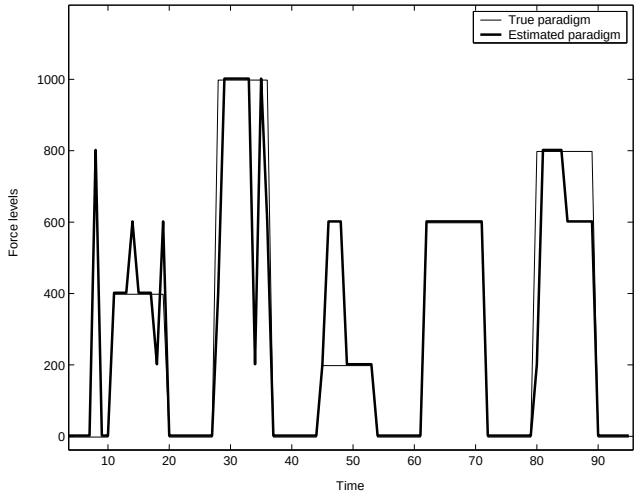


Fig. 7. The predicted activation function, compared with the actual experimental paradigm. The shown predictions are from one of the more easy subjects to predict.

8 Conclusion

Multi subject fMRI analysis based on two different representations, PCA and ICA bases, has been investigated. Based on the two representations, supervised classification been performed using Canonical Variates Analysis. We conclude that ICA allows for higher dimensional representation, providing a less biased estimate, resulting in an improved test-set classification. While the choice of representation is important, spatial smoothing and alignment by warp are still more important determinants for good generalization. We also conclude that it is important apply CVA to the ICA and PCA bases, to get group inference for generalization.

9 Acknowledgement

This work was supported in part by NHI grants NS35273 and MH57180 and by the European Union through the project MAPAWAMO.

References

1. Bell A.J. and Sejnowski T. Blind separation and blind deconvolution: An information-theoretic approach. In *International Conference on Accoustics Speech and Signal Processing (ICASSP)*., volume 5, pages 3415–3418, 1995.

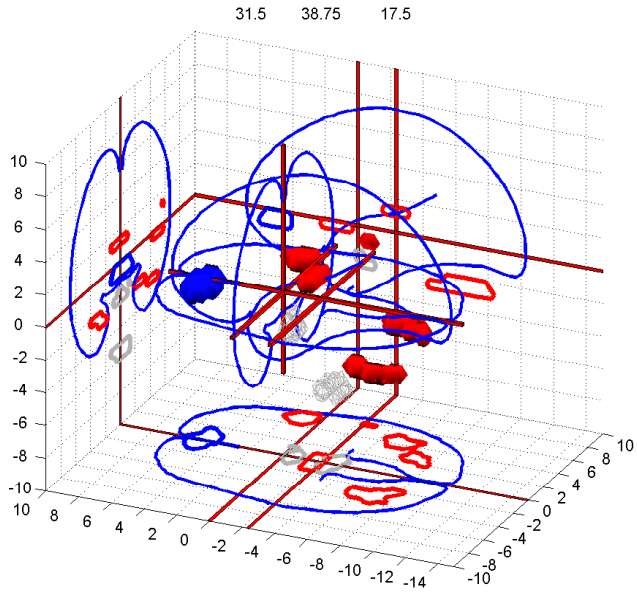


Fig. 8. Corner cube rendering of the most salient spatial pattern found by the CVA model, the dark colored locations are the most active locations for this pattern. These regions primarily encode the discrimination between force application and baseline states. The gray wire-frame regions are the most activated regions for the second most salient CVA patterns. These regions code for the amount of force applied.

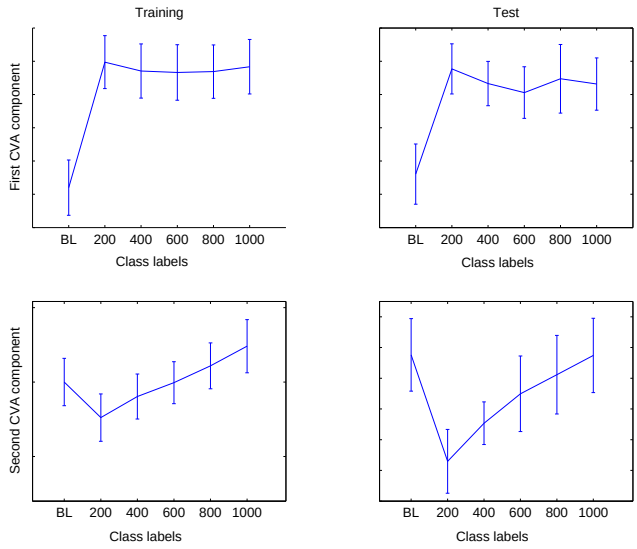


Fig. 9. The mean and std. of two most salient CVA components, for the baseline class and the five force classes. The first component is clearly discriminative between baseline and force activation. The second most salient CVA component encode for the amount of force applied. The CVA components are based on 140 ICA vectors.

2. Bell A.J. and Sejnowski T. An information maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.
3. B.B. Biswal and Ulmer J.L. Blind source separation of multiple signal sources of fmri data sets using independent component analysis. *Journal of Computer Assisted Tomography*, 23(2):265–271, 1999.
4. Bishop C.M. *Neural Networks for Patthern Recognition.* Oxford University Press., 1996. Generel book on patthern recognition methods.
5. MacKay D.J.C. Maximum likelihood and covariant algorithms for independent component analysis., 1996.
6. Marrelec G., Benali H., Ciuciu P., and Poline J.B. Bayesian estimation of the hemodynamic response function in functional mri. In Robert Fry, editor, *Bayesian Inference and Maximum Entropy Methods, Baltimore, MD, USA, MaxEnt Workshop, August*, 2001.
7. Aguirre G.K., Zarahn E., and D'Esposito M. The variability of human, bold hemodynamic responses. *Neuroimage*, 8(4):360–369, 1998.
8. Attias H. and Schreiner C.E. Blind source separation and deconvolution by dynamic component analysis. In *IEEE workshop on Neural Networks for Signal Processing*, volume 7, pages 456–465, 1997.
9. Attias H. and Schreiner C.E. Blind source separation and deconvolution: The dynamic component analysis. *Neural Computation*, 10:1373–1424, 1998.
10. Duann J.R., Jung T.P., Kuo W.J., Yeh T.C., Makeig S., Hsieh J.C., and Sejnowski T.J. Single-trial variability in event-related bold signals. *Neuroimage*, 15(4):823–835, 2002. The hemodynamic responce function (HRF) in BOLD fMRI varies. The HRF from the same type of subject-stimuli varies. It is also shown that the HRF may vary due to trial, site, stimulus and subject.
11. Friston K., Jezzard P., and Turner R. The analysis of functional mri time series. *Human Brain Mapping*, 1:153–171, 1994.
12. Petersen K.S., Hansen L.K., Kolenda T., Rostrup E., and Strother S.C. On the independent components of functional neuroimages. In *ICA-2000, Helsinki, Finland, June 22*, 2000.
13. Mardia K.V., Kent J.T., and Bibby J.M. *Multivariate Analysis.* Academic Press Limited, 1979.
14. Molgedey L. and Schuster H. Separation of independent signals using time-delayed correlations. In *Physical Review Letters*, volume 72, pages 3634–3637, 1994.
15. Hansen L.K. and Larsen J. Source separation in short image sequences using delayed correlation. In *NORSIG'98. 3rd IEEE Nordic Signal Processing Symposium*, pages 253–256. Aalborg Univ, 1998.
16. Hansen L.K., Larsen J., and Kolenda T. On independent component analysis for multimedia signals. In *Multimedia Image and VideoProcessing.* CRC Press., 2000.
17. Hansen L.K., Larsen J., and Kolenda T. Blind detection of independent dynamic components. In *International Conference on Accoustics Speech and Signal Processing (ICASSP).*, 2001.
18. Mrch N., Hansen L.K., Strother S.C., Svarer C., Rottenberg D.A., Lautrup B., Savoy R., and Paulson O.B. Nonlinear versus linear models in functional neuroimaging: Learning curves and generalization crossover. In J. Duncan and G. Gindi., editors, *Proceedings of the 15th International Conference on Information Processing in Medical Imaging*, volume 1230, pages 259–270. Springer Verlag, 1997.
19. Woods R.P., Dapretto M., Sicotte N.L., Toga A.W., and Mazziotta J.C. Creation and use of a talairach-compatible atlas for accurate, automated, nonlinear intersub-

ject registration, and analysis of functional imaging data. volume 8, pages 73–79, 1999.
20. Woods R.P., Grafton S.T., Holmes C.J., Cherry S.R., and Maziotti J.C. Automated image registration: I. general methods and intrasubject, intramodality validation. *Journal of Computer Assisted Tomography*, 22(1):139–152, 1998.
21. Kolenda T. Mole ica matlab toolbox. mole.imm.dtu.dk/toolbox/ica/, 2002.
22. Kolenda T., Hansen L.K., and Larsen J. Signal detection using ica: Application to chat room topic spotting. In *ICA'2001, San Diego, USA, December 9-13*, 2001.
23. Jung T.P., Makeig S., McKeown M.J., Bell A., Lee T.W., and Sejnowski T.J. Imaging brain dynamics using independent component analysis. In *Proceedings of the IEEE*, volume 89, pages 1107–1122, 2001.
24. Lee T.W., Girolami M., and Sejnowski T.J. Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources. *Neural Computation*, 11(2):409–433, 1999.

Bibliography

Cognitive science laboratory at princeton university (2003), wordnet 2.0. http://www.cogsci.princeton.edu/ wn/.

Ibm almaden research center (2003), the WebFountain. http://www.almaden.ibm.com /webfountain/publications/.

Aizawa, A. (2001). Linguistic techniques to improve the performance of automatic text categorization. *Proceedings of NLPRS-01, 6th Natural Language Processing Pacific Rim Symposium* (pp. 307–314). Tokyo, JP.

Aizawa, A. (2003). An information-theoretic perspective of tf-idf measures. *Information Processing and Management*, *39*, 45–65.

Andrieu, C., de Freitas N., Doucet, A., & M., J. (2003). An introduction to mcmc for machine learning. *Machine Learning*, *50*, 5–43.

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., & Spyropoulos, C. D. (2000). An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval* (pp. 160–167). Athens, GR: ACM Press, New York, US.

Apte, C., Damerau, F. J., & Weiss, S. M. (1994). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, *12*, 233–251.

Banerjee, A., Merugu, S., Dhillon, I., & J., G. (2005). Clustering with bregman divergences. *to appear in Journal of Machine Learning Research*, *6*.

Basili, R., & Moschitti, A. (2001). A robust model for intelligent text classification. *Proceedings of ICTAI-01, 13th IEEE International Conference on Tools with Artificial Intelligence* (pp. 265–272). Dallas, US: IEEE Computer Society Press, Los Alamitos, US.

Basili, R., Moschitti, A., & Pazienza, M. (2001). NLP-driven IR: Evaluating performances over a text classification task. *Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence* (pp. 1286–1291). Seattle, US.

Beal, M. (2003). Variational algorithms for approximate bayesian inference. PhD. Thesis, Gatsby Computational Neuroscience Unit, University College London.

Bell, A., & Sejnowski, T. (1995a). Blind separation and blind deconvolution: An information-theoretic approach. *International Conference on Accoustics Speech and Signal Processing (ICASSP)* (pp. 3415–3418). Detroit, US.

Bell, A., & Sejnowski, T. (1995b). An information maximization approach to blind separation and blind deconvolution. *Neural Computation*, *7*, 1129–1159.

Bergman, M. K. (2001). The deep web: Surfacing hidden value. *The Journal of Electronic Publishing from the University of Michigan*, *7*.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American*, *284*, 34–43.

Bigi, B. (2003). Using Kullback-Leibler distance for text categorization. *Proceedings of ECIR-03, 25th European Conference on Information Retrieval* (pp. 305–319). Pisa, IT: Springer Verlag.

Bishop, C. (1996). *Neural networks for patthern recognition*. Oxford University Press.

Blei, D., Ng, A., & Jordan, M. (2002). Latent dirichlet allocation. *Advances in Neural Information Processing Systems 14* (pp. 601–608). Cambridge, MA: MIT Press.

Blei, D., Ng, A., & Jordan, M. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Bridle, J. (1990). Probabilistic intrepretation of feedforward classification network outputs with relationship to statistical pattern recognition. *Neurocomputing - Algorithms, Architectures and Applications*, *6*, 227–236.

Chakrabarti, S. (2000). Data mining for hypertext: a tutorial survey. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, *1*, 1–11.

Church, K. W., & Gale, W. A. (1995). Poisson mixtures. *Natural Language Engineering*, *1*, 163–190.

CMU-WebKB (1997). The 4 universities data set. http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/.

CMU-WebKB-2240 (1999). A subset of the webkb. http://www.imm.dtu.dk/∼rem/.

Cohn, D., & Hofmann, T. (2001). The missing link - a probabilistic model of document content and hypertext connectivity. *Advances in Neural Information Processing Systems*. MIT Press.

De Buenaga Rodríguez, M., Gómez-Hidalgo, J. M., & Díaz-Agudo, B. (1997). Using WordNet to complement training information in text categorization. *Proceedings of RANLP-97, 2nd International Conference on Recent Advances in Natural Language Processing*. Tzigov Chark, BL.

Debole, F., & Sebastiani, F. (2003). Supervised term weighting for automated text categorization. *Proceedings of SAC-03, 18th ACM Symposium on Applied Computing* (pp. 784–788). Melbourne, US: ACM Press, New York, US.

Deerwester, S., Dumais, S., Landauer, T., Furnas, G., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, *41*, 391–407.

Dellaert, F. The expectation maximization algorithm, tech. report git-gvu-02-20, college of computing gvu center, georgia institute of technology, february 2002.

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society Series B*, *39*, 1–38.

Furnas, G., Deerwester, S., Dumais, S., Landauer, T., Harshman, R., Streeter, L., & Lochbaum, K. (1988). Information retrieval using a singular value decomposition model of latent semantic structure. *The 11th International Conference on Research and Development in Information Retrieval* (pp. 465–480). Grenoble, France: ACM Press.

Girolami, M., & Kaban, A. (2003). On an equivalence between plsi and lda. *26th Annual ACM Conference on Research and Development in Information Retrieval, SIGIR 2003* (pp. 433–434).

Greiff, W. R. (1998). A theory of term weighting based on exploratory data analysis. *Proceedings of 21st International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 11–19). New York, US: ACM Press.

Guha, R., & McCool, R. (2003). Tap: A semantic web platform. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, *42*, 557–577.

Hansen, L., & Larsen, J. (1998). Source separation in short image sequences using delayed correlation. *NORSIG'98. 3rd IEEE Nordic Signal Processing Symposium.* (pp. 253–256). Aalborg, DK: Aalborg Univ.

Hofmann, T. (1999). Probabilistic latent semantic indexing (PLSI). *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval* (pp. 50–57). Berkeley, California: ACM.

Horn, R. A., & Johnson, C. R. (1990). *Ch. 5 in matrix analysis.* Cambridge University Press.

Huang, L. (2000). A survey on web information retrieval technologies.

Ishii, G., & Hayakawa, R. (1960). On the compound binomial distribution. *Annals of the Institute of Statistical Mathematics*, *12*, 69–80.

Jansche, M. (2003). Parametric models of linguistic count data. *41st Annual Meeting of the Association for Computational Linguistics* (pp. 288–295). Sapporo, Japan.

Jensen, J. L. W. V. (1906). Sur les fonctions convexes et les inegalites entre les valeurs moyennes. *Acta Mathematica*, *30*, 175–193.

Johnson, N., Kotz, S., & Balakrishnan, N. (1997). *Discrete multivariate distributions.* Wiley.

Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, *28*, 11–21.

Jordan, M., Ghahramani, Z., Jaakkola, T., & Saul, L. (1999). An introduction to variational methods for graphical models. *Machine Learning*, *37*, 183–233.

Katz, S. M. (1996). Distribution of content words and phrases in text and language modelling. *Natural Language Engineering*, *2*, 15–59.

Kehagias, A., Petridis, V., Kaburlasos, V. G., & Fragkou, P. (2003). A comparison of word- and sense-based text categorization using several classification algorithms. *Journal of Intelligent Information Systems*, *21*, 227–247.

Kolenda, T. (2002). Multimedia dataset. http://mole.imm.dtu.dk/faq/MMdata/.

Kolenda, T., Hansen, L., Larsen, J., & Winther, O. (2002). Independent component analysis for understanding multimedia content. *Proceedings of IEEE Workshop on Neural Networks for Signal Processing XII* (pp. 757–766). Piscataway, New Jersey: IEEE Press.

Kosala, R., & Blockeel, H. (2000). Web mining research: A survey. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM* (pp. 1–15). ACM Press.

Kullbach, S., & Leibler, R. (1951). On information and sufficiency. *Annals of mathematical Statistics*, *22*, 79–86.

Kvam, P., & Day, D. (2001). The multivariate polya distribution in combat modeling. *Naval Research Logistics*, *28*, 1–17.

Lagus, K., Kaski, S., & Kohonen, T. (2004). Mining massive document collections by the websom method. *Information Sciences*, *163*, 135–156.

Larsen, J., Hansen, L., Have, A., Christiansen, T., & Kolenda, T. (2002). Webmining: learning from the world wide web. *Computational Statistics and Data Analysis*, *38*, 517–532.

Lawrence, S. (2001). Online or invisible. *Nature*, *411*, 521.

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401*, 789–792.

Lee, D. D., & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems 13* (pp. 556–562). Cambridge, MA: MIT Press.

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. *Proceedings of ECML-98, 10th European Conference on Machine Learning* (pp. 4–15). Chemnitz, Germany: Springer Verlag, Heidelberg, Germany.

Lyman, P., & Varian, H. R. (2003). How much information 2003. http://www.sims.berkeley.edu.

Madsen, R., Hansen, L., & Winther, O. (2003). Singular value decomposition and principal component analysis, isp technical report.

Madsen, R., Kauchak, D., & Elkan, C. (2005). Modeling word burstiness using the dirichlet distribution. *to appear in the Proceedings of ICML-05.*

Manning, C., & Schütze, H. (1999). *Foundations of statistical natural language processing.* Cambridge, US: The MIT Press.

Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. (1994). Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, *19*, 313–330.

Mardia, K., Kent, J., & Bibby, J. (1979). *Multivariate analysis.* London, UK: Academic Press Limited.

Mason, O. (2003). Qtag english tagset (statistical based part-of-speech tagger). http://web.bham.ac.uk/o.mason/software/tagger/tagset.html.

Mauldin, M. (1995). Measuring the web with lycos. *Third International World-Wide Web Conference*. http://lazytoad.com/lti/pub/lycos-websize.html.

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification. *AAAI/ICML-98 Workshop on Learning for Text Categorization* (pp. 41–48). AAAI Press.

McCallum, A. K. (1996). *Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering*. www.cs.cmu.edu/~mccallum/bow.

Minka, T. (2003). *Estimating a Dirichlet distribution*. www.stat.cmu.edu/~minka/papers/dirichlet.

Molgedey, L., & Schuster, H. (1994). Separation of independent signals using time-delayed correlations. *Physical Review Letters*, *72*, 3634–3637.

Nielsen, F. (2001). Email data-set. http://www.imm.dtu.dk/∼rem/.

Nigam, K., McCallum, A., Thrun, S., & Mitchell, T. (1998). Learning to classify text from labeled and unlabeled documents. *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence* (pp. 792–799). Madison, US: AAAI Press, Menlo Park, US.

Rabiner, L. (1989). A tutorial on hidden markov models. *Proceedings of the IEEE*, *77*, 257–286.

Rabiner, L., & Juang, B. (1986). An introduction to hidden markov models. *IEEE ASSP Magazine*, *74*, 4–15.

Rennie, J. D. M., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the poor assumptions of naive Bayes text classifiers. *Proceedings of the Twentieth International Conference on Machine Learning* (pp. 616–623). Washington, D.C., US: Morgan Kaufmann Publishers, San Francisco, US.

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., & Stamatopoulos, P. (2001). Stacking classifiers for anti-spam filtering of e-mail. *Proceedings of EMNLP-01, 6th Conference on Empirical Methods in Natural Language Processing* (pp. 44–50). Pittsburgh, US: Association for Computational Linguistics, Morristown, US.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*, 613–620.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. *International Conference on New Methods in Language Processing*. Manchester, UK.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*, 1–47.

Sigurdsson, S. (2002). The dtu: Artificial neural network toolbox. http://mole.imm.dtu.dk/toolbox/ann/.

Sigurdsson, S., Larsen, J., Hansen, L., Philipsen, P., & Wulf, H. (2002). Outlier estimation and detection: Application to skin lesion classification. *International conference on acoustics, speech and signal processing* (pp. 1049–1052).

Sigurdsson, S., Philipsen, P. A., Hansen, L. K., Larsen, J., Gniadecka, M., & Wulf, H. C. (2004). Detection of skin cancer by classification of raman spectra. *IEEE Transactions on Biomedical Engineering*, *51*, 1784 – 1793.

Strother, S., Anderson, J., Hansen, L., Kjems, U., Kustra, R., Siditis, J., Frutiger, S., Muley, S., LaConte, S., & Rottenberg, D. (2002). The quantitative evaluation of functional neuroimaging experiments: The NPAIRS data analysis framework. *Neuroimage*, *15*, 747–771.

Teevan, J., & Karger, D. R. (2003). Empirical development of an exponential probabilistic model for text retrieval: Using textual analysis to build a better model. *Proceedings of the 26th Annual ACM Conference on Research and Development in Information Retrieval (SIGIR '03)* (pp. 18–25). Toronto, CA: ACM.

Tufis, D., & Mason, O. (1998). Tagging romanian texts: a case study for qtag, a language independent probabilistic tagger. *International Conference on Language Resources & Evaluation (LREC)* (pp. 589–596). Granada (Spain).

Xu, W., Liu, X., & Gong, Y. (2003). Document clustering based on non-negative matrix factorization. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (pp. 267 – 273). Toronto, CA: ACM press.

Yang, Y., & Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. *Proceedings of ICML-97, 14th International Conference on Machine Learning* (pp. 412–420). Nashville, US: Morgan Kaufmann Publishers, San Francisco, US.

Zimmermann, M., & Bunke, H. (2004). Optimizing the integration of a statistical language model in hmm based online handwritten text recognition. *Proceedings of the 17th International Conference on (ICPR'04)* (pp. 541–544).

Zipf, G. (1949). *Human behaviour and the principle of least effort: An introduction to human ecology.* Addison-Wesley.

Zurada, J., Malinowski, A., & I., C. (1994). Sensitivity analysis for minimization of input data dimension for feedforward neural network. *Proceedings of the IEEE Symposium on Circuits and Systems* (pp. 447–450).